

INTERNATIONAL STANDARD

NORME INTERNATIONALE



Industrial communication networks – Wireless communication network and communication profiles – WirelessHART™

Réseaux de communications industriels – Réseau de communications sans fil et profils de communication – WirelessHART™



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2010 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de la CEI ou du Comité national de la CEI du pays du demandeur.

Si vous avez des questions sur le copyright de la CEI ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de la CEI de votre pays de résidence.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch

Tel.: +41 22 919 02 11

Fax: +41 22 919 03 00

A propos de la CEI

La Commission Electrotechnique Internationale (CEI) est la première organisation mondiale qui élabore et publie des normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications CEI

Le contenu technique des publications de la CEI est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

- Catalogue des publications de la CEI: www.iec.ch/searchpub/cur_fut-f.htm

Le Catalogue en-ligne de la CEI vous permet d'effectuer des recherches en utilisant différents critères (numéro de référence, texte, comité d'études,...). Il donne aussi des informations sur les projets et les publications retirées ou remplacées.

- Just Published CEI: www.iec.ch/online_news/justpub

Restez informé sur les nouvelles publications de la CEI. Just Published détaille deux fois par mois les nouvelles publications parues. Disponible en-ligne et aussi par email.

- Electropedia: www.electropedia.org

Le premier dictionnaire en ligne au monde de termes électroniques et électriques. Il contient plus de 20 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans les langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International en ligne.

- Service Clients: www.iec.ch/webstore/custserv/custserv_entry-f.htm

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions, visitez le FAQ du Service clients ou contactez-nous:

Email: csc@iec.ch

Tél.: +41 22 919 02 11

Fax: +41 22 919 03 00

INTERNATIONAL STANDARD

NORME INTERNATIONALE



Industrial communication networks – Wireless communication network and communication profiles – WirelessHART™

Réseaux de communications industriels – Réseau de communications sans fil et profils de communication – WirelessHART™

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

PRICE CODE
CODE PRIX

XH

ICS 25.040.40; 35.100.05

ISBN 978-2-88912-590-6

CONTENTS

FOREWORD.....	18
INTRODUCTION.....	20
1 Scope.....	22
2 Normative references	22
3 Terms, definitions, symbols, abbreviations and conventions	23
3.1 Reference model terms and definitions	23
3.2 Terms and definitions from ISO/IEC 7498-1 terms	23
3.2.1 Terms and definitions from ISO/IEC 9545.....	24
3.2.2 Terms and definitions from ISO/IEC 8824.....	24
3.2.3 Terms and definitions from ISO/IEC 8825.....	24
3.3 Specific terms and definitions	24
3.4 Abbreviations.....	33
3.5 Conventions.....	35
3.5.1 Conventions used for service.....	35
3.5.2 Conventions used for DL-protocol (state machine).....	36
3.5.3 Conventions for Application Layer service	36
3.5.4 Conventions for the AL-protocol	39
4 Physical layer specification and service definition.....	44
4.1 Physical layer service definition	44
4.1.1 General overview.....	44
4.1.2 Physical layer services	44
4.2 Physical layer protocol specification.....	48
5 Data-link layer service definition – Type 20 elements	48
5.1 Data-link layer services.....	48
5.1.1 Facilities of the data-link layer services	48
5.1.2 QoS for message service.....	49
5.1.3 Sequence of primitives	50
5.1.4 DL-TRANSMIT service	52
5.1.5 DL-RECEIVE service	54
5.1.6 DL-DISCONNECT service	55
5.1.7 DL-PATH-FAILURE service	55
5.1.8 DL-ADVERTISE service.....	55
5.1.9 DL-NEIGHBOR service.....	56
5.1.10 DLM-SET	56
5.1.11 DLM-GET	57
5.1.12 DLM-ACTION	58
5.1.13 DLM-EVENT.....	59
5.2 Network layer services.....	59
5.2.1 Facilities of the network layer services	59
5.2.2 QoS for message service.....	59
5.2.3 Sequence of primitives	60
5.2.4 NL-TRANSMIT Service.....	60
5.2.5 NLM-SET	61
5.2.6 NLM-GET	62
5.2.7 NLM-ACTION	63
5.3 Transport layer services.....	64

5.3.1	Facilities of the transport layer services	64
5.3.2	QoS for message service	64
5.3.3	Sequence of primitives	65
5.3.4	TL-DATA-TRANSFER Service	66
5.3.5	TL-DATA-EXCHANGE Service	68
5.3.6	TLM-SET	69
5.3.7	TLM-GET	70
6	Data-link layer protocol specification – Type 20 elements	71
6.1	Overview	71
6.1.1	Data-link layer operation	71
6.1.2	Network layer operation	74
6.1.3	Device procedures	76
6.2	Logical link control	76
6.2.1	General DLPDU structure	76
6.2.2	DLPDU-specific structure, encoding and elements of procedure	79
6.2.3	DLPDU Priority and Flow Control	84
6.2.4	Error detection and security	85
6.3	Medium Access Control	86
6.3.1	Function	86
6.3.2	Slot timing	87
6.3.3	Communication tables and buffers	89
6.3.4	Link scheduling	95
6.3.5	MAC operation	100
6.4	DL-management information	112
6.4.1	General	112
6.4.2	DL attributes	112
6.4.3	DLM actions	113
6.5	Network layer	113
6.5.1	Two level model of NL	113
6.5.2	NPDU structure	116
6.5.3	Security function	118
6.5.4	Network layer tables	123
6.5.5	NLE machine	126
6.5.6	NLE elements of procedure	128
6.5.7	NL-management information	131
6.6	Transport layer	133
6.6.1	General	133
6.6.2	TPDU structure	133
6.6.3	Transport pipe table	134
6.6.4	TLE elements of procedure	134
6.6.5	TLE state machines	137
6.6.6	TL-management information	140
7	Application layer service definition – Type 20 elements	141
7.1	Common concepts	141
7.2	Data type ASE	141
7.2.1	Overview	141
7.2.2	Formal definition of data type objects	143
7.2.3	FAL defined data types	145
7.2.4	Data type ASE service specification	148

7.2.5	Summary of data types	148
7.3	Communication model specification	149
7.3.1	Common parameters	149
7.3.2	ASEs	150
7.3.3	ARs	217
7.3.4	Summary of classes	220
7.3.5	Permitted services by AREP role	220
8	Application layer protocol specification – Type 20 elements	221
8.1	Abstract syntax	221
8.2	Transfer syntax	221
8.2.1	Common APDU fields	221
8.2.2	Common APDU structure	222
8.2.3	Device application service-specific APDU structures	225
8.2.4	Layer management service specific APDU structures	249
8.2.5	Gateway and network manager specific APDU structures	291
8.2.6	Network management configuration specific APDU structures	303
8.2.7	Data coding rules	319
8.3	Common procedures	324
8.3.1	Delayed response	324
8.3.2	Response time	327
8.4	FAL service protocol machine (FSPM)	328
8.4.1	General	328
8.4.2	FSPM state tables	328
8.4.3	Functions used by FSPM	330
8.4.4	Parameters of FSPM/ARPM primitives	331
8.5	Application relationship protocol machines (ARPMs)	332
8.5.1	AREP mapping to transport layer	332
8.5.2	Application relationship protocol machines (ARPMs)	333
8.5.3	AREP state machine primitive definitions	335
8.5.4	AREP state machine functions	336
8.6	TL mapping protocol machine (TMPM)	336
8.6.1	TMPM states	336
8.6.2	TMPM state machines	336
8.6.3	Primitives exchanged between transport layer and TMPM	337
8.6.4	Functions used by TMPM	337
9	Communication Profile Family 9 (HART™)	338
9.1	General Overview	338
9.2	Profile 9/1	338
9.3	Profile 9/2	338
9.3.1	Conformance statement	338
9.3.2	Device types	339
9.3.3	Physical layer	342
9.3.4	Data-link layer	346
9.3.5	Application layer	347
Annex A	(normative) Security	355
Annex B	(normative) Wireless procedures	368
Annex C	(normative) Network management	372
Annex D	(normative) Gateway	376

Annex E (normative) Command response codes	381
Annex F (normative) Common tables	390
Annex G (informative) Application of publish data mode and event commands	404
Annex H (informative) Network redundancy	408
Annex I (informative) Network manager implementation	409
Annex J (informative) Gateway implementation	433
Annex K (informative) Handheld device	444
Bibliography	446
Figure 1 – Superframe class example	41
Figure 2 – Inheritance example	42
Figure 3 – Basic aggregation example	43
Figure 4 – Composition aggregation example	43
Figure 5 – Physical layer service sequences	45
Figure 6 – Message service error-free sequences	51
Figure 7 – Message service sequences with errors	52
Figure 8 – Receive only message service	52
Figure 9 – NLL Data transfer service sequences	60
Figure 10 – Data transfer service error-free sequences	65
Figure 11 – Data transfer service sequences with errors	66
Figure 12 – A TDMA slot and superframe	72
Figure 13 – Channel hopping	73
Figure 14 – Wireless network	75
Figure 15 – DLPDU Structure	76
Figure 16 – Address specifier	76
Figure 17 – Construction of 8-octet EUI-64 Address	78
Figure 18 – DLPDU specifier	78
Figure 19 – Slot timing details	87
Figure 20 – DLE tables and relationship	90
Figure 21 – Relationships used for link scheduling	97
Figure 22 – MAC components	100
Figure 23 – TDMA state machine	102
Figure 24 – XMIT state machine	106
Figure 25 – RECV state machine	109
Figure 26 – ACK transmission state machine	111
Figure 27 – Graph routing	114
Figure 28 – Source routing	115
Figure 29 – NPDU Structure	116
Figure 30 – NPDU control octet	116
Figure 31 – Security control octet	118
Figure 32 – NLE tables and relationship	124
Figure 33 – NLE state machine	127
Figure 34 – Routing decision of the NPDU	130

Figure 35 – TPDU Structure	133
Figure 36 – TPDU control octet	133
Figure 37 – Transition diagram of the client data transfer state machine	138
Figure 38 – Transition diagram of the client data exchange state machine	138
Figure 39 – Transition diagram of the server state machine	139
Figure 40 – Data type class hierarchy	142
Figure 41 – Request APDU format from master to slave	222
Figure 42 – Normal response APDU format from slave to master	223
Figure 43 – Command error response APDU format from slave to master	223
Figure 44 – Aggregated command APDU	224
Figure 45 – Windowed trigger mode	236
Figure 46 – Windowed trigger mode with maximum update time expired	236
Figure 47 – Level trigger mode	237
Figure 48 – Coding without identification	319
Figure 49 – Coding of Integer type data	320
Figure 50 – Coding of Unsigned type data	320
Figure 51 – Coding of single precision Floating Point type data	321
Figure 52 – Coding of double precision Floating Point type data	321
Figure 53 – Coding of Date type data	322
Figure 54 – Normal DR operation	325
Figure 55 – Command responses during DR processing	326
Figure 56 – Slaves with multiple DR buffers	327
Figure 57 – State transition diagram of FSPM	328
Figure 58 – State transition diagram of the client ARPM	334
Figure 59 – State transition diagram of the server ARPM	335
Figure 60 – State transition diagram of TMPM	336
Figure 61 – Wireless network components	340
Figure A.1 – Join sequence	358
Figure A.2 – Network layer join procedure	362
Figure A.3 – Data-link layer join procedure	364
Figure A.4 – Network Key change operation	366
Figure B.1 – Neighbor Discovery	369
Figure B.2 – Path failure	370
Figure B.3 – Device leaving the network	371
Figure C.1 – Network manager in wireless network	372
Figure D.1 – Single network access point with clock	377
Figure D.2 – Multiple network access points with clocks	377
Figure D.3 – Network access point not providing clock	378
Figure G.1 – Enabling or disabling publishing sequence	405
Figure H.1 – Network routing	408
Figure I.1 – General model for network manager	410
Figure I.2 – Device type data model	411
Figure I.3 – Network routing	412

Figure I.4 – Network schedule	414
Figure I.5 – Example of a three-slot superframe.....	415
Figure I.6 – Multiple superframes in a network.....	415
Figure I.7 – Security manager.....	417
Figure I.8 – Detailed model of the network manager	418
Figure I.9 – Four network device wireless network	424
Figure I.10 – APDU sequences	429
Figure I.11 – Initializing a wireless network.....	430
Figure I.12 – Allocating and using network resources	431
Figure I.13 – Adjusting network schedule.....	432
Figure I.14 – Health reports	432
Figure J.1 – General model for gateway.....	434
Figure J.2 – Logical network device	435
Figure J.3 – Physical network device	435
Figure J.4 – Managing notification services.....	440
Table 1 – Conventions used for state machines	40
Table 2 – Class relationship notation	42
Table 3 – Multiplicity notation	43
Table 4 – PH-ENABLE primitives and parameters	45
Table 5 – PH-CCA primitives and parameters.....	46
Table 6 – PH-DATA primitives and parameters.....	46
Table 7 – PHM-SET primitive and parameters.....	47
Table 8 – PHM-GET primitive and parameters	48
Table 9 – DL-Transmit primitives and parameters	53
Table 10 – DL-RECEIVE primitive and parameters.....	54
Table 11 – DL-DISCONNECT primitive and parameters.....	55
Table 12 – DL-PATH-FAILURE primitive and parameters.....	55
Table 13 – DL-ADVERTISE primitive and parameters	56
Table 14 – DL-NEIGHBOR primitive and parameters	56
Table 15 – DLM-SET primitive and parameters.....	57
Table 16 – DLM-GET primitive and parameters	57
Table 17 – DLM-ACTION primitive and parameters.....	58
Table 18 – NL-TRANSMIT primitives and parameters	60
Table 19 – NLM-SET primitive and parameters.....	62
Table 20 – NLM-GET primitive and parameters	62
Table 21 – NLM-ACTION primitive and parameters.....	63
Table 22 – TL-DATA-TRANSFER primitives and parameters	66
Table 23 – TL-DATA-EXCHANGE primitives and parameters.....	68
Table 24 – TLM-Set primitive and parameters.....	70
Table 25 – TLM-Get primitive and parameters	70
Table 26 – Network_ID Allocation	77
Table 27 – Contents of the ACK DLPDU payload	80

Table 28 – ACK response-code	80
Table 29 – Contents of the Advertise DLPDU payload	82
Table 30 – Beginning of the superframe.....	82
Table 31 – The next field after the last superframe size	82
Table 32 – Repeated for each link	82
Table 33 – Channel and frequency assignments	83
Table 34 – Slot timing definitions and values	88
Table 35 – Minimum DLE Table and buffer space requirement.....	90
Table 36 – Superframe attributes	91
Table 37 – Link attributes	91
Table 38 – Neighbor attributes and variables	93
Table 39 – Graph attributes	94
Table 40 – Packet record.....	95
Table 41 – Example Backoff_cntr selection sets	99
Table 42 – TDMA state machine	103
Table 43 – XMIT state transitions.....	107
Table 44 – RECV state machine	110
Table 45 – ACK transmission state machine	112
Table 46 – DL attributes.....	112
Table 47 – DL actions	113
Table 48 – Counter value.....	118
Table 49 – Session attributes.....	120
Table 50 – NPDU nonce	121
Table 51 – Minimum NLE Table space requirement	124
Table 52 – Route attributes.....	125
Table 53 – Source route attributes.....	125
Table 54 – TimeTable attributes.....	126
Table 55 – NLE states	126
Table 56 – NPDU construction.....	129
Table 57 – Routing actions	131
Table 58 – NL attributes.....	132
Table 59 – NL actions	132
Table 60 – Transport pipe attributes	134
Table 61 – Transport control encoding for TL-DATA-TRANSFER request	135
Table 62 – Transport control encoding for TL-DATA-EXCHANGE request.....	135
Table 63 – Transport control encoding for TL-DATA-EXCHANGE response	136
Table 64 – Client data transfer state table	138
Table 65 – Client data exchange state table	139
Table 66 – Server state table	140
Table 67 – Data type summary	149
Table 68 – Response code values	149
Table 69 – Read service parameters.....	153
Table 70 – Write service parameters.....	155

Table 71 – Information report service parameters	156
Table 72 – Action service parameters	158
Table 73 – AR get attributes service parameters	218
Table 74 – AR set attributes service parameters	219
Table 75 – Class summary	220
Table 76 – Confirmed services by AREP class	220
Table 77 – Unconfirmed services by AREP class	220
Table 78 – Response code values	221
Table 79 – Extended status values	222
Table 80 – Reset configuration changed flags value field	226
Table 81 – Reset configuration changed flag command specific Response codes	226
Table 82 – Perform self test command specific Response codes	227
Table 83 – Perform device reset command specific Response codes	227
Table 84 – Read additional device status value field	228
Table 85 – Read additional device status command specific Response codes	228
Table 86 – Read additional device status value field	229
Table 87 – Read additional device status command specific Response codes	229
Table 88 – Read device variable information value field	230
Table 89 – Read device variable information response value field	230
Table 90 – Read device variable information command specific Response codes	231
Table 91 – Write device variable value field	231
Table 92 – Write device variable command specific response codes	232
Table 93 – Read real-time clock value field	233
Table 94 – Read real-time clock command specific response codes	233
Table 95 – Write publish data period value field	233
Table 96 – Write publish data period command specific response codes	234
Table 97 – Write publish data trigger field	235
Table 98 – Write publish data trigger command specific response codes	235
Table 99 – Publish data message trigger source	236
Table 100 – Read publish data mode configuration request value field	237
Table 101 – Read publish data mode configuration response value field	238
Table 102 – Read publish data mode configuration command specific response codes	238
Table 103 – Flush delayed responses command specific Response codes	239
Table 104 – Write publish data mode configuration value field	240
Table 105 – Write publish data device variables command specific response codes	240
Table 106 – Write publish data mode command number value field	241
Table 107 – Write publish data mode command number value field	241
Table 108 – Write publish data mode command number command specific response codes	241
Table 109 – Write publish data mode control value field	242
Table 110 – Write publish data mode control command specific response codes	242
Table 111 – Read event notification summary request value field	244
Table 112 – Read event notification summary response value field	244

Table 113 – Read event notification summary command specific response codes.....	244
Table 114 – Write event notification bit mask value field	245
Table 115 – Write event notification bit mask command specific response codes	245
Table 116 – Write event notification timing value field	246
Table 117 – Write event notification timing command specific response codes	246
Table 118 – Write event notification control value field.....	247
Table 119 – Write event notification control command specific response codes	247
Table 120 – Write event notification acknowledgement value field	248
Table 121 – Write event notification acknowledgement command specific response codes.....	248
Table 122 – Write Join_key value field	249
Table 123 – Write Join_key command specific response codes	249
Table 124 – Read join status value field.....	250
Table 125 – Read join status command specific response codes	250
Table 126 – Write Active_search_shed_time request value field	250
Table 127 – Write Active_search_shed_time response value field.....	251
Table 128 – Write Active_search_shed_time command specific response codes	251
Table 129 – Write Join mode configuration value field	252
Table 130 – Write Join mode configuration command specific response codes	252
Table 131 – Read Join mode configuration response value field	253
Table 132 – Read Join mode configuration command specific response codes	253
Table 133 – Write network_ID value field	253
Table 134 – Write network_ID command specific response codes.....	254
Table 135 – Read network_ID value field.....	254
Table 136 – Read network_ID command specific response codes.....	254
Table 137 – Write Network_tag value field	255
Table 138 – Write Network_tag command specific Response codes	255
Table 139 – Read Network_tag response value field.....	256
Table 140 – Read Network_tag command-specific response codes	256
Table 141 – Read wireless device capabilities value field	256
Table 142 – Read wireless device capabilities command specific response codes	256
Table 143 – Read battery life response value field	257
Table 144 – Read battery life command-specific response codes.....	257
Table 145 – Report device health response value field	258
Table 146 – Report device health command specific response codes	258
Table 147 – Read neighbor health list request value field	258
Table 148 – Read neighbor health list response value field.....	259
Table 149 – Read neighbor health list command specific response codes.....	259
Table 150 – Read device nickname response value field	260
Table 151 – Read device nickname command-specific response codes	260
Table 152 – Read session list request value field.....	260
Table 153 – Read session list response value field	260
Table 154 – Read session list command specific response codes.....	261

Table 155 – Read superframe list request value field	261
Table 156 – Read superframe list response value field	261
Table 157 – Read superframe list command specific response codes	262
Table 158 – Read link list request value field	262
Table 159 – Read link list response value field	262
Table 160 – Read link list command specific response codes	263
Table 161 – Read graph list request value field	263
Table 162 – Read graph list response value field	263
Table 163 – Read graph list command specific response codes	263
Table 164 – Read neighbor attribute flag request value field	264
Table 165 – Read neighbor attribute flag response value field	264
Table 166 – Read neighbor attribute flag command specific response codes	264
Table 167 – Read neighbor signal level request value field	265
Table 168 – Read neighbor signal level response value field	265
Table 169 – Read neighbor signal level command specific response codes	265
Table 170 – Report path down alarm response value field	266
Table 171 – Report path down alarm command specific response codes	266
Table 172 – Report source route failed alarm response value field	266
Table 173 – Report source route failed alarm command specific response codes	266
Table 174 – Report graph route failed alarm response value field	267
Table 175 – Report graph route failed alarm command specific response codes	267
Table 176 – Report transport layer failed alarm response value field	267
Table 177 – Report transport layer failed alarm command specific response codes	267
Table 178 – Write UTC time mapping value field	268
Table 179 – Write UTC time mapping command specific response codes	268
Table 180 – Read UTC time mapping response value field	268
Table 181 – Read UTC time mapping command specific response codes	269
Table 182 – Write timer interval value field	269
Table 183 – Write timer interval command specific response codes	269
Table 184 – Read timer interval request value field	270
Table 185 – Read timer interval response value field	270
Table 186 – Read timer interval command specific response codes	270
Table 187 – Write radio power output value field	270
Table 188 – Write radio power output command specific response codes	271
Table 189 – Read radio power output response value field	271
Table 190 – Read radio power output command specific response codes	272
Table 191 – Request TimeTable value field	272
Table 192 – Request TimeTable command specific response codes	273
Table 193 – Read TimeTable list request value field	273
Table 194 – Read TimeTable list response value field	274
Table 195 – Read TimeTable list command specific response codes	274
Table 196 – Delete TimeTable request value field	275
Table 197 – Delete TimeTable response value field	275

Table 198 – Delete TimeTable command specific response codes	275
Table 199 – Read route list request value field	275
Table 200 – Read route list response value field	276
Table 201 – Read route list command specific response codes	276
Table 202 – Read source route request value field	276
Table 203 – Read source route response value field	277
Table 204 – Read source route command specific response codes	277
Table 205 – Read CCA mode response value field	277
Table 206 – Read CCA mode command specific response codes	277
Table 207 – Write CCA mode value field	278
Table 208 – Write CCA mode command specific response codes	278
Table 209 – Read handheld superframe response value field	279
Table 210 – Read handheld superframe command specific response codes	279
Table 211 – Write handheld superframe value field	279
Table 212 – Write handheld superframe command specific response codes	279
Table 213 – Read packet hop-to-live response value field	280
Table 214 – Read packet hop-to-live command specific response codes	280
Table 215 – Write packet hop-to-live value field	280
Table 216 – Write packet hop-to-live command specific response codes	281
Table 217 – Read Join_priority response value field	281
Table 218 – Read Join_priority command specific response codes	281
Table 219 – Write Join_priority value field	282
Table 220 – Write Join_priority command specific response codes	282
Table 221 – Read receive Priority_threshold response value field	282
Table 222 – Read receive Priority_threshold command specific response codes	283
Table 223 – Write receive Priority_threshold value field	283
Table 224 – Write receive Priority_threshold command specific response codes	283
Table 225 – Read device list request value field	284
Table 226 – Read device list response value field	284
Table 227 – Read device list command specific response codes	284
Table 228 – Add device list entry request value field	285
Table 229 – Add device list entry response value field	285
Table 230 – Add device list entry command specific response codes	285
Table 231 – Delete device list entry request value field	286
Table 232 – Delete device list entry response value field	286
Table 233 – Delete device list entry command specific response codes	286
Table 234 – Read channel blacklist response value field	287
Table 235 – Read channel blacklist command specific response codes	287
Table 236 – Write channel blacklist value field	287
Table 237 – Write channel blacklist command specific response codes	288
Table 238 – Read back-off exponent response value field	288
Table 239 – Read back-off exponent command specific response codes	288
Table 240 – Write back-off exponent value field	289

Table 241 – Write back-off exponent command specific response codes	289
Table 242 – Write network access mode value field	289
Table 243 – Write network access mode command specific response codes	290
Table 244 – Read network access mode response value field	290
Table 245 – Read network access mode command specific response codes	290
Table 246 – Request Session request value field	291
Table 247 – Request Session response value field	291
Table 248 – Request Session command specific response codes	291
Table 249 – Read network device identity request value field	292
Table 250 – Read network device identity response value field	292
Table 251 – Read network device identity command specific response codes	292
Table 252 – Read device's neighbor health request value field	292
Table 253 – Read device's neighbor health response value field	293
Table 254 – Read device's neighbor health command specific response codes	293
Table 255 – Read network topology information request value field	293
Table 256 – Read network topology information response value field	294
Table 257 – Read network topology information command specific response codes	294
Table 258 – Read publish data message list request value field	294
Table 259 – Read publish data message list response value field	295
Table 260 – Read publish data message list command specific response codes	295
Table 261 – Flush cached value field	295
Table 262 – Flush cached responses command specific response codes	295
Table 263 – Write update notification bit mask for a device value field	296
Table 264 – Write update notification bit mask for a device command specific response codes	296
Table 265 – Read update notification bit mask for a device request value field	296
Table 266 – Read update notification bit mask for a device response value field	297
Table 267 – Read update notification bit mask for a device command specific response codes	297
Table 268 – Report change notification request value field	297
Table 269 – Report change notification response value field	297
Table 270 – Report change notification command specific response codes	298
Table 271 – Read network device statistics request value field	298
Table 272 – Read network device statistics response value field	299
Table 273 – Read network device statistics command specific response codes	299
Table 274 – Read network device identity request value field	299
Table 275 – Read network device identity response value field	300
Table 276 – Read network device identity command specific response codes	300
Table 277 – Write network device's scheduling flags value field	300
Table 278 – Write network device's scheduling flags command specific response codes	301
Table 279 – Read network device's scheduling flags request value field	301
Table 280 – Read network device's scheduling flags response value field	301
Table 281 – Read network device's scheduling flags command specific response codes	301
Table 282 – Read network constraints value field	302

Table 283 – Read network constraints command specific response codes	302
Table 284 – Write network constraints value field	302
Table 285 – Write network constraints command specific response codes	303
Table 286 – Disconnect device value field	303
Table 287 – Disconnect device command specific response codes	303
Table 288 – Write network key value field	304
Table 289 – Write network key command specific response codes	304
Table 290 – Write device nickname value field	304
Table 291 – Write device nickname command specific response codes	305
Table 292 – Write session request value field	305
Table 293 – Write session response value field	305
Table 294 – Write session command specific response codes	306
Table 295 – Delete session request value field	306
Table 296 – Delete session response value field	306
Table 297 – Delete session command specific response codes	307
Table 298 – Write superframe request value field	307
Table 299 – Write superframe response value field	307
Table 300 – Write superframe command specific response codes	308
Table 301 – Delete superframe request value field	308
Table 302 – Delete superframe response value field	308
Table 303 – Delete superframe command specific response codes	309
Table 304 – Write link request value field	309
Table 305 – Write link response value field	309
Table 306 – Write link command specific response codes	310
Table 307 – Delete link request value field	310
Table 308 – Delete link response value field	311
Table 309 – Delete link command specific response codes	311
Table 310 – Write graph connection request value field	311
Table 311 – Write graph connection response value field	311
Table 312 – Write graph connection command specific response codes	312
Table 313 – Delete graph connection request value field	312
Table 314 – Delete graph connection response value field	312
Table 315 – Delete graph connection command specific response codes	313
Table 316 – Write neighbor attribute flag value field	313
Table 317 – Write neighbor attribute command specific response codes	313
Table 318 – Write network suspend value field	314
Table 319 – Write network suspend command specific response codes	314
Table 320 – Write TimeTable request value field	315
Table 321 – Write TimeTable response value field	315
Table 322 – Write TimeTable command specific response codes	315
Table 323 – Write route request value field	316
Table 324 – Write route response value field	316
Table 325 – Write route command specific response codes	316

Table 326 – Delete route request value field	317
Table 327 – Delete route response value field	317
Table 328 – Delete route command specific response codes	317
Table 329 – Write source route request value field	318
Table 330 – Write source route response value field	318
Table 331 – Write source route command specific response codes	318
Table 332 – Delete source route request value field	319
Table 333 – Delete source route response value field	319
Table 334 – Delete source route command specific response codes	319
Table 335 – Coding of Integer16 type data	320
Table 336 – Coding of Unsigned16 type data	320
Table 337 – Coding for Date type	322
Table 338 – Coding of 1-octet Bit Field type data	322
Table 339 – Coding of 2-octet Bit Field type data	322
Table 340 – Packed ASCII character set	323
Table 341 – Subset of ISO Latin-1 characters	324
Table 342 – DRM related response codes	325
Table 343 – FSPM state Table – client transactions	328
Table 344 – FSPM state Table – server transactions	330
Table 345 – Function TransportType ()	330
Table 346 – Function PriorityType ()	330
Table 347 – Function FormReqApdu ()	330
Table 348 – Function FormRspApdu ()	331
Table 349 – Function RetrieveCommand ()	331
Table 350 – Function CommandErr ()	331
Table 351 – Function ResponseCode ()	331
Table 352 – Function DeviceStat ()	331
Table 353 – Function Value ()	331
Table 354 – Parameters used with primitives exchanged between FSPM and ARPM	332
Table 355 – Client ARPM states	333
Table 356 – Client ARPM state table	334
Table 357 – Server ARPM states	334
Table 358 – Server ARPM state table	335
Table 359 – Primitives issued from ARPM to TPM	335
Table 360 – Primitives issued by TPM to ARPM	336
Table 361 – TPM state descriptions	336
Table 362 – TPM state Table – Client transactions	336
Table 363 – TPM state Table – Server transactions	337
Table 364 – Primitives exchanged between network layer and TPM	337
Table 365 – CPF 9: overview of profile sets	338
Table 366 – CP 9/2: PhL protocol selection	344
Table 367 – Transceiver Specifications	344
Table 368 – Frequency assignments	345

Table 369 – PHY PIB attributes	346
Table 370 – CP 9/2: DLL service selection.....	346
Table 371 – CP 9/2: DLL protocol selection	347
Table 372 – CP 9/2: AL service selection.....	347
Table 373 – CP 9/2: Device application service selection.....	348
Table 374 – CP 9/2: Layer management service selection	349
Table 375 – CP 9/2: Gateway and network manager service selection	350
Table 376 – CP 9/2: Network management configuration service selection	350
Table 377 – CP 9/2: AL protocol selection	351
Table 378 – Application process status values.....	351
Table 379 – CP 9/2: Device application service-specific APDU selection	352
Table 380 – CP 9/2: Layer management service-specific APDU selection	353
Table 381 – CP 9/2: Gateway and network manager service-specific APDU selection.....	354
Table 382 – CP 9/2: Network management configuration service-specific APDU selection	354
Table C.1 – Network manager requirements	374
Table D.1 – Cached Response Messages	380
Table E.1 – Response code values	381
Table F.1 – Reserved values	390
Table F.2 – Publish mode control codes.....	391
Table F.3 – Write device variable codes	392
Table F.4 – Device variable family codes.....	392
Table F.5 – Device variable classification codes	393
Table F.6 – Analog channel saturated codes	394
Table F.7 – Analog channel fixed codes.....	394
Table F.8 – Standardized status 0 codes	394
Table F.9 – Standardized status 1 codes	394
Table F.10 – Standardized status 2 codes	395
Table F.11 – Standardized status 3 codes	395
Table F.12 – Publish trigger mode codes	395
Table F.13 – Event notification control codes.....	396
Table F.14 – Event status codes.....	396
Table F.15 – TimeTable request codes	396
Table F.16 – TimeTable application domain codes.....	396
Table F.17 – Synchronous action control codes	397
Table F.18 – Real-time clock flags codes.....	397
Table F.19 – Wireless timer codes	397
Table F.20 – Device power source codes.....	398
Table F.21 – Link_type codes	398
Table F.22 – Link option flags codes.....	398
Table F.23 – Superframe mode flags codes	398
Table F.24 – Session_type codes	399
Table F.25 – TimeTable deletion reason codes.....	399

Table F.26 – Disconnect cause codes	399
Table F.27 – Wireless operation codes	399
Table F.28 – Join process status codes	400
Table F.29 – Security type codes	400
Table F.30 – Device list codes	400
Table F.31 – Network access mode codes	401
Table F.32 – Device profile codes	401
Table F.33 – Device power status codes	401
Table F.34 – Neighbor flags codes	402
Table F.35 – Notification mask codes	402
Table F.36 – Join mode codes	402
Table F.37 – Device scheduling flags codes	402
Table F.38 – Network optimization flags codes	403
Table I.1 – Routing requirements	419
Table I.2 – Scheduling requirements	421
Table I.3 – Frame ID 1: 1 s update rate, superframe length 100	424
Table I.4 – Frame ID 4: 4 s update rate, superframe length 400	424
Table I.5 – Frame ID 0: management superframe	425
Table I.6 – Join request (shared with management responses)	425
Table I.7 – Join response (shared with management requests)	425
Table I.8 – Commands	425
Table I.9 – Command responses	425
Table I.10 – Device A	426
Table I.11 – Device B	427
Table I.12 – Device C	427
Table I.13 – Device D	428
Table I.14 – Network manager universal commands	429
Table J.1 – Required command responses	440
Table J.2 – Gateway status flags	442
Table J.3 – Gateway capacity guidelines	443

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – WIRELESS COMMUNICATION NETWORK AND COMMUNICATION PROFILES – WirelessHART™

FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

International Standard IEC 62591 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial process measurement, control and automation.

This standard cancels and replaces IEC/PAS 62591 published in 2009. This first edition constitutes a technical revision.

This bilingual version, published in 2011-07, corresponds to the English version.

The text of this standard is based on the following documents:

FDIS	Report on voting
65C/587/FDIS	65C/595/RVD

Full information on the voting for the approval of this International Standard can be found in the report on voting indicated in the above table.

The French version of this standard has not been voted upon.

This publication has been drafted in accordance with ISO/IEC Directives, Part 2.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

This International standard provides the specification, definitions, and profile for a Wireless communication network. It supplements IEC 61158-5-20, IEC 61158-6-20 where some commands are specified and it supplements IEC 61784-1 where a Communication Profile CP 9/1, universal command, is specified.

This document follows the structure and conventions of IEC 61158 series (for example separation of DL-service definitions and DL-protocol specification) and conventions of IEC 61784-1. IEC 61158 series specify different communication networks. These are structured in different Types. The Type 20 is assigned to technologies of Hart™¹ Communication Foundation (HCF). For other assignments of Type numbers see IEC/TR 61158-1.

NOTE IEC 61158 series do not contain a DLL specification for Type 20. However, this standard includes a DLL specification.

IEC 61784-1, and IEC 61784-2 provide Communication Profile Families (CPF), and, within a family, one to n Communication Profiles. The assigned CPF number for technologies of the HCF is CPF 9. For other assignments of CPF numbers see IEC/TR 61158-1.

A new project number IEC 62591 was assigned to the Type 20 enhancements and the associated CP 9/2, so that this document contains the equivalent subparts of the IEC 61158 series and of IEC 61784-1 organized here in different clauses.

The Type 20 protocol supports two way digital communications for process measurement and control devices. Applications include remote process variable interrogation, cyclical access to process data, parameter setting and diagnostics. This document defines the specification that comprises the Type 20 field communications protocol for wireless devices. Specification of the Type 20 protocol is based largely on the OSI 7-layer Communication Model.

The International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of patents concerning the claims of the patents listed below given in the normative Clauses.

US08/04750 & PCT 12/100986 [HCF] Combined Wired and Wireless Communications with Field Devices
 US08/04777 & PCT 12/100927 [HCF] Increasing Reliability and Reducing Latency in a Wireless Network
 US08/04677 & PCT 12/100995 [HCF] Adaptive Scheduling in a Wireless Network
 US08/04716 & PCT 12/101000 [HCF] Wireless Protocol Adaptor
 US08/04775 & PCT 12/101005 [HCF] Synchronizing Timeslots in a Wireless Communication Protocol
 US08/04751 & PCT 12/101011 [HCF] Suspending Transmissions in a Wireless Communication Network
 US08/04678 & PCT 12/101021 [HCF] Enhancing Security in a Wireless Network
 US08/04746 & PCT 12/101037 [HCF] Wireless Gateway in a Process Control Environment
 US08/04676 & PCT 12/101043 [HCF] Efficient Addressing in Wireless HART Protocol
 US08/04745 & PCT 12/101049 [HCF] Support for Network Management and Device Communications in a Wireless Network
 US08/04749 & PCT 12/101054 [HCF] Priority Based Scheduling and Routing in a Wireless Network
 US08/04740 & PCT 12/101071 [HCF] Routing Packets on a Network Using Directed Graphs
 US08/04761 & PCT 12/101074 [HCF] Scheduling Communication Frames in a Wireless Network

IEC takes no position concerning the evidence, validity and scope of these patent rights.

¹ HART™ and WirelessHART™ are the trade names of HART™ Communication Foundation (HCF). HCF is a non-profit trade organization to support the HART™ Communication. This information is given for the convenience of users of this International Standard and does not constitute an endorsement by IEC of the trade name holder or any of its products. Compliance to this document does not require use of the trade name. Use of the trade name HART™ and WirelessHART™ requires permission of the trade name holder.

The holder of these patent rights has assured IEC that she/he is willing to negotiate licenses either free of charge or under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of these patent rights is registered with IEC. Information may be obtained from:

[HCF] Hart Communication Foundation
9390 Research Boulevard
Suite I-350
Austin TX 78759, USA

Attention is drawn to the possibility that some of the elements of this standard may be the subject of patent rights other than those identified above. IEC shall not be held responsible for identifying any or all such patent rights.

ISO (www.iso.org/patents) and IEC (http://www.iec.ch/tctools/patent_decl.htm) maintain on-line data bases of patents relevant to their standards. Users are encouraged to consult the data bases for the most up to date information concerning patents.

Withdrawing
IECNORM.COM: Click to view the full PDF of IEC 62591:2010

INDUSTRIAL COMMUNICATION NETWORKS – WIRELESS COMMUNICATION NETWORK AND COMMUNICATION PROFILES – WirelessHART™

1 Scope

This International Standard specifies an additional Type 20 communication network to IEC 61158-5-20, IEC 61158-6-20 and a Communication Profile CP 9/2 in addition to IEC 61784-1 CPF 9.

This standard specifies the following:

- Physical layer service definition and protocol specification,
- Data-link layer service and protocol,
- Application layer service and protocol,
- Network management,
- Security,
- Communication profile,
- Wireless procedures and
- Gateway.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 60559, *Binary floating-point arithmetic for microprocessor systems*

IEC/TR 61158-1:2010, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-5-20:2010, *Industrial communication networks – Fieldbus specifications – Part 5-20: Application layer service definition – Type 20 elements*

IEC 61158-6-20:2010, *Industrial communication networks – Fieldbus specifications – Part 6-20: Application layer protocol specification – Type 20 elements*

IEC 61784-1:2010, *Industrial Communication Networks – Profiles – Part 1: Fieldbus profiles*

IEC 61784-2:2010, *Industrial Communication Networks – Profiles – Part 2: Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3*

ISO/IEC 7498-1, *Information technology – Open Systems Interconnection — Basic Reference Model: The Basic Model*

ISO/IEC 8824, *Information Technology – Open Systems Interconnection – Specification of Abstract Syntax Notation One (ASN.1)*

ISO/IEC 8859-1, *Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*

ISO/IEC 9545, *Information technology – Open Systems Interconnection – Application Layer structure*

ISO/IEC 10731, *Information technology – Open Systems Interconnection – Basic Reference Model – Conventions for the definition of OSI services*

IEEE 802-2001, *IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture*

IEEE 802.15.4-2006, *IEEE Standard for Information technology- Telecommunications and information exchange between systems- Local and metropolitan area networks- Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*

3 Terms, definitions, symbols, abbreviations and conventions

For the purposes of this document, the following terms as defined in these publications apply.

3.1 Reference model terms and definitions

This standard is based in part on the concepts developed in ISO/IEC 7498-1 and ISO/IEC 7498-3, and makes use of the following terms defined therein.

DL-entity (N=2)	[7498-1]
DL protocol-data-unit	[7498-1]
DL service-data-unit	[7498-1]
DLS user-data	[7498-1]
NL-entity (N=3)	[7498-1]
NL protocol-data-unit	[7498-1]
NL service-data-unit	[7498-1]
NLS user-data	[7498-1]
TL-entity (N=4)	[7498-1]
TL protocol-data-unit	[7498-1]
TL service-data-unit	[7498-1]
TLS user-data	[7498-1]

3.2 Terms and definitions from ISO/IEC 7498-1 terms

- a) abstract syntax
- b) application entity

- c) application process
- d) application protocol data unit
- e) application service element
- f) application entity invocation
- g) application process invocation
- h) application transaction
- i) presentation context
- j) real open system
- k) transfer syntax

3.2.1 Terms and definitions from ISO/IEC 9545

- a) application-association
- b) application-context
- c) application context name
- d) application-entity-invocation
- e) application-entity-type
- f) application-process-invocation
- g) application-process-type
- h) application-service-element
- i) application control service element

3.2.2 Terms and definitions from ISO/IEC 8824

- a) object identifier
- b) type
- c) value
- d) simple type
- e) structured type
- f) component type
- g) tag
- i) true
- j) false
- k) integer type
- m) octet string type
- n) null type
- o) sequence type
- p) sequence of type
- q) choice type
- r) tagged type
- s) any type
- t) module
- u) production

3.2.3 Terms and definitions from ISO/IEC 8825

- a) encoding (of a data value)
- b) data value
- c) identifier octets (the singular form is used in this standard)
- d) length octet(s) (both singular and plural forms are used in this standard)
- e) contents octets

3.3 Specific terms and definitions

3.3.1

absolute slot number

count of all slots that have occurred since the network was formed

NOTE This number always increments and is never reset to any fixed value or zero. Its current value is always the sequence number of the current slot. Its maximum value is $(2^{40} - 1)$ after that it rolls over to zero.

3.3.2

acknowledge

explicit data-link response to the successful reception of a directed, non-broadcast DLPDU from a DLE source device and the second DLPDU of a two-DLPDU transaction

3.3.3**adapter**

network device that enables connection and communication with a field device which does not have a direct connection to a Type 20 network

3.3.4**advertise DLPDU**

invite new devices into the network

NOTE When a device wishes to join a network, it listens for these DLPDUs and then uses the information in the DLPDU to synchronize with the network and initiate the join process.

3.3.5**analog channel**

continuously varying electrical signal connecting a field device to the remainder of the data acquisition or control system

NOTE 1 Some field devices support multiple analog channels (input or output).

NOTE 2 Each analog channel transmits a single dynamic variable to or from the field device.

3.3.6**application**

function or data structure for which data is consumed or produced

[IEC 61158-6-9]

3.3.7**application domain**

classification of the application process based on communication priority and capacity requirement

3.3.8**application object**

object class that manages and provides the run time exchange of messages across the network and within the network device

NOTE Multiple types of application object classes may be defined.

[IEC 61158-6-9]

3.3.9**application relationship**

cooperative association between two or more application-entity-invocations for the purpose of exchange of information and coordination of their joint operation

NOTE This relationship is activated either by the exchange of application-protocol-data-units or as a result of pre-configuration activities.

[IEC 61158-6-9]

3.3.10**application relationship endpoint**

context and behavior of an application relationship as seen and maintained by one of the application processes involved in the application relationship

NOTE Each application process involved in the application relationship maintains its own application relationship endpoint.

3.3.11

ASN time

time represented as absolute slot number; any instance in time is equal to absolute slot number at that instance

3.3.12

attribute

description of an externally visible characteristic or feature of an object

NOTE The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may also affect the behavior of an object. Attributes are divided into class attributes and instance attributes.

3.3.13

behavior

indication of how the object responds to particular events

NOTE Its description includes the relationship between attribute values and services.

3.3.14

broadcast

process of sending a PDU to all devices that are connected to the network and are able to receive the transmission

3.3.15

broadcast address

address used by a master to send a command to all devices

3.3.16

busy

state of a device which is active on different tasks and cannot execute a command at the time

3.3.17

channel

RF frequency band used to transmit a modulated signal carrying packets

3.3.18

channel blacklisting

method of eliminating an RF channel from usage

3.3.19

channel hopping

regular change of transmit or receive frequency to combat interference and fading

3.3.20

channel offset

link-specific value provided by the network manager that is used to calculate the channel to use when channel hopping

3.3.21

class

set of objects, all of which represent the same kind of system component

NOTE A class is a generalization of the object; a template for defining variables and methods. All objects in a class are identical in form and behavior, but usually contain different data in their attributes.

3.3.22

class attributes

attribute that is shared by all objects within the same class

3.3.23**class code**

unique identifier assigned to each object class

3.3.24**class specific service**

service defined by a particular object class to perform a required function which is not performed by a common service

NOTE A class specific object is unique to the object class which defines it.

3.3.25**clear channel assessment**

testing while the RF channel is in use to avoid initiating a transaction

NOTE It is performed by listening to the channel prior to sending the first DLPDU of a transaction. If signal is detected, then the transaction is deferred to a later TDMA slot.

3.3.26**client**

a) object which uses the services of another (server) object to perform a task

b) initiator of a message to which a server reacts, such as the role of an AR endpoint in which it issues confirmed service request APDUs to a single AR endpoint acting as a server

[IEC 61158-6-9]

3.3.27**conveyance path**

unidirectional flow of APDUs across an application relationship

3.3.28**cyclic**

term used to describe events which repeat in a regular and repetitive manner

3.3.29**device**

any entity containing an implementation of Type 20 fieldbus

3.3.30**device ID**

serial number for a device that is unique among all instances of one type of device

3.3.31**device profile**

collection of device dependent information and functionality providing consistency between similar devices of the same device type

3.3.32**expanded device type**

manufacturer's type of a device as specified in IEC 61158-6-20:2010, Table 7

3.3.33**device variable**

uniquely defined data item within a Field Device that is always associated with cyclical process information

NOTE A device variable's value varies in response to changes and variations in the process to which the device is connected.

3.3.34

discovery

process of finding a new neighbor by listening to the special DLPDU that indicates the presence of the device

3.3.35

discovery link

link that is used for transmitting and receiving the special DLPDU that indicates the presence of the device

3.3.36

dynamic variable

connection between a process and an analog channel

NOTE A device may contain primary, secondary, tertiary, and quaternary variables. These are collectively called the dynamic variables.

3.3.37

endpoint

one of the communicating entities involved in a connection

3.3.38

error

discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition

3.3.39

error code

identification of a specific type of error within an error class

3.3.40

field device

network device that is connected to the process or to plant equipment

NOTE It directly connects to the sensor or actuator or performs process control function and it is directly connected to the physical layer of Type 20.

3.3.41

frequency channel

allocation of the frequency spectrum in a given frequency range

3.3.42

gateway

network device containing at least one host interface such as serial or Ethernet, acting as ingress or an egress point enabling communication between host applications and field devices

3.3.43

graph

collection of directed paths among network devices such that more than one route is possible from a source device to destination device

3.3.44

Graph_ID

identifier used to indicate a specific graph entry

3.3.45

handheld

a) host application residing on a portable device

b) qualifier to other terms such as key, superframe

3.3.46

hop

movement of a packet directly between two adjacent neighbors in one network transaction without the participation of any other node in the network

3.3.47

hop to live

field in the network header of each packet that specifies how many more hops a packet can travel before being discarded

3.3.48

join

process by which a network device is authenticated and allowed to participate in the network

NOTE A device is considered Joined when it has the network key, a network manager session and a normal (not join) superframe and links.

3.3.49

join_key

security key that is used to start the join process

3.3.50

latency

time it takes for a packet to cross a network connection, from sender to receiver

3.3.51

link

full communication specification between adjacent devices in a network and it includes the communication parameters necessary to move a DLPDU one hop

NOTE A Link is a function of source and destination address pairing, slot and channel offset assignment, direction of communication, dedicated or shared communication, and type. Links are assigned to superframes as part of the scheduling process.

3.3.52

logical link control

higher of the two data-link layer sublayers and used for error detection, flow control, framing, and addressing

3.3.53

long_tag

32-character restricted ISO Latin-1 string used to identify a field device

3.3.54

loop current

value measured by a mA in series with the field device

NOTE The loop current is a near DC analog 4 mA to 20 mA signal used to communicate a single value between the control system and the field device. Voltage mode devices use "Volts DC" as their engineering units where "loop current" values are used.

3.3.55

maintenance port

interface in the network device that is used for provisioning

NOTE This interface is used to write the Join_key and network identifier, and to monitor the join process.

3.3.56

management information

network-accessible information that supports managing the operation of the fieldbus system, including the application layer

NOTE Managing includes functions such as controlling, monitoring, and diagnosing.

3.3.57

manufacturer ID

2 octet enumeration identifying the manufacturer that produced a device

NOTE A manufacturer is required to use the value assigned to it and is not permitted to use the value assigned to another manufacturer.

3.3.58

master

device that initiates communication activity by sending request APDU to a device and expecting a response PDU

3.3.59

medium access control

lower of the two data-link sublayers and used for controlling the access to the communication channel

3.3.60

multicast

sending of one packet to more than one device in the network using only one transmission

3.3.61

neighbor

adjacent node in the network such that the receive signal level (RSL) from it suggests that the communication is possible in at least one direction

3.3.62

neighbor table

list of all devices that the device considers as neighbor

NOTE It also stores the attributes of the neighbor.

3.3.63

network

series of nodes connected by some type of communication medium

NOTE The connection paths between any pair of nodes can include repeaters, routers and gateways.

3.3.64

network device

device with a direct physical layer connection to the network

NOTE Each network device has a unique address that is used in communication with the device. Network devices include field device, access point (i.e. gateway), adapter and handheld.

3.3.65

network_ID

identifier used to indicate a network to which all inter-communicating devices are connected

3.3.66

network manager

entity that is responsible for configuration of the network, scheduling communication between network devices, management of the routing tables and monitoring and reporting the health of the network

3.3.67**nickname**

identifier assigned to a device such that it is unique within the network to which the device is connected

3.3.68**node**

addressable logical or physical device attached to the network

3.3.69**nonce**

non-repeating number constructed so as to be unique to the current packet and to ensure that old communications cannot be reused in replay attacks

NOTE Nonce is necessary for maintaining packet secrecy and providing sender authenticity and packet integrity.

3.3.70**packet**

formatted, aggregated bits that are transmitted together in time across the physical medium

[IEEE 802.15.4, modified]

3.3.71**path**

direct communication link between two neighbor devices in the mesh

3.3.72**payload data**

contents of a data message that is being transmitted

3.3.73**pipe**

secure, bi-directional, virtual communication connection between peer devices

3.3.74**polling address**

integer used to identify a device

NOTE The polling address is used to construct a one-octet address.

3.3.75**pre-defined AR endpoint**

AR endpoint that is defined locally within a device without use of the create service

NOTE Pre-defined ARs that are not pre-established are established before being used.

3.3.76**pre-established AR endpoint**

AR endpoint that is placed in an established state during configuration of the AEs that control its endpoints

3.3.77**route**

one sequence of unidirectional paths from one device to another connecting a source device to a destination device

3.3.78**route ID**

identifier used to indicate a specific route

3.3.79

server

<communication> role of an AREP in which it returns a confirmed service response APDU to the client that initiated the request

3.3.80

server

<system> object which provides services to another (client) object

3.3.81

service

operation or function than an object and/or object class performs upon request from another object and/or object class

NOTE A set of common services is defined and provisions for the definition of object-specific services are provided. Object-specific services are those which are defined by a particular object class to perform a required function which is not performed by a common service.

3.3.82

session ID

identifier used to indicate a specific session entry

3.3.83

slave

device that initiates communication activity only after it receives a request PDU from a master device and it is required to send a response to that request

3.3.84

slot

fixed time interval that may be used for communication between neighbors

3.3.85

superframe

collection of slots repeating at a constant rate; each slot has a link associated with it

3.3.86

superframe_ID

identifier used to indicate a specific superframe entry

3.3.87

tag

8-character ASCII string used to identify a field device

3.3.88

time division multiple access

medium access control technique that uses time slots where communications between devices can occur

NOTE It provides collision free, deterministic communications.

3.3.89

transaction

exchange of related, consecutive frames between two peer medium access control entities, required for a successful transmission

NOTE A transaction consists of either (a) a single PhPDU transmission from a source device, or (b) one PhPDU from the source device followed by a second, link-level acknowledgement PhPDU from the destination device.

3.3.90**unicast**

sending of a packet to a single device in the network

3.3.91**unique ID**

identifier assigned to a device which is unique among all instances of the devices compliant to this standard and compliant to Communication Profile CP 9/2 using identical organization name as specified in 9.3.1

3.4 Abbreviations

ACK	Acknowledge
AE	Application entity
AL	Application layer
AP	Application process
APDU	Application protocol data unit
APO	Application Object
AR	Application relationship
AREP	Application relationship endpoint
ARPM	Application Relationship Protocol Machine
ASCII	American Standard Code for Information Interchange
ASE	Application Service Element
ASN	Absolute Slot Number
BACK	Burst acknowledge
CCA	Clear channel assessment
Cnf	Confirmation
CP	Communication Profile
CPF	Communication Profile Family
CRC	Cyclic redundancy check
dBm	Abbreviation for the power ratio in decibels (dB) of the measured power, referenced to one mW (1 mW), 0 dBm = 1 mW, 10 dBm = 10 mW, 20 dBm = 100 mW, 30 dBm = 1 W
DL-	Data-link layer (as a prefix)
DLE	DL-entity (the local active instance of the data-link layer)
DLL	Data-link layer
DLM	DL-management
DLMS	DL-management service
DLPDU	Data-link protocol data unit
DLS	DL-service
DLSDU	DL-service-data-unit
DR	Delayed response
DRM	Delayed response mechanism
EUI-64	Extended unique identifier (64 bits long)
FAL	Fieldbus application layer
FSMP	FAL Service Protocol Machine
HCF	HART™ Communication Foundation

HTL	Hop to live	
ID	Identifier	
IEC	International Electrotechnical Commission	
Ind	Indication	
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector	
LLC	Logical link control	
LSB	Least significant byte	
MAC	Medium access control	
MIC	Message integrity code	
MSB	Most significant byte	
NL	Network layer	
NL-	Network layer (as a prefix)	
NLE	NL-entity (the local active instance of the network layer)	
NLM	NL-management	
NLMS	NL-management service	
NLS	NL-service	
NLSDU	NL-service-data-unit	
NPDU	NL-protocol-data-unit	
NSDU	NL-service-data-unit	
O-QPSK	Offset Quadrature Phase-Shift Keying	
OSI	Open Systems Interconnect	
OUI	Organizationally Unique Identifier	
PDU	Protocol data unit	
PhE	PhL-entity (the local active instance of the physical layer)	
PhL-	Physical layer (as a prefix)	
PhPDU	PhL-protocol-data-unit	[ISO/IEC 7498-1]
PhS	Physical layer service	[ISO/IEC 7498-1]
QoS	Quality-of-service	
Req	Request	
RF	Radio Frequency	
RSL	Receive signal level	
Rsp	Response	
RSV	Reserved	
SAP	Service access point	
SN	Sign bit	
SOM	Start of message	
STX	Start of a transaction	
TDMA	Time Division Multiple Access	
TL	Transport layer	
TL-	Transport layer (as a prefix)	
TLE	TL-entity (the local active instance of the transport layer)	
TLM	TL-management	

TLMS	TL-management service
TLS	TL-service
TMPM	Transport layer Mapping Protocol Machine
TPDU	TL-protocol-data-unit
TSDU	TL-service-data-unit
VFD	Virtual field device

3.5 Conventions

3.5.1 Conventions used for service

This standard uses the descriptive conventions given in ISO/IEC 10731.

The service model, service primitives, and time-sequence diagrams used are entirely abstract descriptions; they do not represent a specification for implementation.

Service primitives, used to represent service user/service provider interactions (see ISO/IEC 10731), convey parameters that indicate information available in the user/provider interaction.

This standard uses a tabular format to describe the component parameters of the Physical, Data Link, Network, Transport and Application layer service primitives. The parameters that apply to each group of primitives are set out in tables throughout the remainder of this standard. Each table consists of up to six columns, containing the name of the service parameter, and a column each for those primitives and parameter-transfer directions used by the service:

- the request primitive's input parameters;
- the request primitive's output parameters;
- the indication primitive's output parameters;
- the response primitive's input parameters; and
- the confirm primitive's output parameters.

One parameter (or part of it) is listed in each row of each table. For local services, only request primitive with input and output parameters is used. Under the appropriate service primitive columns, a code is used to specify the type of usage of the parameter on the primitive and parameter direction specified in the column:

M parameter is mandatory for the primitive;

U parameter is a user option and may or may not be provided depending on the dynamic usage of the DLS-user. When not provided, a default value for the parameter is assumed;

S only one parameter in a group of parameters using this code is required.

C parameter is conditional upon other parameters or upon the environment of the DLS-user;

CU parameter is a conditional user option, and may or may not be permitted depending upon other parameters or upon the environment of the DLS-user. When permitted, it may or may not be provided depending on the dynamic usage of the DLS-user. When permitted and not provided, a default value for the parameter is assumed.

(blank) parameter is never present.

Some entries are further qualified by items in brackets. These may be:

a) a parameter-specific constraint

(=) indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table;

- ($\leq$) indicates that the set of parameter values has an implicit order and that the parameter's value is less than or equal to that of the parameter in the service primitive to its immediate left in the table (that is, left by one or two columns);
- ($\geq$) indicates that the set of parameter values has an implicit order and that the parameter's value is greater than, or equal to, that of the parameter in the service primitive to its immediate left in the table (that is, left by one or two columns).
- b) an indication that some note applies to the entry
 - (n) indicates that the following note n contains additional information pertaining to the parameter and its use.

3.5.2 Conventions used for DL-protocol (state machine)

This standard uses a diagram to represent a state machine. The following convention is used.

- The labeled rectangles represent states that a component can be in.
- State transitions are directed lines. They show which state a component leaves and which state it transitions to.
- A transition is labeled with the events that caused it and the corresponding actions (this may be empty). The events (above the line) causing the transition is separated from the resulting actions (below the line) by a horizontal line.
- A tilde (" ~ ") means the one's complement or negation.

3.5.3 Conventions for Application Layer service

3.5.3.1 Overview

The Fieldbus Application Layer (FAL) is defined as a set of object-oriented Application Service Elements (ASE). Each ASE is specified in a separate subclause. Each ASE specification is composed of two parts, its class specification, and its service specification.

The class specification defines the attributes of the class. The service specification defines the services that are provided by the ASE.

3.5.3.2 Conventions for class definitions

Class definitions are described using templates. Each template consists of a list of attributes for the class. An example of the template is shown below:

FAL ASE:		ASE Name
CLASS:	Class Name	
CLASS ID:	#	
PARENT CLASS:	Parent Class Name	
ATTRIBUTES:		
1	(o) Key Attribute:	numeric identifier
2	(o) Key Attribute:	name
3	(m) Attribute:	attribute name(values)
4	(m) Attribute:	attribute name(values)
4.1	(s) Attribute:	attribute name(values)
4.2	(s) Attribute:	attribute name(values)
4.3	(s) Attribute:	attribute name(values)
5.	(c) Constraint:	constraint expression
5.1	(m) Attribute:	attribute name(values)
5.2	(o) Attribute:	attribute name(values)
6	(m) Attribute:	attribute name(values)
6.1	(s) Attribute:	attribute name(values)
6.2	(s) Attribute:	attribute name(values)

SERVICES:

- 1 (o) OpsService: service name
- 2. (c) Constraint: constraint expression
- 2.1 (o) OpsService: service name
- 3 (m) MgtService: service name

- (1) The "FAL ASE:" entry is the name of the FAL ASE that provides the services for the class being specified.
- (2) The "CLASS:" entry is the name of the class being specified. All objects defined using this template will be an instance of this class. The class may be specified by this standard, or by a user of this standard.
- (3) The "CLASS ID:" entry is a number that identifies the class being specified. This number is unique within the FAL ASE that will provide the services for this class. When qualified by the identity of its FAL ASE, it unambiguously identifies the class within the scope of the FAL. The value "NULL" indicates that the class cannot be instantiated. Class IDs between 1 and 255 are reserved by this standard to identify standardized classes. They have been assigned to maintain compatibility with existing national standards. CLASS IDs between 256 and 2048 are allocated for identifying user defined classes.
- (4) The "PARENT CLASS:" entry is the name of the parent class for the class being specified. All attributes defined for the parent class and inherited by it are inherited for the class being defined, and therefore do not have to be redefined in the template for this class.

NOTE The parent-class "TOP" indicates that the class being defined is an initial class definition. The parent class TOP is used as a starting point from which all other classes are defined. The use of TOP is reserved for classes defined by this standard.

- (5) The "ATTRIBUTES" label indicate that the following entries are attributes defined for the class.
 - a) Each of the attribute entries contains a line number in column 1, a mandatory (m) / optional (o) / conditional (c) / selector (s) indicator in column 2, an attribute type label in column 3, a name or a conditional expression in column 4, and optionally a list of enumerated values in column 5. In the column following the list of values, the default value for the attribute may be specified.
 - b) Objects are normally identified by a numeric identifier or by an object name, or by both. In the class templates, these key attributes are defined under the key attribute.
 - c) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting is used to specify
 - i) fields of a structured attribute (4.1, 4.2, 4.3),
 - ii) attributes conditional on a constraint statement (5). Attributes may be mandatory (5.1) or optional (5.2) if the constraint is true. Not all optional attributes require constraint statements as does the attribute defined in (5.2).
 - iii) the selection fields of a choice type attribute (6.1 and 6.2).
- (6) The "SERVICES" label indicates that the following entries are services defined for the class.
 - a) An (m) in column 2 indicates that the service is mandatory for the class, while an (o) indicates that it is optional. A (c) in this column indicates that the service is conditional. When all services defined for a class are defined as optional, at least one has to be selected when an instance of the class is defined.

- b) The label "OpsService" designates an operational service (1).
- c) The label "MgtService" designates a management service (2).
- d) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting within the list of services is used to specify services conditional on a constraint statement.

3.5.3.3 Conventions for service definitions

3.5.3.3.1 General

This standard uses the descriptive conventions given in ISO/IEC 10731.

The service model, service primitives, and time-sequence diagrams used are entirely abstract descriptions; they do not represent a specification for implementation.

3.5.3.3.2 Service parameters

Service primitives are used to represent service user/service provider interactions (ISO/IEC 10731). They convey parameters which indicate information available in the user/provider interaction.

NOTE 1 See the note under 3.5.3.3.3 relative to the non-inclusion of service parameters that are appropriate to a protocol specification or programming interface specification or implementation specification, but not to an abstract service definition.

This standard uses a tabular format to describe the component parameters of the service primitives. The parameters that apply to each group of service primitives are set out in tables throughout the remainder of this standard. Each table consists of up to six columns: a column for the name of the service parameter, and a column each for those primitives and parameter-transfer directions used by the service. The possible six columns are

- 1) the parameter name;
- 2) the request primitive's input parameters;
- 3) the request primitive's output parameters;

NOTE 2 This is a seldom-used capability. Unless otherwise specified, request primitive parameters are input parameters.

- 4) the indication primitive's output parameters;
- 5) the response primitive's input parameters; and
- 6) the confirm primitive's output parameters.

NOTE 3 The request, indication, response and confirm primitives are also known as requestor.submit, acceptor.deliver, acceptor.submit, and requestor.deliver primitives, respectively (see ISO/IEC 10731).

One parameter (or component of it) is listed in each row of each table. Under the appropriate service primitive columns, a code is used to specify the type of usage of the parameter on the primitive specified in the column:

- M parameter is mandatory for the primitive
- U parameter is a User option, and may or may not be provided depending on dynamic usage of the service user. When not provided, a default value for the parameter is assumed.
- C parameter is conditional upon other parameters or upon the environment of the service user.
- (blank) parameter is never present.
- S parameter is a selected item.

Some entries are further qualified by items in brackets. These may be

- a) a parameter-specific constraint:
“(=)” indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.
- b) an indication that some note applies to the entry:
“(n)” indicates that the following note “n” contains additional information pertaining to the parameter and its use.

3.5.3.3.3 Service procedures

The procedures are defined in terms of

- the interactions between application entities through the exchange of fieldbus Application Protocol Data Units, and
- the interactions between an application layer service provider and an application layer service user in the same system through the invocation of application layer service primitives.

These procedures are applicable to instances of communication between systems which support time-constrained communications services within the fieldbus application layer.

NOTE The IEC 61158-5 subseries of standards define sets of abstract services. They are neither protocol specifications nor implementation specifications nor concrete programming interface specifications. Therefore there are restrictions on the extent to which service procedures can be mandated in the parts of IEC 61158-5 subseries. Protocol aspects that can vary among different protocol specifications or different implementations that instantiate the same abstract services are unsuitable for inclusion in these service definitions, except at the level of abstraction that is necessarily common to all such expressions. In this document, Clause 7 is the equivalent of the IEC 61158-5 subseries.

For example the means by which service providers pair request and reply Protocol data units (PDU) is appropriate for specification in an IEC 61158-6 subseries protocol specification standard but not in an IEC 61158-5 subseries abstract service definition standard. Similarly, local implementation methods by which a service provider or service user pairs request and confirm(-ation) primitives, or indication and response primitives, is appropriate for an implementation specification or for a programming interface specification, but not for an abstract service standard or for a protocol standard, except at a level of abstraction that is necessarily common to all embodiments of the specifying standard. In all cases, the abstract definition is not permitted to over-specify the more concrete instantiating realization.

Further information on the conceptual service procedures of an implementation of a protocol that realizes the services of one of the IEC 61158-5 subseries abstract service definitions can be found in IEC/TR 61158-1:2007, 9.6.

3.5.4 Conventions for the AL-protocol

3.5.4.1 General concept

The FAL is defined as a set of object-oriented ASEs. Each ASE is specified in a separate subclause. Each ASE specification is composed of three parts: its class definitions, its services, and its protocol specification. The first two are contained in IEC 61158-5. The protocol specification for each of the ASEs is defined in this standard.

The class definitions define the attributes of the classes supported by each ASE. The attributes are accessible from instances of the class using the Management ASE services specified in IEC 61158-5. The service specification defines the services that are provided by the ASE.

This standard uses the descriptive conventions given in ISO/IEC 10731.

3.5.4.2 Conventions for class definitions

The data-link layer mapping definitions are described using templates. Each template consists of a list of attributes for the class. The general form of the template is defined in IEC 61158-5-20:2010, 3.8.2.

3.5.4.3 Conventions used in state machines

The state machines are described in Table 1.

Table 1 – Conventions used for state machines

#	Current state	Event / condition => action	Next state
Name of this transition	The current state to which this state transition applies	Events or conditions that trigger this state transaction. => The actions that are taken when the above events or conditions are met. The actions are always indented below events or conditions	The next state after the actions in this transition is taken

The conventions used in the state machines are as follows:

:= Value of an item on the left is replaced by value of an item on the right. If an item on the right is a parameter, it comes from the primitive shown as an input event.

xxx A parameter name.

Example:
Identifier := reason
means value of a 'reason' parameter is assigned to a parameter called 'Identifier.'

"xxx" Indicates fixed value.

Example:
Identifier := "abc"
means value "abc" is assigned to a parameter named 'Identifier.'

= A logical condition to indicate an item on the left is equal to an item on the right.

< A logical condition to indicate an item on the left is less than the item on the right.

> A logical condition to indicate an item on the left is greater than the item on the right.

<> A logical condition to indicate an item on the left is not equal to an item on the right.

&& Logical "AND"

|| Logical "OR"

This construct allows the execution of a sequence of actions in a loop within one transition. The loop is executed for all values from start_value to end_value.

Example:
for (Identifier := start_value to end_value)
actions
endfor

This construct allows the execution of alternative actions depending on some condition (which might be the value of some identifier or the outcome of a previous action) within one transition.

Example:
If (condition)
actions
else
actions
endif

Readers are strongly recommended to refer to the subclauses for the Application relationship endpoint (AREP) attribute definitions, the local functions, and the FAL-PDU definitions to

understand protocol machines. It is assumed that readers have sufficient knowledge of these definitions, and they are used without further explanations.

3.5.4.4 Class diagram

3.5.4.4.1 General

UML is used in this document to show class diagram. The purpose of a class diagram is to show the objects and their relationships within a system. In a class diagram objects are classes (as opposed to an instance diagram where the objects are instances of classes). There are several forms of relationships including associations, generalization/specialization, aggregation, and dependency.

3.5.4.4.2 Classes

Classes contain three compartments arranged vertically. The top compartment is the class name. The middle compartment lists the class's attributes. The bottom compartment lists the class's operations. When drawing a class element on a class diagram, the top compartment is mandatory, and the bottom two compartments are optional. The bottom compartment is not required in a diagram in which the purpose is to show only the relationship between classes. It is not necessary to show all attributes in all drawings where the class is used. Figure 1 shows a Superframe class with operations suppressed. In this figure, the class name is 'Superframe'. The class is illustrated with three attributes, Superframe_ID, NumSlots, and Active.

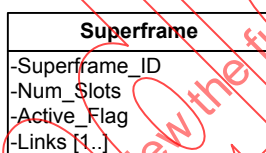


Figure 1 – Superframe class example

3.5.4.4.3 Relationships

The forms of relations used in this document are summarized in Table 2.

Table 2 – Class relationship notation

Relationship	Function	Notation
Bi-directional association	A bi-directional association is indicated by a solid line between the two classes.	
Uni-directional association	In a uni-directional association, two classes are related, but only one class knows that the relationship exists. It is indicated by a line with one arrow between the two classes.	
Association showing multiplicity	Multiplicity specifies how many instances of one class can be related to one instance of another class. In this example the 0..1 means that when an instance of a 'A' exists, it can either have zero or one instance of a 'B' associated with it (B may not have yet been created).	
Generalization	A relationship more specific and more general description, used for inheritance.	
Basic aggregation	Part-whole relationship – the lifecycle of a part class is independent from the whole class's lifecycle.	
Composite aggregation	Part-whole relationship – the lifecycle of a part class is dependent on the containing object.	
Dependency	Shows a relationship between two model elements; it indicates that one object depends on another object for some behavior.	

3.5.4.4.4 Inheritance

An important concept in object-oriented design is inheritance. Inheritance refers to the ability of one class (child class) to inherit the identical functionality of another class (super class), and then add new functionality of its own. To model inheritance on a class diagram, a solid line is drawn from the child class (the class inheriting the behavior) with a closed, arrowhead (or triangle) pointing to the super class. Figure 2 shows a Superframe class modeled as a specialized type of Graph.



Figure 2 – Inheritance example

3.5.4.4.5 Basic aggregation

Another important concept in object-oriented design is aggregation. Aggregation is used to model a "whole to its parts" relationship. There are two forms of aggregation; basic aggregation and composition aggregation. In basic aggregation relationships, the lifecycle of a part class is independent from the whole class's lifecycle. As shown in Figure 3, Car is a whole entity and Wheel is part of the overall Car. The Wheel can be created ahead of time, and it can sit in a warehouse before being placed on a car during assembly.

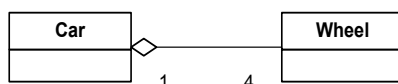


Figure 3 – Basic aggregation example

3.5.4.4.6 Composition aggregation

The other form of aggregation is composition aggregation, usually simply referred to as composition. Figure 4 shows the relationship of a superframe and its links. In this case a link cannot exist without the superframe. Composition is drawn with a filled diamond shape.

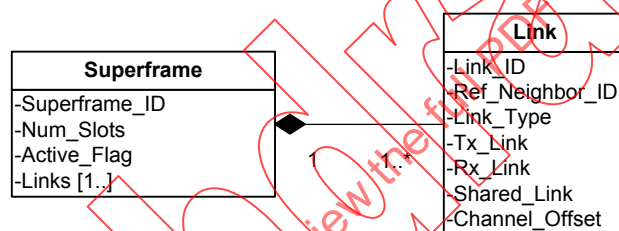


Figure 4 – Composition aggregation example

3.5.4.4.7 Multiplicity

Multiplicity shows how many instances of one class can be related to one instance of another class. Table 3 lists the examples of multiplicity indicators along with their meanings.

Table 3 – Multiplicity notation

Potential Multiplicity Values	
Indicator	Meaning
0..1	Zero or one
1	Only one
0..*	Zero or more
*	Zero or more
1..*	One or more
1..2	One or Two
4	Only four
0..5	Zero to Five

4 Physical layer specification and service definition

4.1 Physical layer service definition

4.1.1 General overview

Type 20 uses Radio Frequencies (RF) to communicate between devices. The distance between two devices can be 100 meters or more depending upon the transmitter power, antenna type and lack of obstruction between two devices. Devices complying with Type 20 can be powered either by mains power, battery, energy harvesting devices (for example solar, vibration, etc), or a combination there of.

This standard defines the Type 20 physical layer (PhL). It is used for devices implementing wireless mesh network communication as defined by this standard. The PhL is responsible for the signaling method, signal strength, device sensitivity, and environment for sending bits across the network media. This wireless PhL specification is based on the 2.450 MHz DSSS physical layer and the employing offset quadrature phase-shift keying (O-QPSK) modulation according to IEEE STD 802.15.4-2006. This standard defines the services that the physical layer provides to higher layers of this standard to send and receive data on the media. It also defines the local management parameters used to control the operation of the PhL.

The PhL of this standard deals with radio characteristics making a network connection. It associates with any part of the network structure that doesn't process information in any way. It is responsible for sending the bits across the network media. It does not define what a bit is or how it is used, merely how it's sent. The PhL is responsible for transmitting and receiving the data.

4.1.2 Physical layer services

4.1.2.1 Facilities of the physical layer services

The PhL provides following services:

- Control service – it supports enabling or disabling of the PhL entity (PhE) and to do clear channel assessment of the underlying medium,
- Message service – it supports the transfer of Physical layer service (PhS) -user-data from a source PhS-user to a destination PhS-user, and
- PhL-management services – it supports configuration of the PhL.

4.1.2.2 Sequence of primitives

The relations of primitives of each type at the interface of PhS-user and one PhE to primitives at the PhS-user and the other PhEs for error-free cases are summarized in the diagrams of Figure 5.

Before using the PhE, the PhS-user has to enable it as shown in the sequence a). The sequence c) shows the transmission of PhPDU from the source PhL-entity (PhE) to the destination PhE. The transmission starts with clear channel assessment as shown in the sequence b). At the destination, the reception starts with PH-RECV-SD indication after the start delimiter has been received by the PhE. The error free reception ends with PH-DATA indication. If there is any error in the reception then the PhE at the destination indicates it by a status parameter and does not deliver any Physical layer service (PhS) user-data.

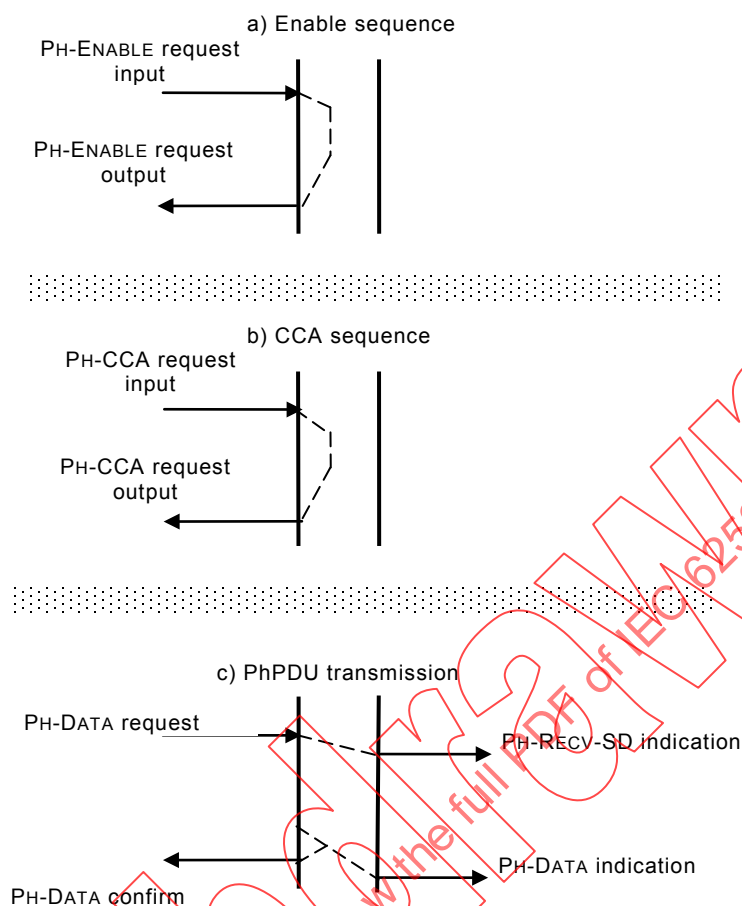


Figure 5 – Physical layer service sequences

4.1.2.3 PH-ENABLE service

4.1.2.3.1 Types of primitives and parameters

Table 4 indicates the types of primitives and the parameters needed for the PhL-Enable service. This service is used to enable or to disable the PhE and to place it in transmit or receive mode on the specified frequency channel. The request service is successful only if the channel is in idle state.

Table 4 – PH-ENABLE primitives and parameters

Parameter name	Request	
	Input	Output
Channel	M	M
State	M	M

4.1.2.3.2 Parameters

4.1.2.3.2.1 Channel

It specifies one of the available channels for which this service is used.

4.1.2.3.2.2 State

In the request primitive, it specifies the state in which the channel should be, when this service is completed. In the confirm primitive, it specifies the current state of the channel after the service is completed. Its value can be either Enable or Disable.

4.1.2.4 PH-CCA service

4.1.2.4.1 Types of primitives and parameters

Table 5 indicates the types of primitives and the parameters needed for the PH-CCA service. This service is used to check the state of the channel. The request service is successful only if the channel is enabled in receive mode.

Table 5 – PH-CCA primitives and parameters

Parameter name	Request	
	Input	Output
State		M
NOTE The request primitive does not require any parameter.		

4.1.2.4.2 Parameter - state

This parameter shows the result of clear channel assessment. The values of this parameter are:

- Channel Idle,
- Channel busy,
- Transceiver off and unable to perform assessment.

4.1.2.5 PH-RCV-SD service

The only primitive of this service is indication and it has no parameter. This primitive is used by the PhE at the end of reception of the start delimiter in receive mode.

4.1.2.6 PH-DATA service

4.1.2.6.1 Types of primitives and parameters

Table 6 indicates the types of primitives and the parameters needed for the PH-DATA service.

Table 6 – PH-DATA primitives and parameters

Parameter name	Request	Indication	Confirm
PhS-user-data	M	C(=)	
Receive signal level		C	
Status		M	M

4.1.2.6.2 Parameters

4.1.2.6.2.1 PhS-user-data

This parameter allows the transmission of data between PhS-users without alteration by the PhS-provider. If there is any error in reception then this parameter is absent in the indication primitive. Else, its value is identical to the value in the request primitive.

4.1.2.6.2.2 Receive signal level

This parameter indicates the estimated receive signal level for this PhPDU. It is present only if there is no reception error.

4.1.2.6.2.3 Status

In the indication primitive, this parameter indicates either success or one of the following reasons for error:

- Receive buffer overflow,
- Incomplete packet.

In the Confirm primitive, this parameter indicates either success or one of the following reasons for error:

- Transceiver off,
 - Receiver on,
- NOTE If the Receiver is on, then the PhE cannot transmit.
- Transmitter busy.

4.1.2.7 PHM-SET service

4.1.2.7.1 Types of primitives and parameters

Table 7 indicates the primitive and parameters of this service.

Table 7 – PHM-SET primitive and parameters

Parameter name	Request	
	Input	Output
PHM-object-identifier	M	
Desired-value	M	
Status		M

4.1.2.7.2 Parameters

4.1.2.7.2.1 PHM-object-identifier

This parameter specifies the object within the PhE whose value is to be altered. The naming-domain of the PHM-object-identifier is the local-view.

4.1.2.7.2.2 Desired-value

This parameter specifies the desired value for the object specified by the associated PHM-object-identifier. Its type is identical to that of the specified object.

4.1.2.7.2.3 Status

This parameter allows the PhS-user to determine whether the requested service was provided successfully, or failed.

NOTE The failure reason may also be provided.

4.1.2.8 PHM-GET service

4.1.2.8.1 Types of primitives and parameters

Table 8 indicates the primitive and parameters of this service.

Table 8 – PHM-GET primitive and parameters

Parameter name	Request	
	Input	Output
PHM-object-identifier	M	
Status		M
Current-value		C

4.1.2.8.2 Parameters

4.1.2.8.2.1 PHM-object-identifier

This parameter specifies the object within the PhE whose value is being requested. The naming-domain of the PHM-object-identifier is the local-view.

4.1.2.8.2.2 Status

This parameter allows the PhS-user to determine whether the requested service was provided successfully, or failed.

NOTE The failure reason may also be provided.

4.1.2.8.2.3 Current-value

This parameter is present when the status parameter indicates that the requested service was performed successfully. This parameter specifies the current value for the object specified by the associated PHM-object-identifier. Its type is identical to that of the specified object.

4.2 Physical layer protocol specification

The physical layer protocol is specified in IEEE 802.15.4-2006. The profile for this standard is specified in 9.3.3.

5 Data-link layer service definition – Type 20 elements

5.1 Data-link layer services

5.1.1 Facilities of the data-link layer services

5.1.1.1 Data transfer service

The Message services support the transfer of DL-service-data-unit (DLSDU) from a source DLS-user to a destination DLS-user. The data-link layer supports automatic retransmission to ensure error-free data transfer.

5.1.1.2 Data receive service

This service supports reception only of a DL-service-data-unit (DLSDU) without sending any acknowledgement.

5.1.1.3 Event service

These services indicate events on the link:

- reception of a DLPDU from a DLE indicating that it is disconnecting from the link and provide an indication of this event to the DLS-user,
- detection of the failure of the path to another DLE and provide an indication of this event to the DLS-user,
- reception of a DLPDU that invites new DLE to connect to the link and provide an indication of this event to the DLS-user, and
- reception of a DLPDU from a DLE that is not in the neighbor table of the receiving DLE and provide an indication of this event to the DLS-user.

5.1.1.4 Local DL-management service

This service supports configuration of the DLL and access to DLL statistics. DL-management facilities provide a means for

- writing DLE configuration parameters;
- reading DLE configuration parameters, operational parameters and statistics;
- commanding major DLE actions; and
- receiving notification of significant DLE events.

Together these facilities constitute the DL-management-service (DLMS).

5.1.1.5 Security function

The DL provides security function for communications between pairs of communicating devices on one link. A key is used to ensure that the DLPDU is originated from an approved, authenticated device.

There is no explicit user service for security. The security parameters are written by using DL-management service.

5.1.2 QoS for message service

5.1.2.1 Priority

The message service primitives specify the priority of the transferred DLSDU. The four priorities from highest priority to lowest priority are defined below.

- Command – this priority should be used by the DLS-user to send network-related diagnostics, configuration, or control information DLSDU.
- Process-Data – this priority should be used by the DLS-user to send DLSDU that contains measurements from process transmitters or setpoints to control devices or any other process related data or network statistics data.
- Normal – this priority should be used by the DLS-user to send DLSDU that contains any other data not meeting the criteria for "Command", "Process-Data", or "Alarm" priority.
- Alarm – this priority should be used by the DLS-user to send DLSDU that contains only alarm or event DLSDU.

5.1.2.2 Timeout

Each message request specifies upper bounds on the maximum time duration permitted for the completion of each related instance of a sequence of message primitives.

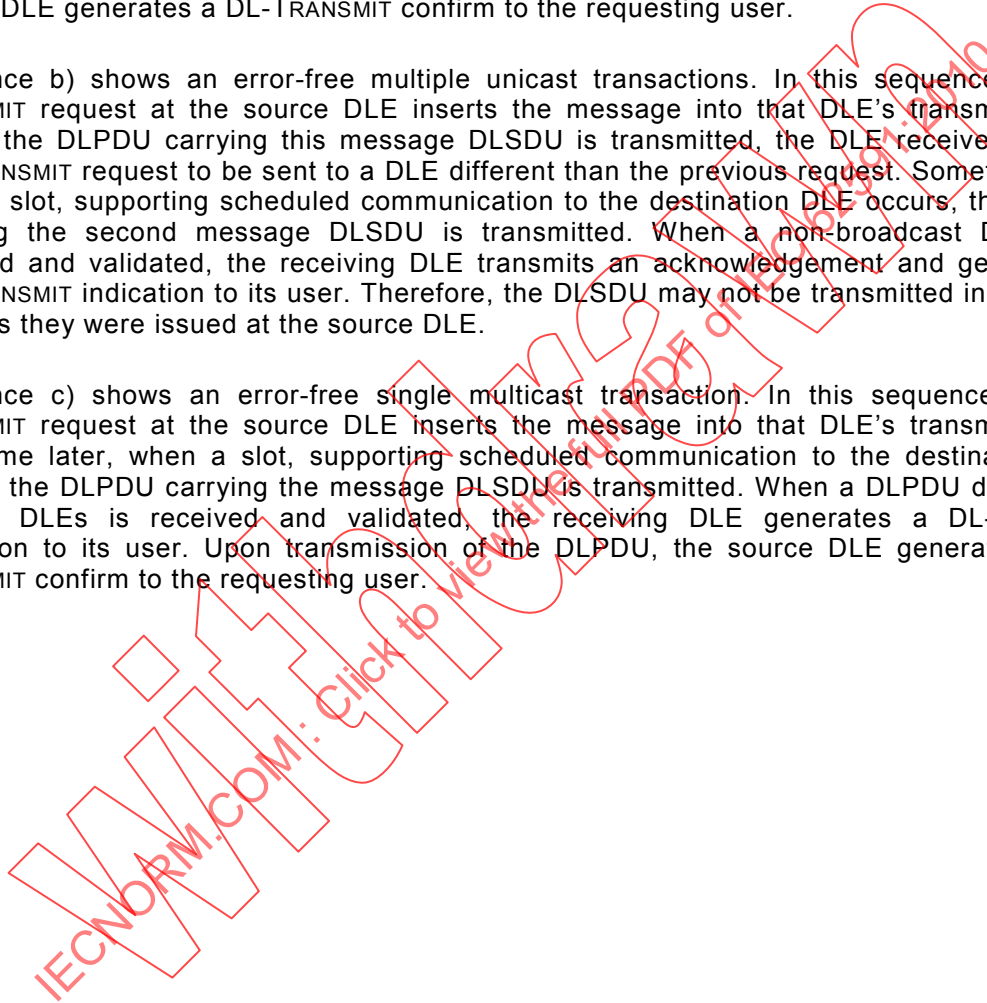
5.1.3 Sequence of primitives

The relations of primitives of each type at the interface of DLS-user and one DLE to primitives at the DLS-user and the other DLEs for error-free cases are summarized in the diagrams of Figure 6.

Sequence a) shows an error-free single unicast transaction. In this sequence the DL-TRANSMIT request at the source DLE inserts the message into that DLE's transmit queue. Sometime later, when a slot, supporting scheduled communication to the destination DLE occurs, the DLPDU carrying the message DLSDU is transmitted. When a non-broadcast DLPDU is received and validated, the receiving DLE transmits an acknowledgement and generates a DL-TRANSMIT indication to its user. Upon reception of the acknowledgement, the source DLE generates a DL-TRANSMIT confirm to the requesting user.

Sequence b) shows an error-free multiple unicast transactions. In this sequence the DL-TRANSMIT request at the source DLE inserts the message into that DLE's transmit queue. Before the DLPDU carrying this message DLSDU is transmitted, the DLE receives another DL-TRANSMIT request to be sent to a DLE different than the previous request. Sometime later, when a slot, supporting scheduled communication to the destination DLE occurs, the DLPDU carrying the second message DLSDU is transmitted. When a non-broadcast DLPDU is received and validated, the receiving DLE transmits an acknowledgement and generates a DL-TRANSMIT indication to its user. Therefore, the DLSDU may not be transmitted in the same order as they were issued at the source DLE.

Sequence c) shows an error-free single multicast transaction. In this sequence the DL-TRANSMIT request at the source DLE inserts the message into that DLE's transmit queue. Sometime later, when a slot, supporting scheduled communication to the destination DLE occurs, the DLPDU carrying the message DLSDU is transmitted. When a DLPDU destined to several DLEs is received and validated, the receiving DLE generates a DL-TRANSMIT indication to its user. Upon transmission of the DLPDU, the source DLE generates a DL-TRANSMIT confirm to the requesting user.



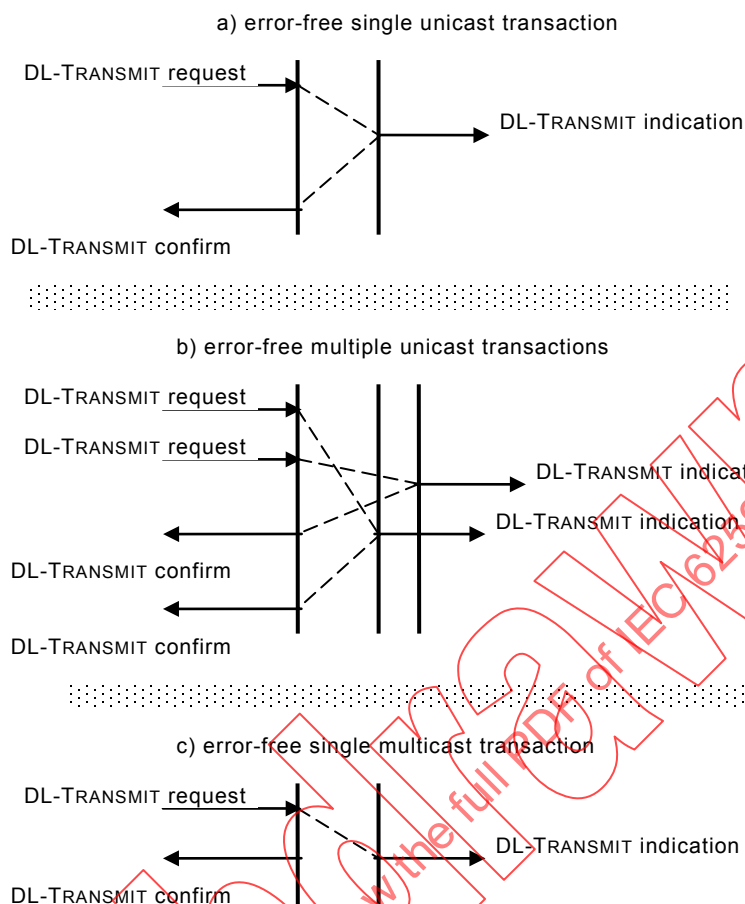


Figure 6 – Message service error-free sequences

The relations of primitives of each type at the interface of DLS-user and one DLE to primitives at the DLS-user and the other DLEs for error cases are summarized in the diagrams of Figure 7.

The sequence a) shows the DLPDU lost in the transmission. The sequence b) shows that the receiving DLE discarded the DLPDU due to lack of buffer resource and sends an acknowledgment with a response code of "Error". As shown, the source DLE retransmits the message until either an acknowledgement with a response code of "Success" is received, or a limit on maximum number of retries is reached. If the packet resides in the source DLE without an acknowledgement from the destination DLE, for a duration which is greater than the specified timeout value, then the DLE stops retrying.

The sequence c) shows that the destination DLE received the DLPDU with either an invalid message integrity code or a destination address that does not match. The receiving DLE discards the DLPDU and does not send any acknowledgment.

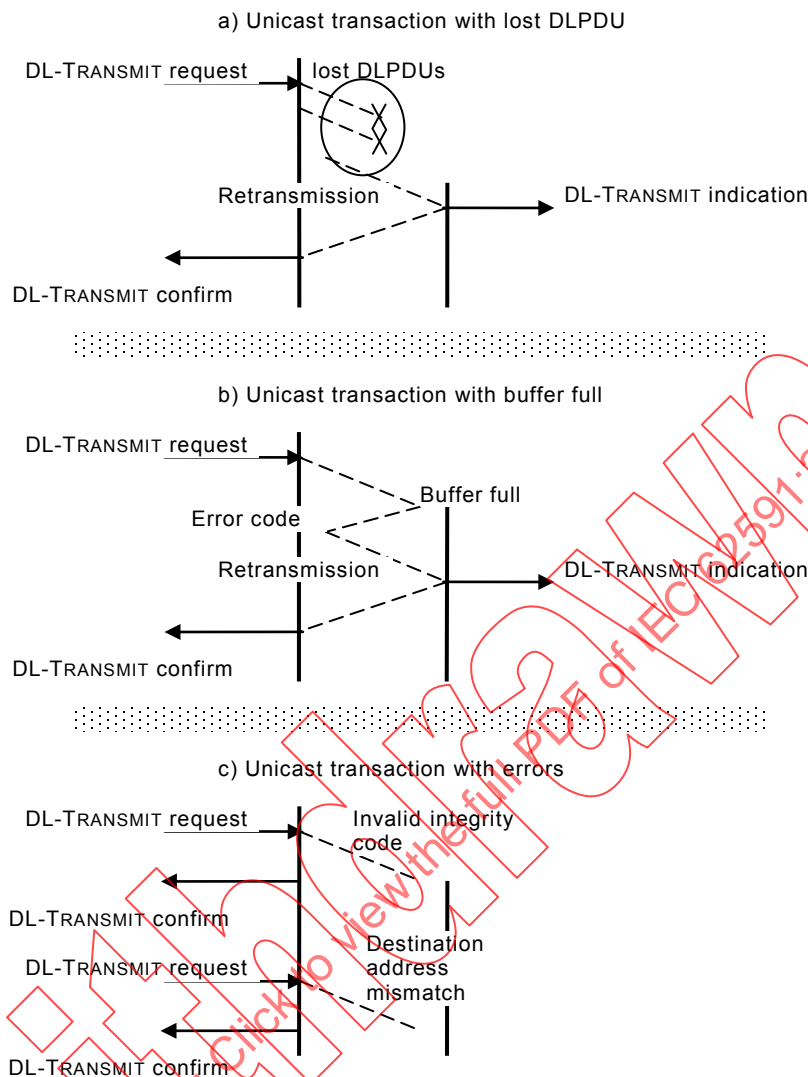


Figure 7 – Message service sequences with errors

Figure 8 shows the receive-only primitive. This is available in a DLE that supports a promiscuous operating mode where communications from other DLEs are captured.

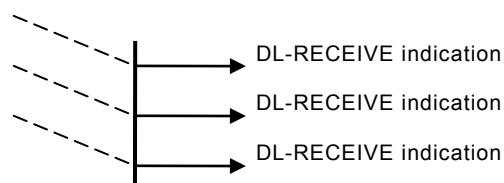


Figure 8 – Receive only message service

5.1.4 DL-TRANSMIT service

5.1.4.1 Function

It can be used to transmit an independent, self-contained DLSDU from one DLE to another DLE in a single service access, and to return the status of that delivery to the originating DLS-user.

5.1.4.2 Types of primitives and parameters

Table 9 indicates the types of primitives and the parameters needed for the DL-Transmit service.

Table 9 – DL-Transmit primitives and parameters

Parameter name	Request	Indication	Confirm
Handle	M		M(=) ^a
Broadcast	M	M	
Source address		M	
Destination address			
Short address	S		
Long address	S		
Superframe_ID	C		
Graph	C		
Priority	M	M(=)	
Timeout	M		
DLS-user-data	M	M(=)	
Status		M	M
a The value of this parameter is equal to the value in the request primitive.			

5.1.4.3 Parameters

5.1.4.3.1 Handle

This parameter is supported for the convenience of the DLS-user. Its value in confirm primitive is identical to the value in the corresponding request primitive.

5.1.4.3.2 Broadcast

If the value of this parameter is TRUE, then the DLSDU is broadcast in the superframe specified in the request primitive.

5.1.4.3.3 Source address

This parameter identifies the source of the DLSDU conveyed in the indication primitive.

5.1.4.3.4 Destination address

5.1.4.3.4.1 Short address

This parameter is present only if Broadcast is FALSE. It is the nickname of the destination DLE that is used for the next hop.

5.1.4.3.4.2 Long address

This parameter is present only if Broadcast is FALSE. It is the Unique ID of the destination DLE that is used for the next hop.

5.1.4.3.5 Superframe_ID

This parameter is present only if Broadcast is TRUE. It specifies the superframe whose broadcast links has to be used to transmit the DLSDU.

5.1.4.3.6 Graph

This parameter is present only if graph routing is to be used. When used, the graph indicates the neighbors that can be used as the destination for the next hop.

5.1.4.3.7 Priority

This parameter specifies the priority for the transmission. Its value is any one member of the set: {command, process-data, normal, alarm}.

5.1.4.3.8 Timeout

This is the maximum time duration for the completion of each related instance of a sequence of DL-TRANSMIT primitives.

5.1.4.3.9 DLS-user-data

This parameter allows the transmission of data between DLS-users without alteration by the DLS-provider.

5.1.4.3.10 Status

In indication primitive, this parameter indicates the data-link key used to authenticate the DLPDU and whether the DLPDU was a broadcast.

In confirm primitive, this parameter indicates that the service was provided successfully, or failed for the reason specified.

5.1.5 DL-RECEIVE service

5.1.5.1 Function

It can be used by a DLE to receive any DLPDU and provide the DLSDU in that DLPDU to it DLS-user. This service is also called "promiscuous operating mode" and can be used for network troubleshooting.

5.1.5.2 Type of primitive and parameters

Table 10 indicates the type of primitive and the parameters needed for the Receive service.

Table 10 – DL-RECEIVE primitive and parameters

Parameter name	Indication
Packet RSL	M
DLS-user-data	M

5.1.5.3 Parameters

5.1.5.3.1 Packet RSL

This parameter is RSL of the just received DLPDU.

5.1.5.3.2 DLS-user-data

This parameter conveys the data received by the DLE.

5.1.6 DL-DISCONNECT service

5.1.6.1 Function

This service notifies that another DLE is disconnecting from the link. A DLPDU has been received from that DLE indicating that it is disconnecting from the link.

NOTE The Disconnect DLPDU is expected only from a device that is a neighbor. It is possible that the receiving device does not know all of its neighbors. Therefore it always receives Disconnect DLPDU and provides an indication, even if it is from a device that is not known as neighbor.

5.1.6.2 Type of primitive and parameters

5.1.6.2.1 General

Table 11 indicates the type of primitive and the parameters needed for the Disconnect service.

Table 11 – DL-DISCONNECT primitive and parameters

Parameter name	Indication
Source address	M

5.1.6.2.2 Source address

This parameter identifies the address of the DLE that just disconnected.

5.1.7 DL-PATH-FAILURE service

5.1.7.1 Function

This service notifies that the path to a DLE with which this DLE is connected has failed. The communications with the indicated DLE has timed out unexpectedly and the DLE appears to be unavailable.

5.1.7.2 Type of primitive and parameters

5.1.7.2.1 General

Table 12 indicates the type of primitive and the parameters needed for PATH-FAILURE service.

Table 12 – DL-PATH-FAILURE primitive and parameters

Parameter name	Indication
Source address	M

5.1.7.2.2 Source address

This parameter identifies the address of the DLE to which the path has just failed.

5.1.8 DL-ADVERTISE service

5.1.8.1 Function

This service notifies that the DLE has received Advertise DLPDU. The DLS-user should use the information provided in this service before joining the network.

5.1.8.2 Type of primitive and parameters

5.1.8.2.1 General

Table 13 indicates the type of primitive and the parameters needed for the Advertise service.

Table 13 – DL-ADVERTISE primitive and parameters

Parameter name	Indication
DLS-user-data	M

5.1.8.2.2 DLS-user-data

This parameter conveys the data received by the DLE in the Advertise DLPDU.

5.1.9 DL-NEIGHBOR service

5.1.9.1 Function

This service notifies that the DLE has received a DLPDU from a DLE that is not listed in its neighbor table (see 6.3.3.4).

5.1.9.2 Type of primitive and parameters

Table 14 indicates the type of primitive and the parameters needed for the neighbor service.

Table 14 – DL-NEIGHBOR primitive and parameters

Parameter name	Indication
Source address	M
Packet RSL	M

5.1.9.3 Parameters

5.1.9.3.1 Source address

This parameter identifies the address of the DLE that is not in its neighbor table.

5.1.9.3.2 Packet RSL

This primitive indicates the RSL of the received DLPDU.

5.1.10 DLM-SET

5.1.10.1 Function

This primitive can be used to set (write) the value of a DLE configuration parameter.

5.1.10.2 Types of parameters

Table 15 indicates the primitive and parameters of the Set service.

Table 15 – DLM-SET primitive and parameters

Parameter name	Request	
	Input	Output
DLM-object-identifier	M	
Desired-value	M	
Status		M

5.1.10.3 Parameters**5.1.10.3.1 DLM-object-identifier**

This parameter specifies the primitive or composite object within the DLE whose value is to be altered. The naming-domain of the DL-management (DLM) object-identifier is the DLM local-view.

5.1.10.3.2 Desired-value

This parameter specifies the desired value for the DLM object specified by the associated DLM-object-identifier. Its type is identical to that of the specified DLM object.

5.1.10.3.3 Status

This parameter allows the DLM service (DLMS) user to determine whether the requested DLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.1.11 DLM-GET**5.1.11.1 Function**

This primitive can be used to get (read) the value of a DLE configuration parameter, operational parameter or statistics.

5.1.11.2 Types of parameters

Table 16 indicates the primitive and parameters of the GET service.

Table 16 – DLM-GET primitive and parameters

Parameter name	Request	
	Input	Output
DLM-object-identifier	M	
Status		M
Current-value		C

5.1.11.3 Parameters**5.1.11.3.1 DLM object-identifier**

This parameter specifies the primitive or composite object within the DLE whose value is being requested. The naming-domain of the DLM-object-identifier is the DLM local-view.

5.1.11.3.2 Status

This parameter allows the DLMS-user to determine whether the requested DLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.1.11.3.3 Current-value

This parameter is present when the status parameter indicates that the requested service was performed successfully. It specifies the current value for the DLM-object specified by the associated DLM-object-identifier. Its type is identical to that of the specified DLM-object.

5.1.12 DLM-ACTION

5.1.12.1 Function

This primitive can be used to command a specified action of the DLE.

5.1.12.2 Types of parameters

Table 17 indicates the primitives and parameters of the ACTION service.

Table 17 – DLM-ACTION primitive and parameters

	Request	Confirm
Parameter name	Input	Output
Desired-action	M	
Action-qualifiers	C	
Status		M
Additional-information		C

5.1.12.3 Parameters

5.1.12.3.1 Desired-action

This parameter specifies the desired action which the DLE is to take. The possible values for this parameter are listed in Table 47.

5.1.12.3.2 Action-qualifiers

This parameter specifies additional action-specific parameters which serve to qualify the commanded action.

NOTE Some of the actions require additional parameters, such as DELETE-SUPERFRAME requires Superframe_ID.

5.1.12.3.3 Status

This parameter allows the DLMS-user to determine whether the requested DLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.1.12.3.4 Additional-information

This optional parameter provides action-specific additional information which augments the returned status.

5.1.13 DLM-EVENT

5.1.13.1 Function

This primitive is used to report the occurrence of a DL-event which could be of significance to DL-management.

5.1.13.2 Types of parameters

No parameters defined.

5.2 Network layer services

5.2.1 Facilities of the network layer services

5.2.1.1 Data transfer service

The network layer provides service to transmit Network layer (NL) service-data-unit (NSDU) from source NL-entity (NLE) to destination NLE. It may require routing an NL-protocol-data-unit (NPDU) through intermediate NLEs. The route for a NPDU is setup using NL-management services (NLMS).

5.2.1.2 Local NL-management service

This service supports configuration of the NL and access to NL statistics. NL-management facilities provide a means for

- writing NLE configuration parameters;
- reading NLE configuration parameters, operational parameters and statistics;
- commanding major NLE actions; and
- receiving notification of significant NLE events.

Together these facilities constitute the NLMS

5.2.1.3 Security function

The NL provides security function for end to end communications between pairs of communicating devices. NL is session oriented for security purpose. A session enables private and secure communication between a pair of network addresses. The following types of security keys are supported.

- Join – this key is used when the device tries to join the network.
- Network manager – this key is for communication between network manager and a network device.
- Gateway – this key is for communication between gateway and a network device.

There is no explicit user service for security. The security parameters are written by using NL-management service.

5.2.2 QoS for message service

The data transfer service primitives specify the priority of the transferred NSDU. The four priorities are, from highest priority to lowest priority:

- Command – this priority should be used by the NLS-user to send network-related diagnostics, configuration, or control information data.
- Process-Data – this priority should be used by the NLS-user to send data that contains measurements from process transmitters or set-points to control devices or any other process related data or network statistics data.

- Normal – this priority should be used by the NLS-user to send data that contains any other data not meeting the criteria for "Command", "Process-Data", or "Alarm" priority.
- Alarm – this priority should be used by the NLS-user to send data that contains only alarm or event information.

5.2.3 Sequence of primitives

The relations of primitives at the end-points of message service of each type at the interface of NLS-user and one NLE to primitives at the NLS-user and the other NLEs are summarized in the diagrams of Figure 9. After the DLE provides the confirmation of transmission, the NLE generates a NL-Transmit confirm to its requesting user. There is no confirmation of delivery from the remote NLE.

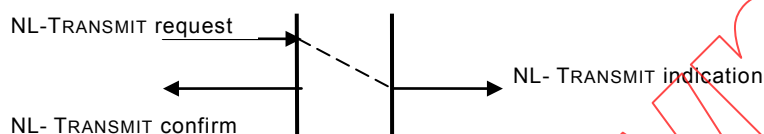


Figure 9 – NLL Data transfer service sequences

5.2.4 NL-TRANSMIT Service

5.2.4.1 Function

It can be used to transmit an independent, self-contained NSDU from one NLE to another NLE in a single service access, and to return the status of that delivery to the originating NLS-user.

5.2.4.2 Types of primitives and parameters

Table 18 indicates the types of primitives and the parameters needed for the Transmit service.

Table 18 – NL-TRANSMIT primitives and parameters

Parameter name	Request	Indication	Confirm
Handle	M		M(=) ^a
Source address		M	
Destination address			
Short address	S		
Long address	S		
Broadcast	S		
Priority	M	M(=)	
Timetable_ID	C		
NLS-user-data	M	M(=)	
Status			M
^a The value of this parameter is equal to the value in the request primitive.			

5.2.4.3 Parameters

5.2.4.3.1 Handle

In the request, this parameter is supported for the convenience of the NLS-user. Its value in confirm primitive is identical to the value in the corresponding request primitive.

5.2.4.3.2 Source address

This parameter identifies the source of the NSDU conveyed in the indication primitive.

5.2.4.3.3 Destination address

5.2.4.3.3.1 Short address

It is the nickname of the destination NLE.

5.2.4.3.3.2 Long address

It is the unique ID of the destination NLE.

5.2.4.3.3.3 Broadcast

All NLEs in the network are the destinations.

5.2.4.3.4 Priority

This parameter specifies the priority for the transmission. Its value is any one member of the set: {command, process-data, normal, alarm}.

5.2.4.3.5 Timetable_ID

This parameter specifies the Timetable that was used to allocate the network communication bandwidth for this service primitive. This parameter is present only if the destination address is not that of the network manager and if this request is required to use the allocated bandwidth.

5.2.4.3.6 NLS-user-data

This parameter allows the transmission of data between NLS-users without alteration by the NLS-provider.

5.2.4.3.7 Source address

This parameter identifies the source of the NSDU conveyed in the indication primitive.

5.2.4.3.8 Status

In confirm primitive, this parameter indicates success, warning or failure of the service. For warning and failure, it also indicates the cause of the warning or error.

5.2.5 NLM-SET

5.2.5.1 Function

This primitive can be used to set (write) the value of a NLE configuration parameter.

5.2.5.2 Types of parameters

Table 19 indicates the primitive and parameters of the SET service.

Table 19 – NLM-SET primitive and parameters

Parameter name	Request	
	Input	Output
NLM-object-identifier	M	
Desired-value	M	
Status		M

5.2.5.3 Parameters

5.2.5.3.1 NLM-object-identifier

This parameter specifies the primitive or composite object within the NLE whose value is to be altered. The naming-domain of the NLM-object-identifier is the NLM local-view.

5.2.5.3.2 Desired-value

This parameter specifies the desired value for the NLM object specified by the associated NLM-object-identifier. Its type is identical to that of the specified NLM object.

5.2.5.3.3 Status

This parameter allows the NLMS-user to determine whether the requested NLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.2.6 NLM-GET

5.2.6.1 Function

This primitive can be used to get (read) the value of a NLE configuration parameter, operational parameter or statistics.

5.2.6.2 Types of parameters

Table 20 indicates the primitive and parameters of the GET service.

Table 20 – NLM-GET primitive and parameters

Parameter name	Request	
	Input	Output
NLM-object-identifier	M	
Status		M
Current-value		C

5.2.6.3 Parameters

5.2.6.3.1 NLM-object-identifier

This parameter specifies the primitive or composite object within the NLE whose value is being requested. The naming-domain of the NLM-object-identifier is the NLM local-view.

5.2.6.3.2 Status

This parameter allows the NLMS-user to determine whether the requested NLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.2.6.3.3 Current-value

This parameter is present when the status parameter indicates that the requested service was performed successfully. This parameter specifies the current value for the NLM-object specified by the associated NLM-object-identifier. Its type is identical to that of the specified NLM-object.

5.2.7 NLM-ACTION

5.2.7.1 Function

This primitive can be used to command a specified action of the NLE.

5.2.7.2 Types of parameters

Table 21 indicates the primitives and parameters of the Action service.

Table 21 – NLM-ACTION primitive and parameters

	Request	Confirm
Parameter name	Input	Output
Desired-action	M	
Action-qualifiers	C	
Status		M
Additional-information		C

5.2.7.3 Parameters

5.2.7.3.1 Desired-action

This parameter specifies the desired action which the NLE is to take. The possible values for this parameter are listed in Table 59.

5.2.7.3.2 Action-qualifiers

This parameter specifies additional action-specific parameters which serve to qualify the commanded action.

NOTE Some of the actions require additional parameters, such as DELETE-SESSION requires Session ID.

5.2.7.3.3 Status

This parameter allows the NLMS-user to determine whether the requested NLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.2.7.3.4 Additional-information

This optional parameter provides action-specific additional information which augments the returned status.

5.3 Transport layer services

5.3.1 Facilities of the transport layer services

5.3.1.1 Data transfer service

The transport layer (TL) supports both acknowledged and unacknowledged services.

The Unacknowledged service allows devices to send packets without requiring acknowledgement from the receiving end. The packets may be delivered at the destination device in a different order than the order in which it was originally sent from the source device. This service is useful for publishing the process data. Since process data is propagated periodically, end to end acknowledgement and retries have limited utility considering a new data point will be generated on a regular basis.

The acknowledged service is used to construct a synchronous transport pipe across the network connecting two devices. This pipe allows devices to send packets and confirm their delivery. The TL-service (TLS) user can send packet which will be delivered without loss, duplication or misordering to the receiving TLS-user. This service is suited for user request and response traffic and event notifications. When the acknowledged service is used, the communication is synchronous. Only one outstanding transaction across the network for that transport pipe is permitted at a time.

Using acknowledged service allows the client layer to ensure retries are automatically performed, if communication latency becomes excessive. Statistically, communication reliability for a well-formed Type 20 mesh network is greater than 3σ ($3\text{-sigma} = 99,730\ 020\ 4\%$) and normally greater than 4σ ($99,999\ 5\%$). The primary benefit provided by the acknowledged service is that the client layer will receive positive acknowledgement after TL-protocol-data-unit (TPDU) arrival at the destination. This allows synchronous operation across the network as well as assuring the client layer that the TPDU was delivered and acknowledged.

5.3.1.2 Local TL-management service

This service supports configuration of the TL and access to TL statistics. TL-management facilities provide a means for

- writing TL-entity (TLE) configuration parameters; and
- reading TLE configuration parameters, operational parameters and statistics;

Together these facilities constitute the NL-management-service (NLMS).

5.3.2 QoS for message service

5.3.2.1 Priority

The data transfer service primitives specify the priority of the transferred data. The four priorities are, from highest priority to lowest priority:

- Command – this priority should be used by the TLS-user to send network-related diagnostics, configuration, or control information data.
- Process-Data – this priority should be used by the TLS -user to send data that contains measurements from process transmitters or set-points to control devices or any other process related data or network statistics data.
- Normal – this priority should be used by the TLS -user to send data that contains any other data not meeting the criteria for "Command", "Process-Data", or "Alarm" priority.
- Alarm – this priority should be used by the TLS -user to send data that contains only alarm and event information.

5.3.2.2 Retry

The TL-management specifies upper bounds on the maximum time duration permitted for the completion of each related instance of a sequence of data transfer primitives. It also specifies the maximum number of retries to send request and receive response primitive.

5.3.3 Sequence of primitives

This subclause specifies relation of primitives at the end-points of data transfer service.

The relations of primitives of each type at the interface of TLS-user and one TLE to primitives at the TLS-user and the other TLEs for error-free cases are summarized in the diagrams of Figure 10.

Sequence a) shows an error-free unidirectional transfer of data without acknowledgment from the destination TLE. After the NLE provides the confirmation of transmission, the TLE generates a TL-DATA-TRANSFER confirm to its requesting user.

Sequence b) shows an error-free bidirectional acknowledged exchange of data. In this sequence, TL-DATA-EXCHANGE request at the source TLE inserts the message into transmit queue of the TLE. When a TPDU is received and validated, the destination TLE generates a TL-DATA-EXCHANGE indication. The TLS-user at that destination is expected to provide response data. The TLE transmits this response to the source TLE. Upon reception of the response, the source TLE generates a TL-DATA-EXCHANGE confirm to its user.

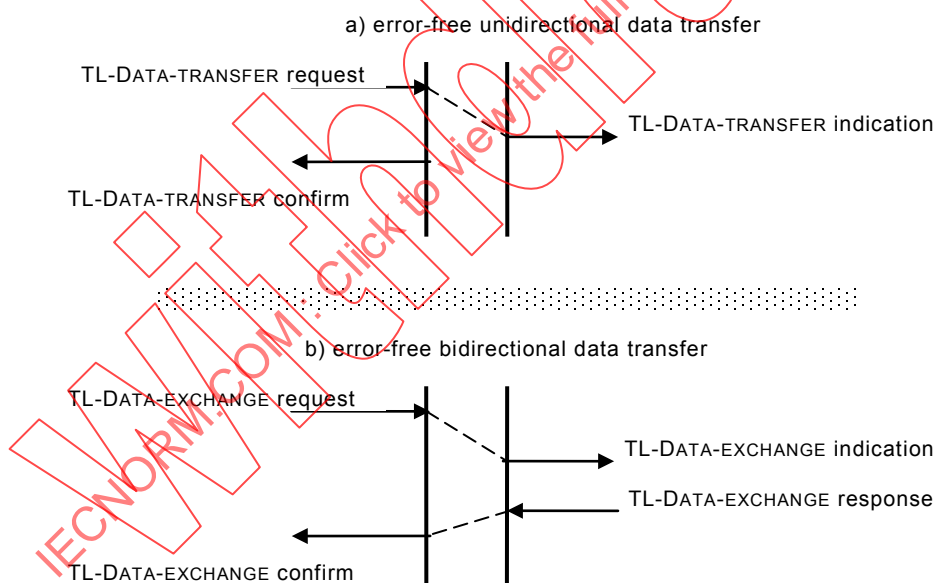


Figure 10 – Data transfer service error-free sequences

The relations of primitives of each type at the interface of TLS-user and one TLE to primitives at the TLS-user and the other TLEs for error cases are summarized in the diagram of Figure 11. The sequence shows the TL-protocol-data-unit (TPDU) lost in the transmission. As shown, the source TLE retransmits the TPDU until either a response TPDU is received, or a limit on maximum number of retries is reached. If the packet resides in the source TLE without a response from the destination TLE, for a duration which is greater than the timeout value, then the TLE stops retrying.

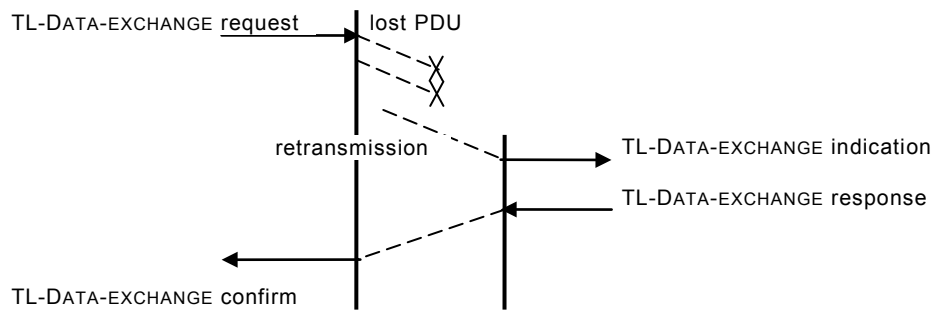


Figure 11 – Data transfer service sequences with errors

5.3.4 TL-DATA-TRANSFER Service

5.3.4.1 Function

It can be used to transmit an independent, self-contained request TL-service-data-unit (TSDU) from one TLE to another TLE in a single service access without any delivery confirmation.

5.3.4.2 Types of primitives and parameters

Table 22 indicates the types of primitives and the parameters needed for the Data-transfer service.

Table 22 – TL-DATA-TRANSFER primitives and parameters

Parameter name	Request	Indication	Confirm
Handle	M	M(=) ^a	M(=) ^b
Transport type	M	M	
Source address		M	
Destination address			
Short address	S		
Long address	S		
Broadcast	S		
Priority	M	M(=)	
Timetable_ID	C		
TLS-user-data	M	M(=)	
Status			M
^a This parameter in indication is considered as the sequence number.			
^b The value of this parameter is equal to the value in the request primitive.			

5.3.4.3 Parameters

5.3.4.3.1 Handle

In the request, this parameter is supported for the convenience of the DLS-user. Its value in confirm primitive is identical to the value in the corresponding request primitive. In the indication primitive this parameter is used to provide a sequence number.

5.3.4.3.2 Transport type

In the request primitive, this parameter indicates the type of service requested. Its value is one of the following:

- Block-transfer request,
- Block-transfer response,
- Search-broadcast,
- Publish-broadcast,
- Response-broadcast,
- Publish-unicast.

5.3.4.3.3 Source address

This parameter identifies the source of the TSDU conveyed in the indication primitive.

5.3.4.3.4 Destination address

5.3.4.3.4.1 Short address

It is the nickname of the destination TLE.

5.3.4.3.4.2 Long address

It is the Unique ID of the destination TLE.

5.3.4.3.4.3 Broadcast

All TLEs in the network are the destinations.

5.3.4.3.5 Priority

This parameter specifies the priority for the transmission. Its value is any one member of the set: {command, process-data, normal, alarm}.

5.3.4.3.6 Timetable_ID

This parameter specifies the Timetable that was used to allocate the network communication bandwidth for this service primitive. This parameter is present only if the destination address is not that of the network manager.

5.3.4.3.7 TLS-user-data

This parameter allows the transmission of data between TLS-users without alteration by the TLS-provider.

5.3.4.3.8 Source address

This parameter identifies the source of the TSDU conveyed in the indication primitive.

5.3.4.3.9 Status

In confirm primitive, this parameter indicates success, warning or failure of the service. For warning and failure, it also indicates the cause of the warning or error.

5.3.5 TL-DATA-EXCHANGE Service

5.3.5.1 Function

It can be used to transmit an independent, self-contained request TSDU from one TLE to another TLE in a single service access, to receive response TSDU from the destination TLE and to return the status of that exchange to the originating TLS-user.

5.3.5.2 Types of primitives and parameters

Table 23 indicates the types of primitives and the parameters needed for the TL-DATA-EXCHANGE service.

Table 23 – TL-DATA-EXCHANGE primitives and parameters

Parameter name	Request	Indication	Response	Confirm
Handle	M	M(=) ^a	M(=)	M(=) ^b
Transport type	M	M(=)		
Source address		M		
Destination address				
Short address	S			
Long address	S			
Broadcast	S			
Priority	M	M(=)		
Timetable_ID	C			
TLS-user-data	M	M(=)	M	C(=)
Status				M
^a This parameter in indication is considered as the sequence number.				
^b This parameter is equal to the value in the request primitive.				

5.3.5.3 Parameters

5.3.5.3.1 Handle

In the request, this parameter is supported for the convenience of the DLS-user. Its value in confirm primitive is identical to the value in the corresponding request primitive. In the indication primitive this parameter is used to associate the request with the response.

5.3.5.3.2 Transport type

In the request primitive, this parameter indicates the type of service requested by master. Its value is one of the following:

- Request-unicast,
- Request-broadcast.

In the response primitive, this parameter indicates the type of service requested by slave. Its value is – Response-unicast.

5.3.5.3.3 Source address

This parameter identifies the source of the TSDU conveyed in the indication primitive.

5.3.5.3.4 Destination address

5.3.5.3.4.1 Short address

It is the nickname of the destination TLE.

5.3.5.3.4.2 Long address

It is the Unique ID of the destination TLE.

5.3.5.3.4.3 Broadcast

All TLEs in the network are the destinations.

5.3.5.3.5 Priority

This parameter specifies the priority for the transmission. Its value is any one member of the set: {command, process-data, normal, alarm}.

5.3.5.3.6 Timetable_ID

This parameter specifies the Timetable that was used to allocate the network communication bandwidth for this service primitive. This parameter is present only if the destination address is not that of the network manager.

NOTE The communication bandwidth allocation is requested by the transaction initiator and the response is not included in the allocation for any Timetables in the device. Therefore, Timetable_ID is not included in the response primitive.

5.3.5.3.7 TLS-user-data

This parameter allows the transmission of data between TLS-users without alteration by the TLS-provider.

5.3.5.3.8 Source address

This parameter identifies the source of the TSDU conveyed in the indication primitive.

5.3.5.3.9 Status

In confirm primitive, this parameter indicates success, warning or failure of the service. For warning and failure, it also indicates the cause of the warning or error.

5.3.6 TLM-SET

5.3.6.1 Function

This primitive can be used to set (write) the value of a TLE configuration parameter.

5.3.6.2 Types of parameters

Table 24 indicates the primitive and parameters of the SET service.

Table 24 – TLM-Set primitive and parameters

Parameter name	Request	
	Input	Output
TLM-object-identifier	M	
Desired-value	M	
Status		M

5.3.6.3 Parameters

5.3.6.3.1 TLM-object-identifier

This parameter specifies the primitive or composite object within the TLE whose value is to be altered. The naming-domain of the TLM-object-identifier is the TLM local-view.

5.3.6.3.2 Desired-value

This parameter specifies the desired value for the TLM object specified by the associated TLM object-identifier. Its type is identical to that of the specified TLM object.

5.3.6.3.3 Status

This parameter allows the TL-management service (TLMS) user to determine whether the requested TLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.3.7 TLM-GET

5.3.7.1 Function

This primitive can be used to get (read) the value of a TLE configuration parameter, operational parameter or statistic.

5.3.7.2 Types of parameters

Table 25 indicates the primitive and parameters of the GET service.

Table 25 – TLM-Get primitive and parameters

Parameter name	Request	
	Input	Output
TLM-object-identifier	M	
Status		M
Current-value		C

5.3.7.3 Parameters

5.3.7.3.1 TLM-object-identifier

This parameter specifies the primitive or composite object within the TLE whose value is being requested. The naming-domain of the TLM-object-identifier is the TLM local-view.

5.3.7.3.2 Status

This parameter allows the TLMS-user to determine whether the requested TLMS was provided successfully, or failed.

NOTE The failure reason may also be provided.

5.3.7.3.3 Current-value

This parameter is present when the status parameter indicates that the requested service was performed successfully. This parameter specifies the current value for the TLM-object specified by the associated TLM-object-identifier. Its type is identical to that of the specified TLM-object.

6 Data-link layer protocol specification – Type 20 elements

6.1 Overview

6.1.1 Data-link layer operation

6.1.1.1 Two sublayer model of DLL

The DLL is modeled as

- a higher sublayer logical link control (LLC), and
- a lower sublayer medium access control (MAC).

Interoperating with all sublayers are DL-management functions.

6.1.1.2 Logical link control

The LLC sublayer provides the higher-level functionality of

- managing all DLE interactions with the DLS-user, converting all DLS-user request and response primitives into the necessary sequence of DLE operations, and generating DLS-user indication and confirm primitives where appropriate;
- preparation of the DL PDU for transmission;
- parsing of the received DL PDU;
- flow control;
- error detection;
- maintenance of the neighbor table; and
- security at the data-link layer.

6.1.1.3 Medium access control

The Type 20 data-link layer uses Time Division Multiple Access (TDMA) and channel hopping to control access to the link. This Medium access control (MAC) technique provides collision free, deterministic communication between two devices. The communication between two devices occurs during one time slot. A TDMA superframe consists of a series of time slots. The basic structure of TDMA time slots and the superframe are shown in Figure 12. Slot sizes and the superframe length in number of slots are fixed and form a link cycle. All devices support multiple superframes, starting with superframe number zero (0). At least one superframe is always enabled while additional superframes can be enabled or disabled. The length of a superframe can be changed only while it is inactive. Superframe may have lengths that differ from each other. Superframes are repeated continuously with a fixed repetition rate.

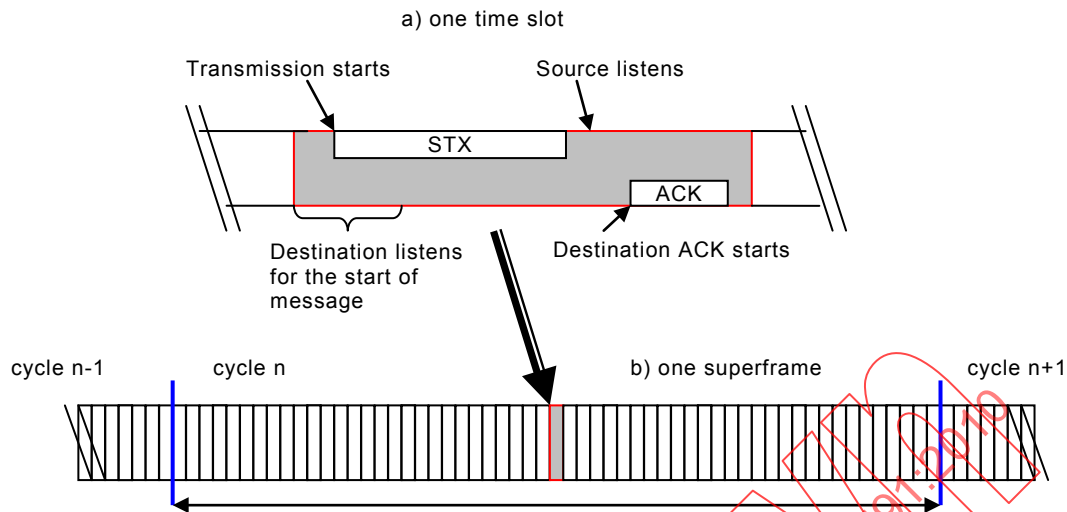


Figure 12 – A TDMA slot and superframe

Typically, two devices are assigned to one time slot. One is designated as the source and the other as the destination. A link transaction within a slot supports the transmission of one DLPDU from a source device and if required it is followed immediately by the transmission of an acknowledgement (ACK) DLPDU by the destination device. The ACK DLPDU contains either "Success" response code indicating the initial DLPDU was successfully received and handled, or an error response code. An error response code indicates that the initial DLPDU was successfully received, but that further processing failed, for example there were no buffers available in the receiving device.

A broadcast message, using the Data-Link destination address as the broadcast address, is never acknowledged. In this case, multiple receivers are assigned to the same slot.

For successful and efficient TDMA communications, it is imperative that devices know when the start of a time slot occurs. This requires synchronization of clocks among all devices in the network. To ensure network-wide device clock synchronization, the tolerances on time keeping and time synchronization mechanisms are specified in this protocol.

Within a time slot, transmission of the source DLPDU starts at a specified offset from the start of the time slot. This offset allows the source and destination to set their frequency channel and allows the receiver to begin listening on the specified channel. Since there is a tolerance on clocks, the receiver starts to listen before the ideal transmission start time and continues to listen after that ideal time. Once the transmission is complete, the communication direction is reversed and the destination device indicates, by transmitting an ACK, whether it received the source device's DLPDU successfully or with a specific class of detected errors. If the destination did not hear any transmission or if there was an error in reception which is not included in this class of errors, then the destination device does not transmit ACK DLPDU.

6.1.1.4 Channel hopping

To enhance reliability, channel hopping is combined with TDMA as shown in Figure 13. Channel hopping provides frequency diversity, which can avoid interferers and reduce multi-path fading effects. TDMA enables efficient, low-power and reliable channel hopping communication because the synchronization of the slot and channel used by the communicating devices allows them to rendezvous in time and frequency, thus promoting successful communications.

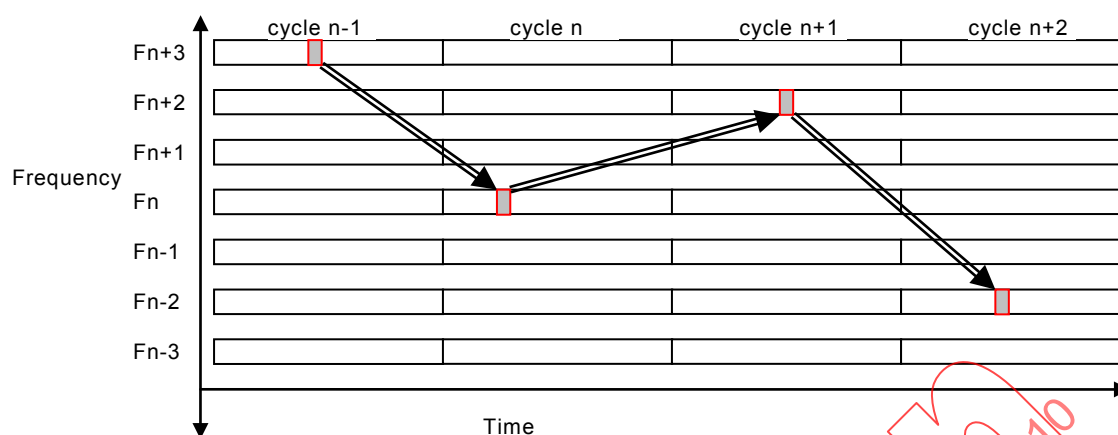


Figure 13 – Channel hopping

Communicating devices are assigned to a superframe, a slot within that superframe and a channel offset. This trio forms a communications link between two communicating devices. All devices support multiple links. The number of possible links is, typically, equal to the number of channels utilized by a network times the number of slots in the superframe. For example using 15 channels and 9 000 slots per superframe results in 135 000 possible links.

Channel hopping provides channel diversity, so each slot is used on multiple channels at the same time by different nodes. This can be achieved by creating links on the same slot, but with different channel offsets. Each device maintains a list of channels in use and the specification (for example the frequency) for that channel. All devices in a network have identical channel lists determined by the network manager. Assignment of links and the devices in a link is the responsibility of the network manager.

Channel blacklisting is a feature of this protocol that allows the network administrator to restrict the channel hopping by the devices to selected channels in the available RF band. For example network administrators can blacklist certain channels in order to protect a wireless service that uses a fixed portion of the RF band that could have been shared by the devices in this network. In practice, communication in the available bandwidth by the devices using other protocols is very random and uses a tiny amount of the total available bandwidth. Consequently, blacklisting seldom provides tangible benefits.

6.1.1.5 Network maintenance

Time and channel are the first two dimensions of a Type 20 network. The third dimension is space (distance). Network devices are installed at various locations in a plant and a given device has a set of other devices with which it can communicate directly. This set of devices is defined as neighborhood. Since the RF environment is subject to change, a device keeps its list of neighbor current. Maintenance activities include discovering potential neighbors, gathering statistics about the communication channel to each neighbor, and maintaining time synchronization with neighbors.

Two special DLPDUs assist in building and maintaining a device's list of neighbors. These are Advertise DLPDU and Keep-alive DLPDU. These DLPDUs contain sufficient information to allow new neighbors to be discovered or to allow a newly installed device to request admission to the network. If accepted, the new device can become a neighbor of the advertising device.

NOTE The network manager can schedule transmission of these DLPDUs.

Every successful communication with a neighbor confirms the neighbor's presence and allows the quality of the communication link to be assessed. Since communications only occurs when the device has a DLPDU for the neighbor, there can be long intervals during which the link is

not exercised. Keep-alive DLPDUs are used to probe quiescent links and to maintain time synchronization.

6.1.1.6 Time keeping

Time synchronization across the network is essential for TDMA communications. Irrespective of the choice of device hardware time source (for example crystals, ceramic resonators etc.), some skew between devices (for example due to temperature or voltage variations or ageing) is inevitable. Consequently, Type 20 data-link layer has several mechanisms to synchronize network-wide time.

When a device receives a DLPDU destined to it, its time of arrival is noted. Using this information the destination calculates the difference between the actual time of arrival and the expected time of arrival for that DLPDU. This delta-t (Δt) is communicated in every ACK reply DLPDU sent to the source device. Thus, every acknowledged transaction measures the alignment of network time between the devices.

Within the neighbor list, selected neighbors, specified by the network manager, are used as time synchronization sources. When a DLPDU from a time synchronization neighbor is received, the network time of the receiving device is adjusted. Time synchronization is based on either the DLPDU arrival time or the delta-t in the ACK DLPDU, depending on which device initiated the transaction.

In addition, device designers should understand the time drift characteristics of their products. When the device's time source drifts, the device transmits Keep-alive DLPDU, as needed, to its time-synch neighbors to maintain time synchronization. Device should be built such that it does not require a Keep-alive DLPDU more often than once per 30 s while temperature is varying 2 °C per minute or less. Furthermore, device designs should tolerate one retry in case of packet loss (i.e. a 10 s safety margin). This corresponds to approximately a compensated clock accuracy of 10 parts per million or better.

6.1.2 Network layer operation

6.1.2.1 General

The Type 20 network is a sensor and actuator mesh communication system that simplifies network and device installation and allows the end user to tailor the installation and its topology to satisfy specific application requirements. The basic elements of this network include field devices, gateway and network manager. Every network includes at least one gateway. While redundant network managers are supported, there is only one active network manager per network. Gateways may include one or more access points.

6.1.2.2 Mesh network

Figure 14 shows an example of network with the field devices deployed in a mesh topology. In this example network, there is one gateway, one network manager and thirteen field devices (labeled a to n). All communication occurs by moving packets from the gateway, through the intermediate devices, to the packet's destination field device. Each movement of a packet from one device to another along the route to the packet's final destination is called a hop.

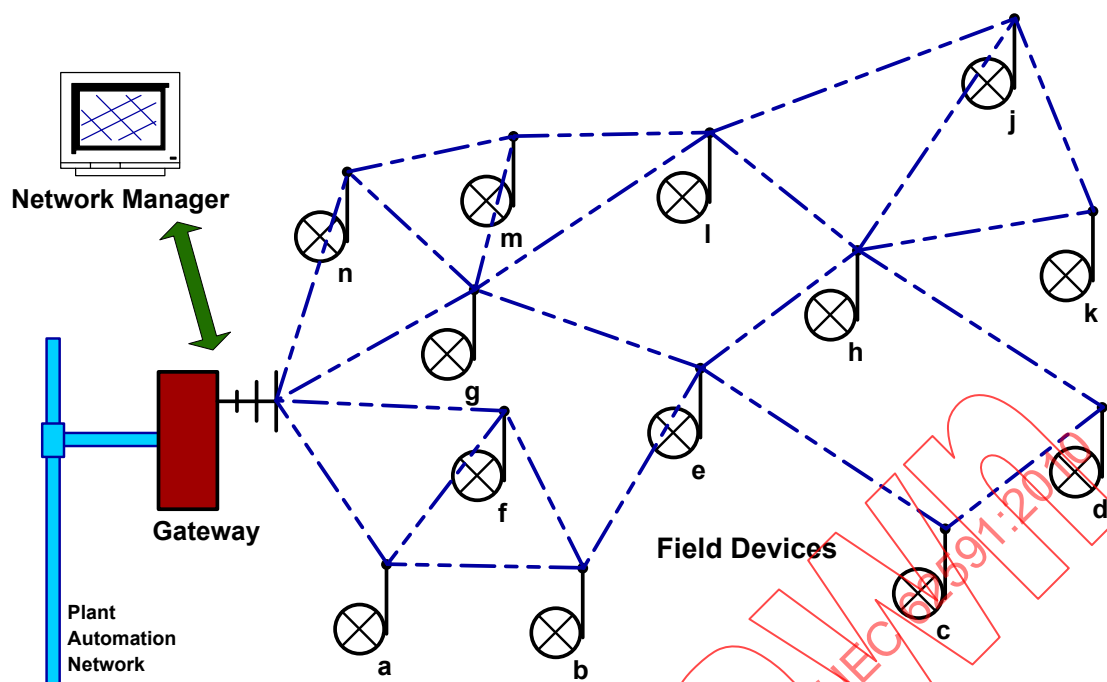


Figure 14 – Wireless network

All devices are able to source and sink packets and are capable of routing packets received from other devices in the network. The routing of packets from their initial source to their final destination may take several hops. The routing of packets is the responsibility of the network layer.

Within this network, nodes 'a', 'f', 'g', and 'n' are one hop from the gateway. Since these devices can pass packets directly to the gateway, communication with these devices has the lowest latency. However, since this network uses mesh topology, redundant links are included to improve system reliability by allowing packets to be routed around interference. In Figure 14, node 'a' can communicate directly to the gateway but can also communicate via node f to the gateway, if the direct route becomes blocked.

Nodes 'c', 'd', 'j' and 'k' are several hops away from the gateway. All intermediate devices (for example field devices 'g' and 'e') are capable of receiving and forwarding packets to and from these devices. Since the packets shall make several hops to and from nodes 'c', 'd', 'j' and 'k', the communication latency to these devices is larger than the devices that are one hop away. However, the mesh topology allows the physical size of the network to become larger and allows communication around obstructions. Furthermore the number of the devices that can be supported in a single mesh network can be larger than that can be supported by single star network.

The protocol for mesh topology can also be used in a star topology i.e. all devices are only one hop from the gateway to support a high performance application. It can also be used in a multi-hop overly-connected mesh topology for a less demanding (for example monitoring) application, or any topology in between. This protocol is flexible enough that a variety of applications (both high and low performance) can operate in the same network.

The wireless network maintains very high reliability using several mechanisms including multiple paths to network devices, multiple frequencies, and retries. If improved reliability is required, more paths can be inserted by adding additional field devices. The gateway uses access point to connect with the network and it can have more than one access points (see Annex D). Additional network access points can be used to increase throughput and reduce latency.

6.1.3 Device procedures

Several device level procedures are specified in Annex B.

6.2 Logical link control

6.2.1 General DLPDU structure

6.2.1.1 Overview

Figure 15 shows the common DLPDU structure. The DLE shall provide the octets to the physical layer in the same sequence as shown in this figure. The most significant octet of a multi-octet field shall be transmitted first, except as specified otherwise.

Each DLPDU shall consist of the fields in a sequence described below:

- A single octet set to 0x41 (That means data mode enabled according to IEEE 802.15.4-2006, Table 79 and security enabled according to IEEE 802.15.4-2006, 7.2.1.1.2),
- A 1-octet address specifier,
- The 1-octet Sequence Number,
- The 2-octet network_ID,
- Destination and Source Addresses either of which can be 2 or 8-octets long,
- A 1-octet DLPDU specifier,
- The DLL payload,
- A 4-octet keyed Message Integrity Code (MIC), and
- A 2-octet ITU-T CRC16.

0x41	Address specifier	Sequence number	Network_ID	Destination address	Source address	DLPDU specifier	DLL payload	MIC	CRC

Figure 15 – DLPDU Structure

6.2.1.2 DLPDU structure fields

6.2.1.2.1 Address specifier

This field specifies the length and type of destination and source addresses as shown in Figure 16. The bit-6 specifies the source address type; the bit-2 specifies the destination address type. If the value of any of these two bits is '1', then the address in the DLPDU, corresponding to that bit is EUI-64, else it is nickname (see 6.2.1.2.4). Any combination of address lengths can be used in a DLPDU.

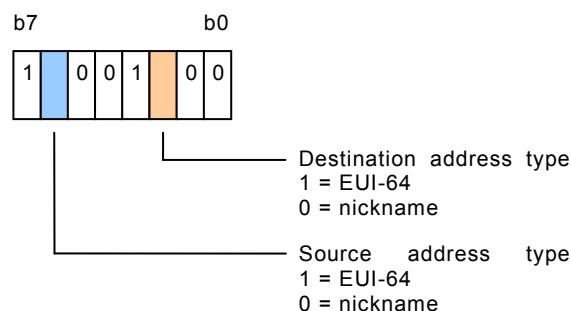


Figure 16 – Address specifier

The other bits in this field shall be set as indicated in the Figure 16.

6.2.1.2.2 Sequence number

The Sequence Number shall be set equal to the least significant octet of the Absolute Slot Number (ASN).

6.2.1.2.3 Network_ID

All networks are identified using a 2-octet number. A device connected to one network shall not send a PDU directly to another device connected to a different network.

This 2-octet value is transmitted, least significant octet (LSB) first, in all DLPDUs. If the network_ID does not match that of the network the device is a member of, then the packet shall be discarded. The ranges of network_ID values and their application are shown in Table 26.

Unless specifically noted the most significant octet of the DLPDU shall be transmitted first. The network_ID shall be transmitted with the least significant octet first.

Table 26 – Network_ID Allocation

Network_ID range	Application of network_ID
00 000-32 767 (0x0000-0x7FFF)	Permanent user defined networks (critical networks)
32 768-36 863 (0x8000-0x8FFF)	Temporary user defined networks (demos, trade shows, field trials, etc.)
36 864-57 343 (0x9000-0xDFFF)	Reserved
57 344-61 439 (0xE000-0xEFFF)	Manufacturing networks (non-public used by device manufacturers)
61 440-65 535 (0xF000-0xFFFF)	Reserved

6.2.1.2.4 Destination and source addresses

The DLL supports two types of addresses: a 2-octet nickname and an 8-octet IEEE EUI-64 address.

The 2-octet nickname is an address assigned to a specific network device (see 3.3.67). It needs to be unique only within the network to which the device is connected. The value 0xFFFF is reserved as broadcast address.

The EUI-64 shall be a concatenation of the 3-octet Organizationally unique identifier (OUI) value as defined by IEEE 802, and a 5-octet Unique ID (see 3.3.91) as shown in Figure 17. The Unique ID shall be a concatenation of the 2-octet Expanded Device type code and the 3-octet Device ID.

NOTE 1 No two devices compliant to this standard have identical EUI-64 address.

NOTE 2 The value of OUI is given by the IEEE Registration Authority Committee – see <http://standards.ieee.org/regauth/tutorials.html>.

The Expanded Device Type code shall be uniquely assigned for devices within a network and devices in range of other networks. Each device manufactured with the identical Expanded Device type code shall be assigned a Device ID that is unique among all instances of that type of device.

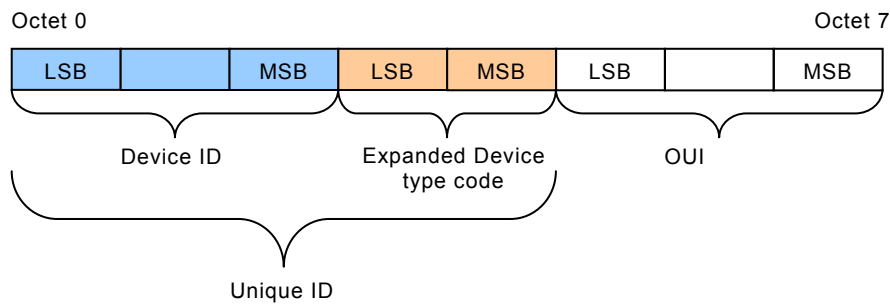


Figure 17 – Construction of 8-octet EUI-64 Address

IEEE STD 802.15.4-2006 requires multi-octet fields to be transmitted LSB first. Therefore, the EUI-64 address in the DLPDU shall be transmitted in the sequence shown in Figure 17 i.e. starting with the LSB of the Device ID and ending with the MSB of the OUI. The nickname shall also be transmitted starting with the LSB.

6.2.1.2.5 DLPDU specifier

This field is shown in Figure 18. The most significant two bits are reserved for future use. The DLE shall set the value of these bits to '00' before transmission. The DLE shall ignore any value of these two bits.

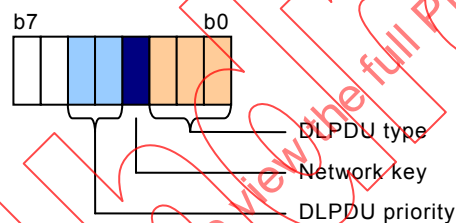


Figure 18 – DLPDU specifier

The bits 5 and 4 indicate the priority of the DLSDU contained in the DLPDU. Its encoding is:

- 00: Alarm (lowest) priority;
- 01: Normal priority;
- 10: Process data;
- 11: Command (highest) priority.

Bit 3 indicates the type of network key being used in the DLPDU. Its encoding is:

- 0: the well known key from this specification, used only by the unauthenticated device;
- 1: the confidential network-key, used by the authenticated device.

It indicates the key being used to authenticate the DLPDU. For all communications within the network between authenticated devices, The DLE shall set this bit to '1' and use the confidential network-key when generating the DLPDU MIC (see 6.2.4.2). This bit shall be reset to '0' during the joining process while the device is unauthenticated. DLPDUs passed between the unauthenticated device and its neighbors shall use the well known key (see 6.2.4.2.3).

The bits 2, 1 and 0 of the DLPDU specifier indicate the DLPDU type. Its encoding is:

- 000: ACK;
- 001: Advertise;
- 010: Keep-alive;
- 011: Disconnect;
- 111: Data.

The ACK, Advertise, Keep-alive and Disconnect DLPDUs shall use Command (highest) priority.

Other encodings of this field are reserved for future use.

6.2.1.2.6 DLL payload

The DLL payload depends on the DLPDU type. Data DLPDUs contain a DLSDU. Other types of DLPDU have DLE maintenance data that depends upon the type of DLPDU. For example ACK DLPDU has a payload that consists of a time adjustment as measured by the destination device specified in the immediately prior DLPDU.

6.2.1.2.7 Keyed message integrity code

A keyed message integrity code (MIC) is used for link-layer authentication of the received DLPDU. The receiving devices shall reply only to unicast, non-acknowledgement DLPDU that has been successfully authenticated.

6.2.1.2.8 Cyclic Redundancy Check

The Cyclic Redundancy Check (CRC) field is based on the 16 bit ITU-T CRC polynomial, also known as a CRC16. The CRC is calculated over the entire frame using the following generator polynomial of degree 16:

$$G_{16}(x) = x^{16} + x^{12} + x^5 + 1.$$

For more information see IEEE 802.15.4-2006, 7.2.1.9. The CRC should be calculated in the device hardware.

NOTE At receivers, the received octet stream is subjected to a similar calculation. When a receiver may predict the octets that will contain a CRC, it may omit those octets from the calculation and check that the received CRC matches those octets. Alternatively, the receiver may compute a CRC over all the octets of the message and check that the result matches an expected residual value of zero.

The CRC is used to detect bit errors and devices shall reply only to non-acknowledgement DLPDU with a CRC matching that calculated by the receiving device, or whose residual value after processing the entire DLPDU, including received CRC octets, matches the expected residual of zero.

6.2.2 DLPDU-specific structure, encoding and elements of procedure

6.2.2.1 General

There are five DLPDU types.

- Data DLPDU – contains DLSDU in transit to their final destination device. The source and sink for Data DLPDUs is the network layer.
- ACK DLPDU – is the immediate link level response to receipt of the source device's transmission DLPDU.
- Keep-alive DLPDU – facilitates connection maintenance between neighboring devices.

- Advertise DLPDU – provides information to neighboring devices trying to join the network.
- Disconnect DLPDU – are used to advise neighboring devices that the device is leaving the network.
- Devices receiving a DLPDU with any type other than the above listed types shall discard it and shall not acknowledge the received DLPDU.
- ACK, Advertise, Keep-Alive and Disconnect DLPDUs are generated and consumed by peer DLEs. These DLPDUs are not propagated to the network layer or onward through the network.

6.2.2.2 Data DLPDU

6.2.2.2.1 Purpose of Data DLPDU

The Data DLPDU is used to send DLS-user data from the source DLE to the destination DLE. It is also used by network layer to forward its user data from an intermediate DLE to another DLE.

6.2.2.2.2 Contents of the Data DLPDU payload

The DLL payload in this DLPDU originates from the network layer of this hop's source device and is passed to the destination device's network layer. It is then forwarded by the peer network layer to its final destination.

6.2.2.3 ACK DLPDU

6.2.2.3.1 Purpose of ACK DLPDU

It is used to send an immediate response to the just received non-ACK DLPDU.

6.2.2.3.2 Contents of the ACK DLPDU payload

The structure of the DLL payload for ACK DLPDU is shown in Table 27.

Table 27 – Contents of the ACK DLPDU payload

Offset from the start of the payload	Length in octets	Description
0	1	Response-code
1	2	Time-adjustment in μ s. The value of the Time adjustment is positive (negative) if the DLPDU was received earlier (later) than expected.

The Response-code shall specify either the successful reception or the error in reception of the immediately prior DLPDU as shown in Table 28.

Table 28 –ACK response-code

Code	Class	Description
0	Success	DLPDU was received without error and was accepted by destination device.
61	Error	No buffer was available to store the DLPDU of priority other than 'Alarm'; DLPDU was discarded.
62	Error	No buffer was available to store the DLPDU of 'alarm' priority; DLPDU was discarded.
63	Error	Priority of the received DLPDU was lower than Priority_threshold (see 6.4.2); DLPDU was discarded.

The Time-adjustment is the difference between the expected time of reception of the complete start delimiter that frames a non-ACK DLPDU and the actual reception of that complete start delimiter, measured in μ s. The Time-adjustment is a 2-octet, two's-complement integer.

6.2.2.3.3 Contents of the ACK MIC

The MIC field of ACK DLPDU shall be computed using the same key as the key in the just received DLPDU. See 6.2.4.2 for generating MIC.

6.2.2.3.4 Sending the ACK DLPDU

The ACK DLPDU shall be sent on the link when the sending DLE has just received any DLPDU except ACK DLPDU and there was no reception error and the destination address in the just received DLPDU is not a broadcast address. The DLE shall check the priority of the received DLPDU and availability of the local storage buffer and either accept or discard the received DLPDU as specified in 6.2.3.

6.2.2.3.5 Receiving the ACK DLPDU

The received DLPDU shall be processed as specified in 6.3.5.2.4.

6.2.2.4 Keep-alive DLPDU

6.2.2.4.1 Purpose of keep-alive DLPDU

Keep-alive DLPDUs is used:

- For network time synchronization. Time synchronization is updated based on the Time adjustment value returned in the ACK DLPDU;
- To confirm connectivity with a neighbor;
- To allow the device to be detected by others.

6.2.2.4.2 Contents of the keep-alive DLPDU payload

This field shall be null.

6.2.2.4.3 Sending the keep-alive DLPDU

This DLPDU is sent as specified in 6.3.3.4.

6.2.2.4.4 Receiving the keep-alive DLPDU

The received DLPDU shall be used to update the neighbor table as specified in 6.3.3.4.

6.2.2.5 Advertise DLPDU

6.2.2.5.1 Purpose of Advertise DLPDU

Advertise DLPDU is used to invite new devices into the network.

6.2.2.5.2 Contents of Destination Addresses

The destination address shall be the broadcast address. The broadcast address shall have be five octets long and shall have all zeros as the value.

6.2.2.5.3 Contents of the Advertise DLPDU payload

The DLL payload includes network information. The structure of the DLL payload for Advertise DLPDU is shown in Table 29.

Table 29 – Contents of the Advertise DLPDU payload

Offset from the start of the payload	Length in octets	Description
0	5	Absolute slot number – the number of slots since the start of the network to the slot used for transmission of this DLPDU.
5	1	Join-control
6	1	The Size of channel map array in number of bits. Maximum size of the channel map array is 256 bits.
7	variable	Channel-map-array – this is an array of bits starting with the least significant bit (bit 0 of octet 0) and adding octets as necessary until all bits are accounted for. Each bit corresponds to one channel. If the bit is set to '1', then the corresponding channel is in use.
n+1	2	Graph_ID
n+3	1	Number of superframes

The Join-control field specifies security level and join-priority as shown below:

Bits 7 – 4: Security level supported by the network;

Bits 3 – 0: Join_priority – an unsigned integer indicating the ability of the advertising device to support another child device. The lower the value the better this advertising device is for joining.

The content of Table 30, Table 31, and Table 32 are repeated for each superframe.

Table 30 – Beginning of the superframe

Offset from the start of the payload	Length in octets	Description
Variable	1	Superframe_ID
Variable	2	Superframe size i.e. the number of slots in this superframe.

Table 31 – The next field after the last superframe size

Offset from the start of the payload	Length in octets	Description
Variable	1	Number of Links

Table 32 – Repeated for each link

Offset from the start of the payload	Length in octets	Description
Variable	2	Join-slot – the specific slot within the superframe for this link
Variable	1	Join-slot-channel-offset

The Join-slot-channel-offset field specifies:

Bit 7: Reserved for future use; the sending DLE shall set it to '0';

Bit 6: If the link is for DLPDU transmission by the joining device, then the sending DLE shall set it to '1';

Bits 5 – 0: Channel offset – the frequency channel offset for this slot. This value is used to calculate the link frequency/channel number.

The size of Channel-map-array depends on the underlying physical layer. The IEEE 802.15.4-2006, 2 450 MHz physical layer provides 16 channels. The channel-map-array is two octets long. The least significant octet contains channels 11 - 18 and the most-significant octet contains channels 19-25 as shown in Table 33. The channel 26 is not used and thus bit 15 is always '0'.

Table 33 – Channel and frequency assignments

Index	Channel number
0	11
1	12
2	13
3	14
4	15
5	16
6	17
7	18
8	19
9	20
10	21
11	22
12	23
13	24
14	25
15	26

The absolute slot number and the channel map allows the current channel offset to be identified when issuing a packet to petition the network manager for admission to the network.

6.2.2.5.4 Contents of the Advertise MIC

The MIC field of Advertise DLPDU shall be computed using the well-known key. See 6.2.4.2 for generating MIC.

6.2.2.5.5 Sending the Advertise DLPDU

An Advertise packet can be sent on any transmit link that is available for transmission by the advertising device. On the joining device, when the DL-management requests that the DLE join a network, the DLE listens for the Advertise DLPDUs sent by other DLEs. It then uses the information in the received DLPDUs to synchronize with the network and to initiate the join process by sending its Join Request DLPDU. The joining device is limited to these links until it is authenticated by the network manager, and has received its network key. By limiting communication to only the join links, the network manager can quarantine the device until it is ready for the device to fully participate in the network.

All other DLPDUs have a higher priority than Advertise DLPDU.

6.2.2.5.6 Receiving the Advertise DLPDU

The DLE that has not joined the network shall receive all Advertise DLPDUs. It shall learn the network parameters included as the DLL payload of Advertise DLPDU. The reception of this DLPDU shall generate DL-Advertise indication primitive to the DLM-user.

6.2.2.6 Disconnect DLPDU

6.2.2.6.1 Purpose of Disconnect DLPDU

Disconnect DLPDU is used by a device that is leaving the network. It is sent to every neighbor known to the device.

6.2.2.6.2 Contents of Destination Addresses

The destination address shall be the neighbor's address.

6.2.2.6.3 Contents of the Disconnect DLPDU payload

This field shall be null, which means no octets shall be used in the payload.

6.2.2.6.4 Contents of the Disconnect MIC

The MIC field of ACK DLPDU shall be computed using the network key. See 6.2.4.2 for generating MIC.

6.2.2.6.5 Sending the Disconnect DLPDU

When the DL-management requests that the DLE leave the network, it shall send Disconnect DLPDU at the first opportunity.

6.2.2.6.6 Receiving the Disconnect DLPDU

The receiving DLE shall remove the sending DLE from its neighbor list, and shall delete all links connecting to the sending DLE. The reception of this DLPDU shall generate DL-DISCONNECT indication primitive to the DLM-user.

6.2.3 DLPDU Priority and Flow Control

The DLPDU priority is used for flow control, to mitigate the network congestion and ensure that the network manager retains control of the network during a process upset or when an adverse RF event occurs. For example the network manager can raise the Priority_threshold to reduce packet flow through the device by changing the value of Priority_threshold (see 6.4.2).

The DLE is required have at least one storage buffer reserved to receive "Alarm" priority DLPDU and at least one storage buffer reserved to receive "Command" priority DLPDU and these reserved buffers shall not be used to store the received DLPDU of any other priority. The receiving DLE shall check the priority of the received DLPDU, the current Priority_threshold and availability of the local storage buffer to determine whether the DLPDU is accepted or discarded as follows.

- a) If the final destination of the just received DLPDU is not the DLE itself and the priority level of the received DLPDU is lower than the Priority_threshold, then the DLE shall discard the received DLPDU and the response code in the ACK DLPDU shall be set to 63.

NOTE 1 To manage DLPDU flow through the network, the network manger can raise or lower the priority level of DLPDUs that a DLE is allowed to accept.

Else if the priority level is "Alarm" then

- i) if one and only one buffer dedicated to store this priority level is not available, then the DLE shall discard the received DLPDU and response code in the ACK DLPDU shall be set to 62;

NOTE 2 Only one buffer for "Alarm" priority is allowed to reduce the flood of alarms. The alarms are not lost, because these are time stamped and the PDU containing alarm information is sent again after a timeout.

- ii) Else the DLE shall accept the received DLPDU and response code in the ACK DLPDU shall be set to Success.

Else the priority level is not "Alarm" and

- i) if a buffer dedicated to store priority level of the received DLPDU is not available, then the DLE shall discard the received DLPDU and response code in the ACK DLPDU shall be set to 61;
- ii) Else the DLE shall accept the received DLPDU and response code in the ACK DLPDU shall be set to "Success".
- iii) If the received DLPDU is Keep-alive or Advertise or Disconnect, then the DLE shall always accept it and response code in the ACK DLPDU shall be set to "Success".

NOTE 3 If a storage buffer is available, the DLE always accepts the "Command" priority DLPDU, because it is the highest priority. The DLE should allocate enough buffers, so that it can receive almost all of the received "Command" priority DLPDUs.

In summary, network management DLPDUs always propagate through the network allowing the network manager to keep the network operational. Alarms flow through the network is restricted ensuring alarm floods do not disrupt network operation. Since alarms are always time-stamped, no information regarding failure sequences is lost.

All other network traffic flows through the network as buffer space and network capacity allows. Within this network traffic, process data has the highest priority. Operation and control of the process is lower priority to prevent network communication disruption.

6.2.4 Error detection and security

6.2.4.1 General

To perform error detection this standard includes a frame check sequence at the end of every DLPDU. The check (CRC) is used to detect communications errors. The CRC is calculated across the entire DLPDU using the 16-bit ITU-T algorithm.

For security, every DLPDU includes a MIC in every DLPDU.

6.2.4.2 MIC computation

6.2.4.2.1 General

A keyed MIC is used to ensure that the DLPDU is originated from an approved, authenticated device. The DLPDU itself is not enciphered; rather its contents are authenticated using the four-octet MIC. The MIC is generated and confirmed using CCM* mode (combined counter with CBC-MAC (corrected)) in conjunction with the AES-128 block cipher to provide authentication as specified in NIST Pub 800-38C. This cipher requires four octet-strings as parameters:

- 'a', the additional data to be authenticated but not enciphered;
- 'm', is the message to be enciphered;
- 'N', the 13-octet nonce; and
- 'K', the 128-bit AES Key.

The parts of a DLPDU, from the 0x41 octet through the end of the payload, is the octet-string 'a'. Since the DLPDU is not enciphered, the octet-string 'm' is empty.

6.2.4.2.2 DLL nonce

The 'N' shall be exactly 13 octets long and is the concatenation of the ASN and the source address in the DLPDU.

The ASN always contains the number of the current slot. The ASN is 5-octets long and is copied into N[0] to N[4], where N[0] is the most significant octet MSB of the ASN.

If the DLPDU has the EUI-64 address then it is copied (MSB to LSB) into N[5] through N[12]. In other words, the OUI is copied into N[5] through N[7]. The Expanded device type code and Device ID are copied (MSB to LSB) into N[8] through N[9] and N[10] through N[12], respectively.

If the DLPDU has the 2-octet nickname then it is copied (MSB to LSB) into N[11] through N[12]. The N[5] through N[10] are set to 0x00.

6.2.4.2.3 DLL key

The key shall be 128-bits (16 octets) long as required by the CCM Mode. It is copied into the 'K' octet-string in most significant octet first. The K[0] and K[15] are the most significant and least significant octet of the key, respectively.

There are two DLL keys:

- the well-known key, which is used in Advertise DLPDU, and
- the network key, which is used for all other DLPDUs.

The well-known key shall be identical for all devices and has a value of 7777 772E 6861 7274 636F 6D6D 2E6F 7267 hexadecimal. It is used for DLPDUs passed between the joining device and devices already part of the network.

NOTE 1 The well-known key is randomly chosen.

This network key is supplied by the network manager to a device, when it joins the network. The device shall not change the value of this key.

NOTE 2 The network manager may change the network key any time.

6.2.4.3 Error detection

When the DLE receives a DLPDU, it checks the CRC in the received DLPDU. If the CRC in the DLPDU does not match the expected CRC, then the DLE shall discard the received DLPDU and shall not send ACK DLPDU in response.

If the CRC in the received DLPDU is correct, then the DLE shall compute the expected MIC. It shall then compare it with the MIC in the received DLPDU. If these two are not identical, then the DLE shall discard the received DLPDU and shall not send ACK DLPDU in response.

6.3 Medium Access Control

6.3.1 Function

The Medium Access Control (MAC) sublayer provides basic communication from one DLE to another. The timing of such communication is regulated by slot synchronization across the network. To accomplish this, the DLE includes:

- Neighbor table, superframes, links, and graphs that configure the communication between the device and its neighbors (see 6.3.3.). These tables are prepared by the network manager. In addition the neighbor table is updated by the DLE, as it discovers addition of a new neighbor to the network or disconnection of an existing neighbor from the network.

- A link scheduler (see 6.3.4) that evaluates the device's tables and chooses the next slot to be serviced by listening for a DLPDU or by sending a DLPDU. In general the link scheduler walks the tables to identify the next slot in which to send a DLPDU and the next slot in which to listen. The slot scheduled is the next of these two slots to occur.
- State machines that control the propagation of DLPDUs through the MAC sublayer. MAC operation (see 6.3.5) consists of schedule maintenance and service slots. MAC operation is event driven and responds to service primitive invocations and the start of a slot needing servicing.

The MAC sublayer sends the DLPDU queued in the device's buffers and it receives DLPDU sent by a neighbor DLE. Both of these operations are performed one slot at a time. Successful communication depends on slot time synchronization between neighbors and the compliant timing of the transaction within the slot.

6.3.2 Slot timing

6.3.2.1 General

Figure 19 shows one slot and provides an overview of single transaction timings. The diagram (a) shows the operation at the source neighbor and the diagram (b) shows the operation at the destination neighbor. The details of slot time parameters are defined in Table 34; it also shows the values required for the physical layer supported by this standard and specified in IEEE 802.15.4-2006, 2 450 MHz physical layer. All transactions are required to comply with these timing parameters.

The slot start time at the source and destination can be different due to different amount of clock drift. This diagram shows that the slot starts at a later time at the destination device.

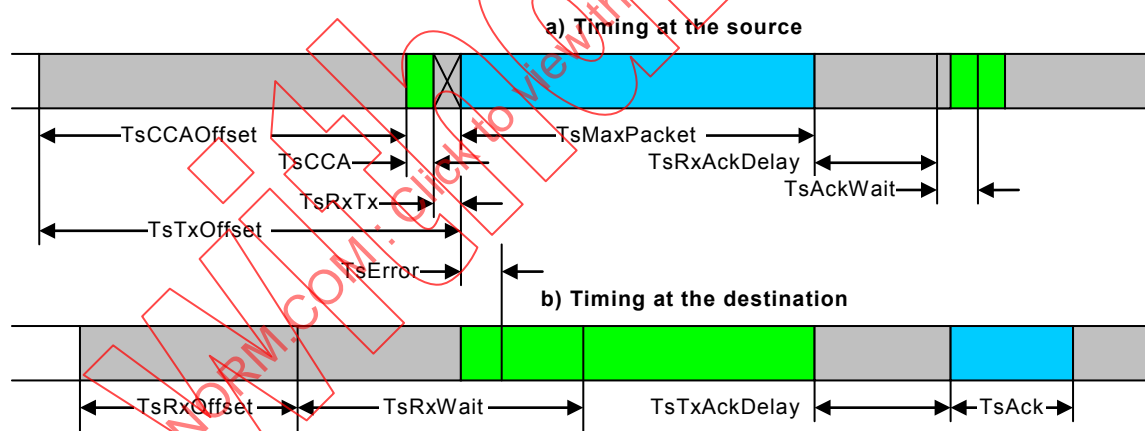


Figure 19 – Slot timing details

Table 34 – Slot timing definitions and values

Symbol	Definition	Required value
TsTxOffset	Start of the slot to start of preamble transmission	2 120 $\mu\text{s} \pm 100 \mu\text{s}$
TsRxOffset	Start of the slot to when transceiver shall be listening	1 120 $\mu\text{s} \pm 100 \mu\text{s}$
TsRxWait	The minimum time to wait for start of message; this correlates to the amount of drift between the neighbors that can be tolerated to maintain communications.	2 200 $\mu\text{s} \pm 100 \mu\text{s}$
TsMaxPacket	The amount of time it takes to transmit the longest possible packet that includes PhL preamble, delimiter, length and DLPDU	4 256 μs
TsTxAckDelay	End of message to start of ACK; the destination device shall validate the STX, and generate an ACK, if required, during this interval.	1 000 $\mu\text{s} \pm 100 \mu\text{s}$
TsRxAckDelay	End of transmit PhPDU to when transceiver shall be listening for ACK	800 $\mu\text{s} \pm 100 \mu\text{s}$
TsAckWait	The minimum time to wait for the start of an ACK	400 $\mu\text{s} \pm 100 \mu\text{s}$
TsAck	Time to transmit an ACK (26 octets)	832 μs
TsCCAOOffset	Start of slot to beginning of CCA	1 800 $\mu\text{s} \pm 100 \mu\text{s}$
TsCCA	Time to perform CCA (8 symbols)	128 μs
TsRxTx	The longer of the time it takes to switch from reception to transmission or vice versa (12 symbols)	192 μs
TsError	This is the difference between the actual start of message and the ideal start of message time as perceived by the receiving device. In other words, this is how much the receiving device perceives the transmitting device to be out of sync.	

If a device has a packet ready for transmission and it is scheduled to transmit in a slot, then that slot begins by allowing a time interval to prepare the PhPDU for transmission. The source shall perform the clear channel assessment (CCA) and transmit the packet. If the received DLPDU requires an acknowledgement, then the destination DLE shall transmit the PhPDU conveying ACK DLPDU.

If a device is scheduled to receive in a slot, then that device shall enter receive mode. The device shall start to listen to the medium after TsRxOffset delay from the start of the slot. The receive window specified by TsRxWait permits device time to drift, while still allowing devices to communicate and resynchronize their slot timers.

If the destination device detects a PhPDU, it shall capture the end of reception of the physical layer Delimiter start time. It shall calculate TsError as the difference between the expected start time and the actual start time of the physical layer Delimiter.

Time is allocated towards the end of the slot for processing the received packet and preparing for the next slot, (for example assessing and prioritizing the packets now queued up in each device). If one of the neighbors was the time source for the other, then the end of the slot time shall be aligned after successful communication.

6.3.2.2 Transmission of a packet

The source device shall wait for TsCCAOOffset duration after the start of a slot and then it shall perform CCA for TsCCA duration. If the channel is occupied, the transaction attempt is rescheduled for a later slot. Else, if the channel is clear, then the device shall switch the transceiver from reception to transmission. After that, the device shall begin its transmission such that the start of message (SOM) occurs exactly TsTxOffset duration from the start of the slot.

6.3.2.3 Reception of a packet

The destination device shall enter the receive mode at the start of the slot. It shall start to listen to the medium $TsRxOffset$ delay from the start of the slot. The destination device shall continue to check for SOM. If the SOM is not detected within $(TsRxOffset + TsRxWait)$ duration from the start of the slot, then it shall stop listening. Else, if the SOM is detected, then it shall receive and validate the received packet.

6.3.2.4 Acknowledged transaction

Most communications consist of the source device propagating a message by transmitting a packet and the destination device acknowledging the reception of that packet. For acknowledged communication the source and destination address in the DLPDU shall contain individual device address.

If the received packet is valid, then the destination device shall inspect the destination address in the DLPDU. If the destination address is not a broadcast address, and there are no reception error then the device shall switch from reception to transmission and shall transmit PhPDU containing ACK DLPDU, such that the SOM of this PhPDU occurs exactly $TsTxAckDelay$ after the end of the transmitted source device's packet.

Meanwhile the source device shall switch from transmission to reception. After $TsRxAckDelay$ duration from the end of its transmission, the source device shall listen to the medium. It shall check for SOM for the duration of $TsAckWait$.

6.3.2.5 Broadcast transaction

Broadcast transmissions are also supported. The DLPDU source address is individual device address and the destination address is the broadcast address. Broadcast messages are not acknowledged by the receiving device. The broadcast address shall be five octets long and shall have all zeros as the value.

If the received packet is valid, then the destination device shall inspect the destination address in the DLPDU. If the destination address is a broadcast address then the receiving device shall not transmit any response.

6.3.3 Communication tables and buffers

6.3.3.1 General

All devices shall maintain a series of tables that control the communications performed by the device and collect statistics on those communications. In addition, the device shall buffer the received packets, process these and forward, if required.

The tables controlling communication activities are:

- Superframe table – it stores the attributes of a superframe;
- Link table – these are used to store the attributes of a link;
- Neighbor table – it stores the list of all devices that the device may be able to directly communicate with;
- Graph table – it stores the next hop destinations legal for propagating the packet onward toward its destination

Some of the information in these tables is shared between the NLE and DLE. The relationship among these tables is shown in Figure 20. Within the device the neighbor table is central. This table contains a list of all devices that have been identified by the device.

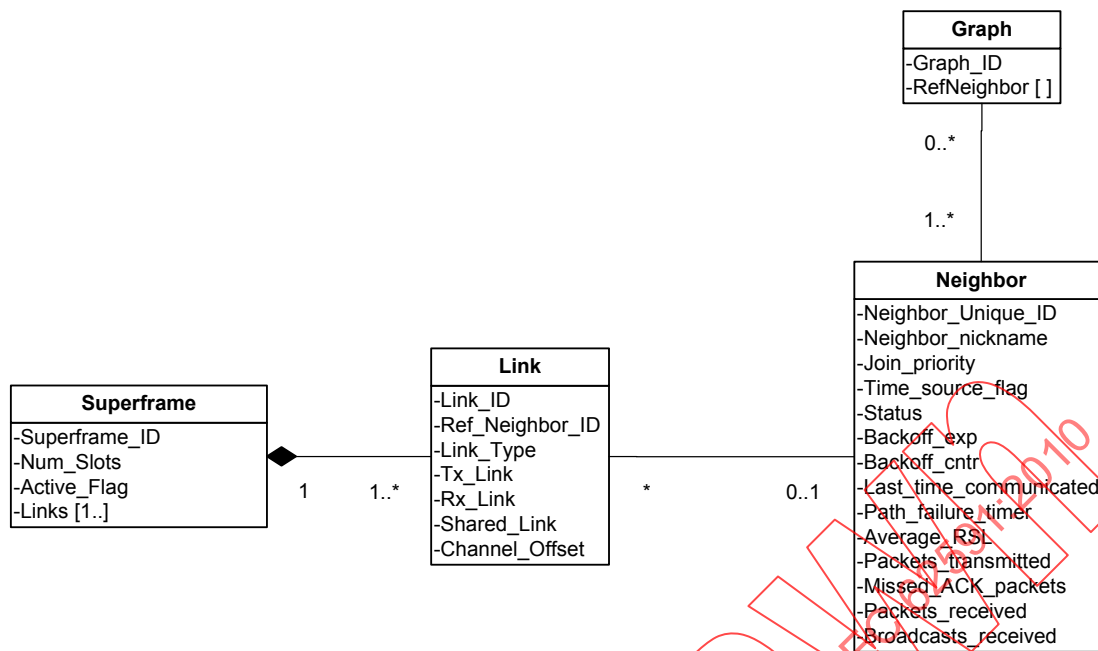


Figure 20 – DLE tables and relationship

A superframe contains 1 or more links. A link is associated with 0 (broadcast link) or 1 neighbor (specific network device). A neighbor can be associated with multiple links. Neighbors can also be associated with 0 or more graphs and graphs can be associated with 1 or more neighbors.

In addition to these tables, there are buffers to store the packets. Table 35 shows the minimum number of table entries and buffer space that every network device shall support.

Table 35 – Minimum DLE Table and buffer space requirement

Table name	Required minimum entries
Superframe	16
Link	64
Neighbor	32
Graph	32
Total Number of Graph-Neighbor Connections	128
Packet buffers	16

NOTE Some of the attributes of these tables described in this standard may be calculated or derived from other information, and it is not necessarily to store these separately. Although specific implementation of data and configuration storage is left up to the designers of the device, general description of the fields contained in the data structures is important to overall operation.

Each superframe has one or more links. The links specify the slot and associated information required to forward or accept a packet.

Each link defines a communications opportunity. A link can belong to one and only one superframe. Links specify a neighbor that is the partner in any communication using the link. A link is either directed to a neighbor or is a broadcast link. For a link, packets are either propagated to or from the neighbor. Broadcast links always propagate packets from the device.

A graph may specify more than one neighbor, any of which may be used for the next hop for packets following the route designated by the Graph_ID. When forwarding a packet using graph routing, the device can propagate it to any of the neighbors associated with that packet's Graph_ID. For more information see 6.3.3.5.

6.3.3.2 Superframe table

The superframe attributes are shown in Table 36. Superframe consists of a fixed number of slots, and initially contains no links and begins as inactive. Superframes are created and maintained by the network manager. Once a superframe is created the network manager can add, delete or modify links within the superframe, thus identifying opportunities for device to device communications. Once configured with links, the superframe can be enabled to allow the link scheduler to begin identifying transmit and receive slots.

Table 36 – Superframe attributes

Attribute name	Range of values	Definition
Superframe_ID	0 to 255	Unique identifier of the superframe, supplied by the network manager.
Num_Slots	0 to 65 535	Number of slots in this superframe i.e. size of superframe
Active_Flag	TRUE, FALSE	Flag indicating if the superframe is currently activated
Links [1..]	Link_ID	List of links in this superframe

6.3.3.3 Link table

6.3.3.3.1 General

The link attributes are shown in Table 37. The link includes a reference to neighbor that is permitted to communicate with the device. This reference can be to a single neighbor or it can be unspecified group of neighbors (for broadcast link). The slot number within the superframe, direction of the communication, link characteristics – shared or dedicated, and the initial communication channel are specified. The link is uniquely identified by the combination of following key parameters:

Superframe_ID, Slot number and neighbor address.

Table 37 – Link attributes

Attribute name	Range of values	Definition
Link_ID	See Note	Identifier of the link
Ref_Neighbor_ID	See Note	Reference to a neighbor table entry
Link_type	NORMAL, BROADCAST, JOIN, DISCOVERY	Indicates the type of link
Tx_Link	TRUE, FALSE	When TRUE, indicates the link shall be used for transmission
Rx_Link	TRUE, FALSE	When TRUE, indicates the link shall be used for reception
Shared_Link	TRUE, FALSE	When TRUE, indicates the link is shared by multiple devices
Slot	0 to 65 535	Slot number in superframe assigned to this link
Channel_Offset	11 – 25	Initial frequency hopping channel offset (see Table 33)
NOTE The range of values is not specified by this standard; it is implementation dependant.		

NOTE The network manager does not delete or suspend any link of JOIN type, while there is any outstanding request join request.

Links are allocated to a superframe by the network manager. When the network manager designates a link as being shared, contention-based, multiple-sender access is performed within the corresponding link. For these links, messages that are not acknowledged result in a random back-off algorithm being applied (see 6.3.4.4).

6.3.3.3.2 Link channel calculation

The link also specifies the Channel_Offset and thus implicitly the order in which the channels are hopped. The remainder of this operation indexes the channel table to obtain the actual channel used for communication in the active slot. The channel table contents and length depends upon the underlying physical layer. In the preferred physical layer specified in IEEE 802.15.4-2006 (2 450 MHz), the number of available channels is 16.

NOTE This standard uses only first 15 out of available 16 channels.

The DLE shall contain a 64-bit ChannelMap variable that tracks the device's active channels. Each bit (bit 0 through bit 63) that is TRUE identifies an active channel. The ChannelMap is initialized to all TRUE (i.e., all channels are active by default). Only bits corresponding to available channels for the physical layer are significant and considered in any calculations.

The DLE shall have a variable ActiveChannelArray to store the ordered set of active bit numbers in ChannelMap. The bit number of the least significant active channel shall be stored at index zero in the array. The bit number corresponds to the index into the physical channel table for the physical layer in use.

The DLE shall compute the value of the variable ActiveChannel by dividing the sum of the Channel_Offset and absolute slot number by the number of channels currently active.

$$\text{ActiveChannel} = (\text{Channel\_Offset} + \text{absolute slot number}) \% \text{number of active channels}.$$

It shall use ActiveChannel value to index into the ActiveChannelArray. The value stored at that index shall be used as the channel to be used for the link in that absolute slot number.

$$\text{Channel} = \text{ActiveChannelArray}[\text{ActiveChannel}].$$

6.3.3.4 Neighbor table

The device shall maintain a list of neighbors it has knowledge of. These are the companion devices that share a link with this device or neighbors whose communications have been overheard.

As shown in Figure 20, the neighbor table is central to driving device communications:

- Each link has a reference to one neighbor, or it is broadcast link.
- Graphs may have references to several neighbors. When a graph is used for routing, the neighbors for that graph are all valid recipients of the packet being propagated.

The neighbor table collects a variety of attributes and statistics pertaining to the neighbor as shown in Table 38 including:

- Basic neighbor identity information;
- Performance and historical statistics; and
- Shared slot attributes.

Basic identity information includes the neighbor's Unique ID, 2-octet nickname address and whether the device is a time source. To support contention-based access in shared slots, the

neighbor table also contains the parameters necessary to support the back-off algorithm (see 6.3.4.4).

Table 38 – Neighbor attributes and variables

Attribute name	Range of values	Definition
Neighbor_Unique_ID	see 6.2.1.2.4	Unique ID of the neighbor device (i.e. the long address)
Neighbor_nickname	see 6.2.1.2.4	The short address of the neighbor as defined in 3.3.67
Join_priority	0 to 15	Join_priority at which the neighbor advertises
Time_source_flag	TRUE, FALSE	Flag indicating if device should take time synchronization from this neighbor
Status	Bit Field	Status information regarding this neighbor (for example path failure) as specified in Table F.34
Backoff_exp	0 to 7	Back-off exponent in collision avoidance algorithm for shared links ^b
Backoff_cntr	0 to 127	Back-off counter in collision avoidance algorithm for shared links ^b
Last_time_communicated	0 to $2^{32}-1$ in units of ms	Time when last communicated with this neighbor
Path_failure_timer	0 to $2^{32}-1$	Cyclical timer in ms to detect failure of the path to the neighbor
Average_RSL	-128 to 127	Average received signal level in dBm for packets received from this neighbor
Packets_transmitted	0 to 65 535 ^a	Number of non-broadcast packets transmitted to this neighbor
Missed_ACK_packets	0 to 65 535 ^a	Number of packets for which an expected ACK was not received from this neighbor
Packets_received	0 to 65 535 ^a	Number of good non-broadcast packets received from this neighbor
Broadcasts_received	0 to 65 535 ^a	Number of good broadcast packets received from this neighbor
^a The value of packet counters rolls over from 65 535 to 0.		
^b See 6.3.4.4 for the algorithm.		

The device's ability to communicate with a neighbor is a key metric in forming and grooming the mesh network. The statistics are maintained for each neighbor. These include Average_RSL; statistics on the packets transmitted and received and the timestamp of the last communication with the neighbor. For linked neighbors, Average_RSL is calculated using a filter with the following equation:

$$\text{Average\_RSL} = \text{Average\_RSL} - (\text{Average\_RSL} / \text{RSL\_damp}) + (\text{MeasuredRSL} / \text{RSL\_damp})$$

where

MeasuredRSL is the RSL for the current packet;

RSL_damp is the damping factor.

RSL_damp shall be a power of 2 with a minimum value of 2 and a maximum value of 128. Its default value shall be 64. If the device does not receive any Data DLPDU from a neighbor, then this filter shall not be used for that neighbor. Instead, the highest value of RSL among all received DLPDU from the neighbor during its discovery shall be used.

If a link to a neighbor exists, the Last_time_communicated is used to trigger transmission of Keep-alive DLPDU. A Keep-alive DLPDU shall be transmitted to the neighbor, whenever the Last_time_communicated time value plus the Keep_alive_time (see Table 46) is greater than the current time. Keep-alive DLPDU transmission shall be repeated until a new DLPDU is received from that neighbor.

The Path_failure_timer shall also be maintained for each neighbor. Whenever a DLPDU from the neighbor is received, the timer shall be initialized to Path_fail_time (see Table 46). When this timer reaches zero, the DLM-event indication with PATH_FAILURE event shall be invoked. When this occurs, the timer shall be re-initialized to Path_fail_time and restarted. The device shall keep trying to use the failed path until the neighbor or its links are removed from the table.

Every received packet is used to update the corresponding neighbor attributes or it creates a new neighbor table. If a device shares links with a neighbor, then that neighbor shall be retained in the table. Neighbors without links to this device shall be deleted. When the neighbor table is full and a new neighbor is detected, the neighbor with the oldest value of Last_time_communicated shall be deleted and the new neighbor shall be added to the table.

6.3.3.5 Graph table

The graph provides the routing information to guide the delivery of a packet to its final destination. A graph is a directed list of paths that connect two devices within the network. Both upstream (toward the gateway) and downstream graphs are used. The network manager is responsible for correctly configuring each graph. Table 39 shows graph attributes. Graphs have an identifier; a list of neighbors and the destination's long and short address. When the Graph_ID value is less than 0x0100 it indicates a Superframe_ID, and when equal to 0xFFFF it indicates the graph is invalid.

Table 39 – Graph attributes

Attribute name	Range of values	Definition
Graph_ID	256 to 65 534	Unique graph identifier as defined in 3.3.44
RefNeighbor []	Neighbor_Unique_ID or Neighbor_nickname	List of references to neighbors that are the next hop toward the destination

At the original source device for a packet the upper layers identify the packet's final destination and the graph to use when routing the packet.

NOTE The device uses the route information to select the Graph_ID.

RefNeighbor identifies those devices that are permissible data-link destinations for the packet's next-hop toward its final destination. Since it is possible for a graph to specify multiple next hops, redundancy and reliability is built into graph routing. Individual neighbor references are also called a "connection".

The destination addresses are required by all devices sourcing data to a specific final destination. However, the addresses are optional since intermediate devices may be merely forwarding the packet along its route and not sourcing data to the same final destination.

6.3.3.6 Packet buffer list

6.3.3.6.1 General

All devices shall maintain a list of packet buffers. These buffers are used to receive, to process and to transmit packets. The record associated with a packet is shown in Table 40. The Packet_ID is used to reference the packet and is created when the packet is added via the DL-TRANSMIT request service primitive.

Table 40 – Packet record

Record member	Definition
Packet_ID	Unique packet identifier
Payload	The DLL payload i.e. the NPDU
Priority	Transmit priority of the packet
Destination	Graph, source or proxy routing information or broadcast destination
Packet_time_stamp	The ASN time when a packet was added to the transmit list

The Packet_time_stamp shall be set when the packet is added to the transmit list. This time stamp shall be used to select the packet to transmit when either of two equal priority packets can be propagated in the same absolute slot. In this case the older packet shall be sent first.

The record contains the packet's priority and the specification of the packet's destination.

6.3.3.6.2 Packet Priorities

Since the device shall be able to store multiple packets, pending their propagation, it is possible that multiple packets could be candidates for transmission in the same slot. When that happens, priority and the age of the packet shall be used to select the packet to be transmitted in that slot. When there are multiple packets that can be transmitted in the slot the highest priority packet shall be chosen for transmission (see 6.2.3). If multiple packets are tied for the highest priority then the oldest packet shall be transmitted first.

6.3.3.6.3 Destinations

The destination of a DLPDU can be specified as one of the following types:

- Graph route – the network layer specifies a Graph_ID as the destination. The DLE can send the packet to any of the devices associated with that graph.
- Source route – the network layer specifies a specific device address as the destination. The DLE shall transmit the packet on a link to that neighbor. If the device does not have a link to that neighbor then the network manager shall be notified that a source route error has occurred.
- Broadcast – the destination address is broadcast. The DLE shall transmitted using a broadcast link from the designated superframe.
- Proxy Route – the network layer indicates the destination address is that of a joining device. The DLE shall transmit the packet using a join link.

The destination type determines the Link_type that can be used and ultimately the slot the in which the DLPDU can be transmitted.

6.3.4 Link scheduling

6.3.4.1 General

All devices shall maintain a link schedule that identifies the next slot that shall be serviced. All communications are scheduled to occur within a superframe during specific slots in that superframe. Servicing a slot consists of either listening for a new packet or transmitting a packet. When a slot has both a packet waiting to be transmitted and it is also has a receive link, transmitting the packet shall have priority over attempting to listen for a new packet.

Preparing the link schedule consists of:

- to evaluate the packets pending transmission and to determine the first ASN that can be used to transmit a packet; and

- to determine the first ASN that can be used in attempting to receive a new packet.

The events external to DLE and internal events can affect the link schedule. Therefore, the DLE has to reassess the schedule whenever such event occurs. An example of the internal event is the failure of high priority transmission transaction. An example of the external event is that the network manager disables a superframe.

6.3.4.2 Servicing transmit link

Figure 21 summarizes the relationships that affect the calculation of the next transmit slot to be serviced. Packets received from the NLE are scheduled for a slot based on the graph, the destination address and whether or not the device is acting as a proxy for a device joining the network. Each packet has a destination and a priority. As specified in 6.3.3.6.3, there are several types of destinations, each of which affects slot selection. For each type of destination the set of transmit links shall be determined.

- For a graph-routed destination, the set of links shall be all transmit links for all neighbors in the graph.
- For a broadcast destination address, the set of links shall be all broadcast links for the designated superframe.
- For a single neighbor destination address, the set of links shall be all transmit links to that single neighbor.
- For a proxy destination address, the set of links shall be all transmit links of type Join.

Once the set of links suitable for transmitting the packet are determined, the set of upcoming slots shall be determined. The intersection of the upcoming slots and the links with a pending packet determines the next slot to be scheduled to propagating that packet.

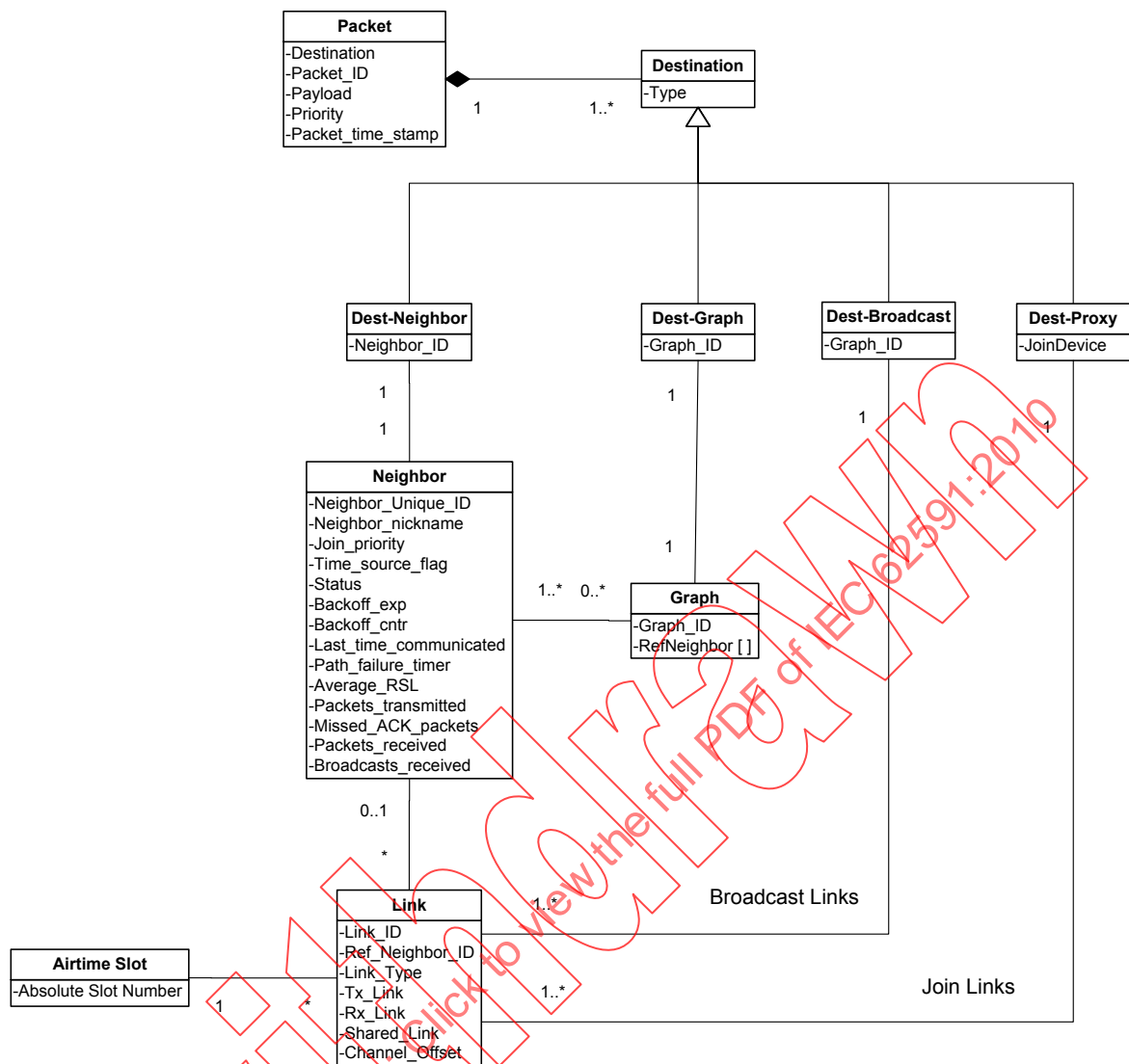


Figure 21 – Relationships used for link scheduling

The figure illustrates the static rules used at the DLL for determining what link will be used for transmission. A new object 'Airtime Slot' is introduced to model the fact that a fully specified link will be associated with an airtime slot.

A packet contains a destination. Destination can be further specialized by the destination type – a specific neighbor, a graph, a broadcast, or a proxy. If proxy address is being used then link shall be a join link. If the address is a broadcast address then a broadcast link shall be used. If the address is a specific neighbor, a specific link associated with a neighbor shall be used. If the destination is a graph then any neighbor on the graph may be used.

The ASN shall be used to track the occurrence of each slot. An ordered list of the links that are soon to occur within a superframe shall be identified using following equation:

$$\text{Superframe slot} = \text{ASN} \% (\text{Num\_Slots of superframe}).$$

NOTE Links may be shared (see 6.3.4.4), used to advertise (see 6.3.4.5) the device's presence to new devices attempting to join the network or used to discover neighbors (see 6.3.4.6)

The back-off algorithm in shared slots shall be employed to determine when a transmission in a shared slot is allowed. Advertise packets (see 6.3.4.5) are transmitted periodically as specified by the network manager.

Links that cannot be used to propagate a pending packet are ignored. From the resulting set of links, the first potential link and its associated slot can be identified. When there are multiple packets that can be propagated in that slot, the rules specified below shall be used.

- Find the packets with the highest priority;
- If there is only one such packet then select that packet for transmission;
- Else if there is more than one such packet, then find that packet destined to those neighbors with which there has not been any communication for the longest time;
- If there is only one such packet then select that packet for transmission;
- Else if there is more than one such packet, then if Keep_alive_time has expired with a neighbor, then prepare Keep-alive DLPDU for transmission to those neighbors with which there has not been any communication for the longest time;
- Else if an Advertise time has elapsed then prepare Advertise DLPDU for transmission.

6.3.4.3 Servicing receive link

For each active superframe, all receive links shall be scheduled. The set of upcoming receive links can be calculated in the same way as with transmit links (see 6.3.4.2). If there is no need to propagate a packet, then all receive links shall be serviced. Once the ordered list of links is created, the earliest slot is selected and the link within that slot with the lowest superframe identifier number shall be the candidate for servicing. If there are no other receive links to be serviced, then the device shall service the discovery receive link. In any case, this receive link will be serviced if there are no pending packets to be transmitted on or before this slot.

6.3.4.4 Shared slots

Shared slots are assigned to more than one source device. One or more of these devices may attempt to convey a packet within that slot and channel. Therefore, collisions may occur within a shared slot. If a collision occurs, the destination device will not be able to successfully receive any source's transmission and will not produce an acknowledgement to any of them. To reduce the probability of repeated collisions, source devices shall use random back-off delay when their transmission in a shared slot is not acknowledged (i.e. no ACK is received by the source device).

A device shall maintain two variables for each neighbor: Back-off exponent (Backoff_exp) and Back-off counter (Backoff_cntr). Both of these variables shall be initialized to 0. When a transaction in a shared slot fails, the random back-off period shall be calculated based on the Backoff_exp. For each unsuccessful attempt by the source device in a shared slot, the Backoff_exp shall be incremented and a sequential set of numbers calculated. The value of Backoff_exp shall not exceed that of Max_backoff_exp. The set of numbers consists of the whole numbers {0, 1, ... L} where

$$L = ((2 \text{ to the power } \text{Backoff\_exp}) - 1).$$

The Table 41 shows the back-off sets for Backoff_exp values of one (1) to seven (7).

Table 41 – Example Backoff_cntr selection sets

Backoff_exp	Set of possible values for Backoff_cntr
1	{0, 1}
2	{0, 1, 2, 3}
3	{0, 1, 2, 3, 4, 5, 6, 7}
4	{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}
5	{0, 1, 2, 3, 4, 5, ..., 28, 29, 30, 31}
6	{0, 1, 2, 3, 4, 5, ..., 60, 61, 62, 63}
7	{0, 1, 2, 3, 4, 5, ..., 124, 125, 126, 127}

From this set calculated based on the Backoff_exp, a random value for the Backoff_cntr shall be selected. For each subsequent shared link to that neighbor, the Backoff_cntr shall be decremented. When the corresponding Backoff_cntr is zero, then only the source device shall attempt a transmission in that shared slot.

Since it is also possible that interference can cause packet loss, the back-off exponent and counter shall be maintained separately for each neighbor. If communication with that neighbor fails on a dedicated link then the device shall assume channel degradation, rather than a collision as the cause of the failure in the shared slot and the Backoff_exp and the Backoff_cntr shall both be reset to zero.

Broadcast messages shall not be transmitted on shared slots.

6.3.4.5 Advertising

Nodes that are already part of the network may be configured by the network manager to advertise to the network and facilitate joining by new devices. The Advertise_time (see 6.4.2) attribute sets the interval at which Advertise packets are generated. Whenever the Advertise_time interval elapses, an Advertise packet shall be transmitted on the first available non-shared transmit link. When Advertise_time is set to zero then an Advertise packet shall be generated whenever a non-shared transmit link is available.

An Advertise packet may be sent on any non-shared transmit link that is not in use. All other traffic has higher priority than advertising.

6.3.4.6 Neighbor discovery

Devices shall continuously listen for communications from their existing neighbor and for communications from new neighbors. Continuous monitoring of neighbors and the discovery of new neighbors is critical to the maintenance of the mesh and the enhancement of communications reliability.

Upon receiving any DLPDU, from a new neighbor, the device shall invoke the DLM-event indication along with that neighbor's address and the RSL for that DLPDU. Periodically the NLE communicates the list of new neighbors to the network manager. The device may receive a DLPDU, addressed to the device, from a new device attempting to join the network. When this happens the device shall add the device to the neighbor table. Then the device shall duplicate the join links and insert the joining device as the link neighbor. These neighbors are perishable and may be deleted based on the Last_time_communicated to make room for new neighbors (see 6.3.3.4). When a neighbor is deleted, the auto-created join links connecting the device to the joining device shall be deleted.

To aid in the discovery of new neighbors, the device listens whenever possible on discovery links. Discovery links are shared by all devices in the network. In addition to listening, a

device shall also randomly transmit a Keep-alive DLPDU in a transmit discovery Link. This allows other devices listening on the discovery link, the opportunity to discover the device.

The slowest rate at which the device transmits on the discovery link is bounded by the `Discovery_time` (see 6.4.2). To schedule the discovery transmission, the device shall select a random time period between 0 and `Discovery_time` and use this to initialize the discovery timer. When that timer expires, the device shall set a `TimeToDiscover` flag to `TRUE` and schedule the next randomly timed discovery transmission. If the value of `Discovery_time` is '-1' (i.e., 0xFFFFFFFF), then the DLE shall not serve discovery link.

Once the `TimeToDiscover` flag is set, link scheduling will transmit the Keep-alive packet at the first available discovery transmit link. The Keep-alive packet shall be addressed to the linked neighbor with the oldest `Last_time_communicated` value.

6.3.5 MAC operation

6.3.5.1 General

This specification decomposes the MAC sublayer into three primary components as shown in Figure 22. The components are:

- The TDMA machine that specifies the overall operation of the MAC sublayer,
- The transmitter that sends a DLPDU – XMIT machine; and
- The receiver that listens to a link and receives a DLPDU – RECV machine.

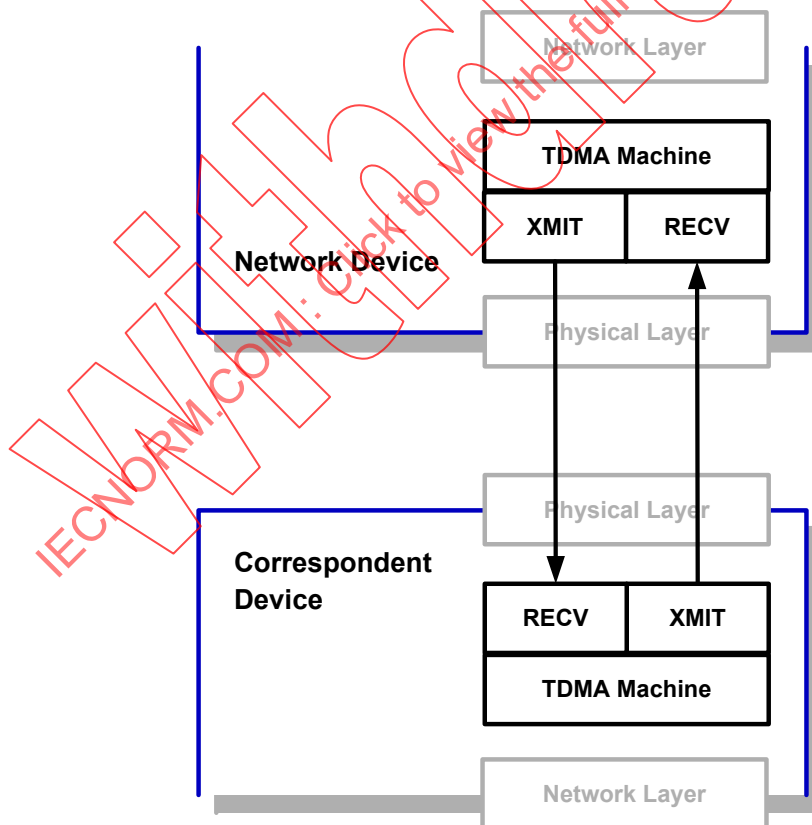


Figure 22 – MAC components

6.3.5.2 TDMA machine

6.3.5.2.1 General

The operation of the TDMA Machine is shown in Figure 23. The state transitions are shown in Table 42. Operation of the TDMA Machine begins in “Idle” state, when the device joins a network; is configured with a list of superframes, graphs and links; and begins receiving DLPDUs from other devices or transmission request from the NLE. Normal operation can be divided into four responsibilities:

- to manage schedule,
- to transmit DLPDUs to other devices,
- to receive DLPDUs from other devices, and
- to maintain time synchronization.

Managing schedule includes creation and maintenance of superframes, links, and neighbor statistics. As DLPDUs are acquired and propagated, the schedules shall be updated. Link scheduling consists of evaluating the active superframes, links, DLPDUs waiting for transmission and to identify the next slot that needs servicing.

The schedule determines the dispatching of DLPDUs. All receive links shall be serviced by attempting DLPDU reception. Since links shall be allocated to support possible retries, often there are more receive links than transmission links. Since transmission is often successful, many of the receive links will be unused, not containing a corresponding transmission. Most received DLPDUs contain DLPDUs destined for the NLE.

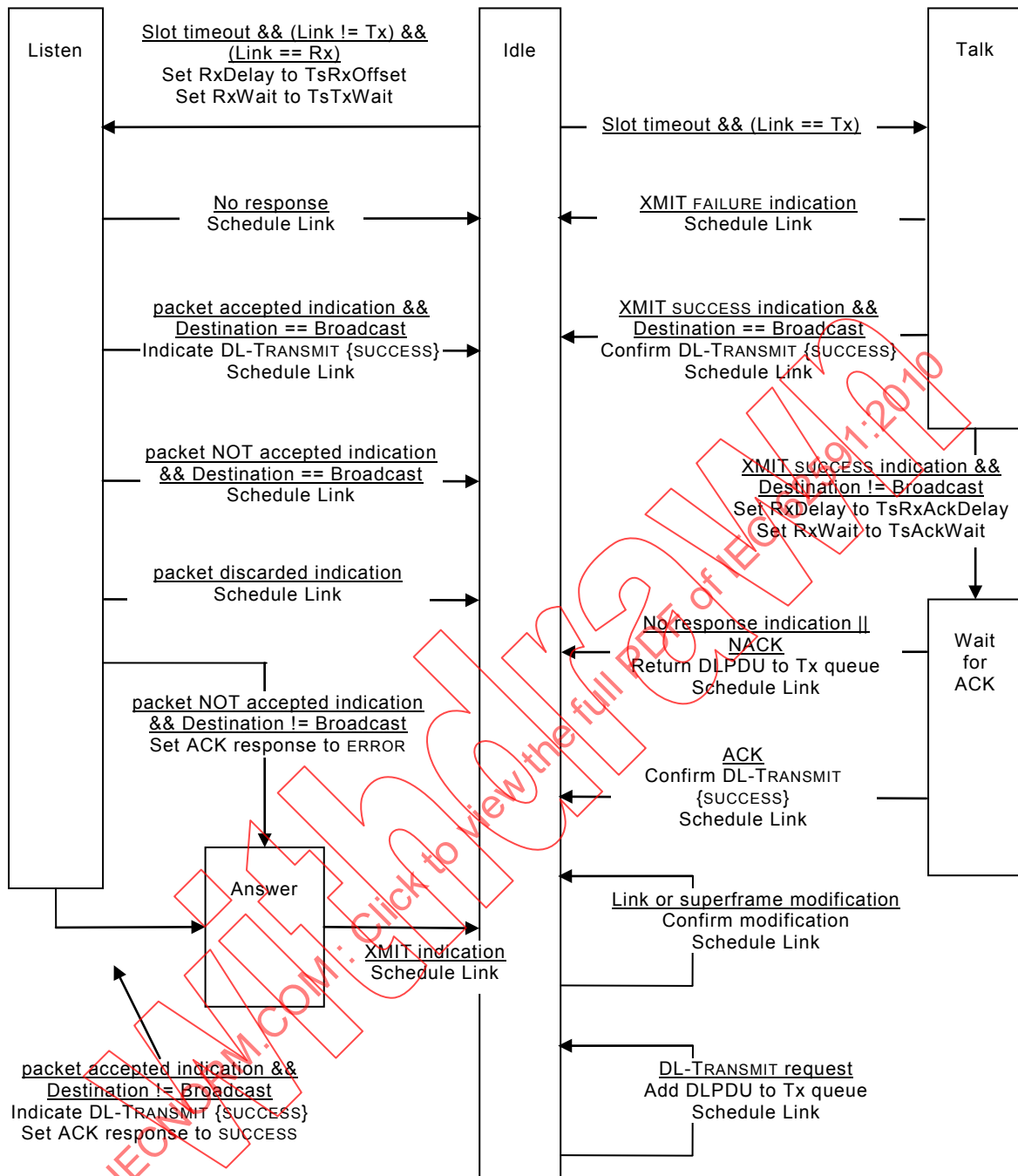


Figure 23 – TDMA state machine

Table 42 – TDMA state machine

Current state	Event or condition => action	Next state
Idle	Slot timeout && (Link == Tx)	Talk
Idle	Slot timeout && (Link != Tx) && (Link == Rx) => Set RxDelay to TsRxOffset; Set RxWait to TsTxWait	Listen
Idle	Link or superframe modification => Confirm modification; Schedule Link	Idle
Idle	DL-TRANSMIT request => Add DLPDU to Tx queue; Schedule Link	Idle
Talk	XMIT FAILURE indication => Schedule Link	Idle
Talk	XMIT SUCCESS indication && Destination == Broadcast => Confirm DL-TRANSMIT {SUCCESS}; Schedule Link	Idle
Talk	XMIT SUCCESS indication && Destination != Broadcast => Confirm DL-TRANSMIT {SUCCESS}; Schedule Link	Wait for ACK
Wait for ACK	No response indication NACK => Return DLPDU to Tx queue ; Schedule Link	Idle
Wait for ACK	ACK => Confirm DL-TRANSMIT {SUCCESS}; Schedule Link	Idle
Listen	No response => Schedule Link	Idle
Listen	packet accepted indication && Destination == Broadcast => Indicate DL-TRANSMIT {SUCCESS}; Schedule Link	Idle
Listen	packet NOT accepted indication && Destination == Broadcast => Schedule Link	Idle
Listen	packet discarded indication => Schedule Link	Idle
Listen	packet NOT accepted indication && Destination != Broadcast => Set ACK response to ERROR	Answer
Listen	packet accepted indication && Destination != Broadcast => Indicate DL-TRANSMIT {SUCCESS}; Set ACK response to SUCCESS	Answer
Answer	XMIT indication => Schedule Link	Idle

6.3.5.2.2 Idle state

The following events can occur while in the "Idle" state.

- a) Slot Timeout: the most frequent activity performed by the TDMA Machine is the servicing of a Slot-timeout event. This event indicates that a slot needs to be serviced.
 - 1) If the slot timed out for a transmission link and there is DLPDU waiting for transmission, then the TDMA shall change to "Talk" state.
 - 2) Else if it is receive link, then the TDMA shall change to "Listen" state.
- b) Modification to the device's list of superframes or links: it affects link scheduling. These changes typically result in link definitions being revised and may result in a different number of transmit and receive attempts per second.
- c) DL-Transmit request: this shall add a DLPDU to the device's Tx queue and it may affect link scheduling.

6.3.5.2.3 Talk state

In this state, the DLE shall attempt to transmit the DLPDU to its neighbor(s). The operation of this state is specified in 6.3.5.3. The attempt will result in either success or one of several negative outcomes.

- Successful propagation of a DLPDU with a destination address that is the broadcast address occurs as soon as the DLPDU is transmitted. The DLPDU buffer shall be released immediately upon completion of the DLPDU transmission and the link schedule shall be evaluated.
- Successful propagation of a DLPDU with a destination address that is not broadcast address occurs when a validated, successful ACK is received. This indicates that DLPDU propagation was completed successfully and the DLPDU buffer shall be released.
- If the ACK contains an error response-code then the destination neighbor has refused the DLPDU. When this occurs, the DLPDU shall be retained and its propagation retried.
- If no ACK response is received then the DLPDU shall be rescheduled and transmission retried. If the DLPDU times-out, the queued DLPDU is returned to the NLE for disposition and possible re-routing.
- If the transmission fails in the XMIT machine, then transaction shall be aborted and link schedule shall be evaluated.

Upon successful transmission and if the DLPDU destination is not broadcast, the TDMA machine shall transition to "Wait for ACK" state, which is described in 6.3.5.4. Upon entry into this state, the RxDelay timer shall be set to TsRxAckDelay and the RxWait timer (receive window) shall be set to TsAckWait.

6.3.5.2.4 Wait for ACK state

The TDMA Machine stays in this state until the RECV machine completes. If the RECV machine indicates there was "No Response" the transaction fails. If it was a shared link, the Backoff_exp and Backoff_cntr shall be re-evaluated to produce a random back-off period before the next transmission attempt in a shared slot (see 6.3.4.4). If it was not a shared link, the Backoff_exp and Backoff_cntr shall be reset to zero. Then, after link schedule evaluation, the TDMA Machine shall change to "Idle" state.

If an ACK containing the "Success" Response Code is received then communication was successful. However, if an error response-code was received, then the neighbor did not accept the DLPDU. In either case, network time synchronization may be assessed using the Time-adjustment field in the neighbor's response. If the neighbor is a time-source for the device, then the device shall resynchronize its network time using the Time-adjustment field. If the neighbor is not a time source then, no time correction shall be performed.

Receiving an ACK indicates the neighbor device has successfully received the message and accepted it. The DLPDU buffer containing the transmitted DLPDU shall be released. The TDMA Machine shall change to "Idle" and evaluate the link schedule.

6.3.5.2.5 Listen state

In this state, the DLE shall attempt to receive the DLPDU. The operation of this state is specified in 6.3.5.4. The reception of a DLPDU has three possible outcomes:

- the DLPDU's final destination is the DLE itself, then the DLPDU shall be retained for processing by the DLE;
- the DLPDU's final destination is not the DLE itself, then the DLPDU shall be forwarded by the DLE toward the message's final destination;
- the DLPDU is not addressed to the device, the DLPDU shall be discarded.

In all cases, when a DLPDU is received, the corresponding neighbor table entry shall be updated or if necessary the table entry shall be created.

The DLPDU reception cycle consists of an attempt to receive a DLPDU, the validation of any DLPDU received and, if the DLPDU destination address is not broadcast, then the transmission of a response to a valid received DLPDU.

When attempting to receive a DLPDU, the RxDelay timer shall be set to TsRxOffset and the RxWait timer (receive window) shall be set to TsTxWait. The TsTxWait time sets the duration of the receive window and represents the largest time drift between neighboring devices allowed by this standard.

If no DLPDU is received in "Listen" state, then the TDMA Machine shall re-evaluate the link schedule and return to the "Idle" state. If a DLPDU is received, then network time synchronization shall be assessed. TsError shall be calculated by taking the difference between the actual start time and the expected start time of the received DLPDU. If the neighbor sending the DLPDU is one of the time sources of the DLE, then the device shall use the measured TsError to resynchronize its network time. If the neighbor is not a time source then, no time correction shall be performed.

While in the "Listen" state and upon successful DLPDU reception the device shall determine whether to accept the DLPDU or discard it. The device accepts or discards the DLPDU based on the DLPDU priority, the current Priority_threshold (see 6.2.3) and the number of DLPDU buffers currently occupied in the device. If the DLPDU is accepted by the device, it is either consumed by the DLE itself or forwarded to the NLE using DL-TRANSMIT indication service and the DLE shall update the link schedule.

Once the data-link layer has determined DLPDU disposition (accepting or discarding the DLPDU), the TDMA machine shall exit "Listen" state as follows:

- If the DLPDU destination address is the broadcast address, then the TDMA machine shall change to the "Idle" state.
- If the destination address is not the broadcast address and the DLPDU was accepted, then the DLE shall transmit ACK DLPDU, with response-code value equal to "Success" to the sender of the just received DLPDU. The TDMA machine shall change to "Answer" state.
- If the destination address is not the broadcast address and the DLPDU was not accepted, then the DLE shall transmit ACK DLPDU, with response-code value equal to an error code to the sender of the just received DLPDU. The TDMA machine shall change to "Answer" state.

6.3.5.2.6 Answer state

The DLE shall copy the measured TsError into the Time-adjustment field of the ACK DLPDU. The TDMA machine operates as ACK transmission machine described in 6.3.5.5.

After transmission of the ACK is completed, the TDMA Machine shall evaluate the link schedule and change to the "Idle" state.

6.3.5.3 XMIT machine

6.3.5.3.1 General

Figure 24 shows the state machine that is used to transmit a DLPDU through either a dedicated or a shared slot. The transitions are shown in Table 43. This machine is started to transmit a DLPDU.

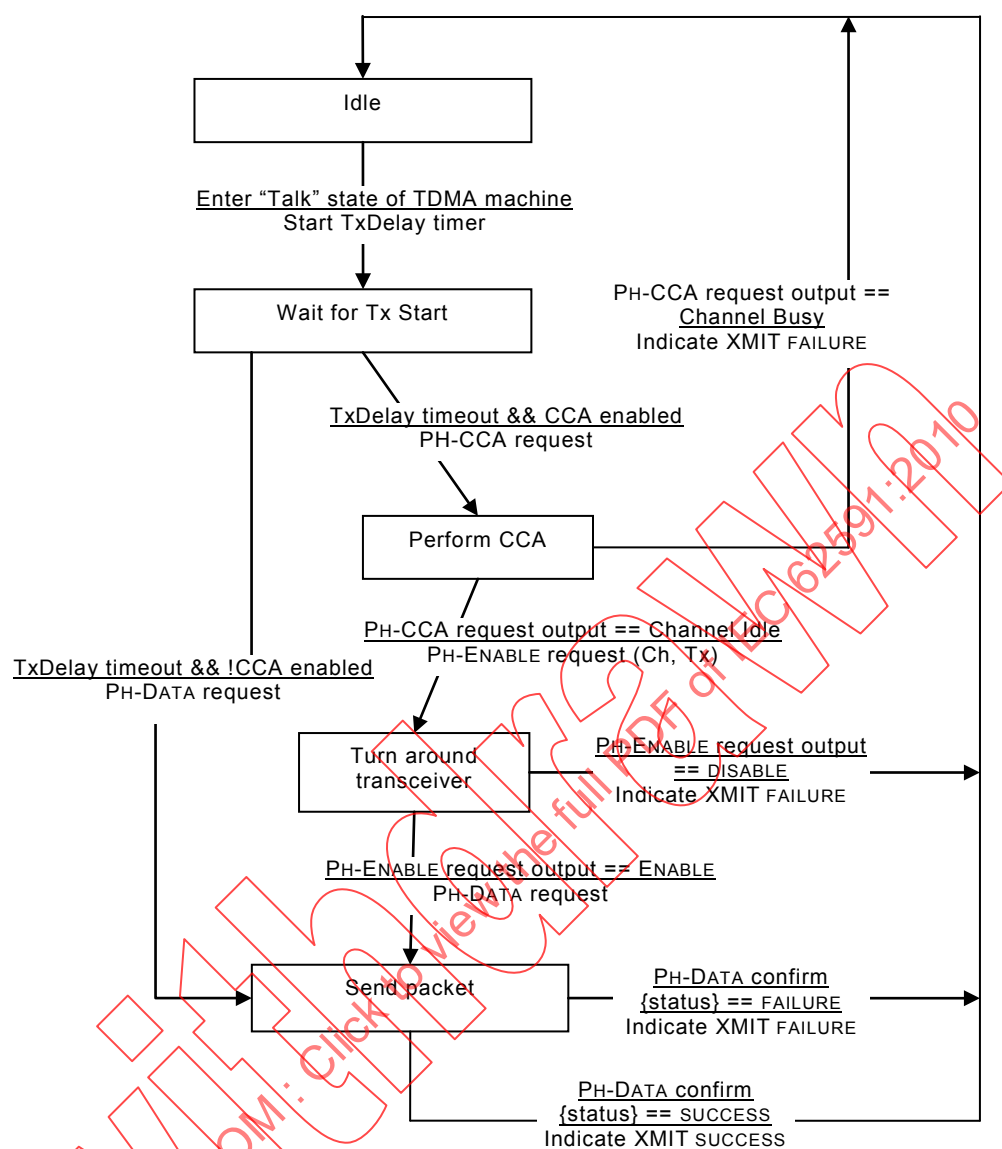


Figure 24 – XMIT state machine

Table 43 – XMIT state transitions

Current state	Event or condition => action	Next state
Idle	Enter "Talk" state of TDMA machine => Start TxDelay timer	Wait for Tx Start
Wait for Tx Start	TxDelay timeout && !CCA enabled => PH-DATA request {DLPDU}	Send packet
Wait for Tx Start	TxDelay timeout && CCA enabled => PH-CCA request	Perform CCA
Perform CCA	PH-CCA request output == Channel Idle => PH-ENABLE request {channel number, transmit}	Turn around transceiver
Perform CCA	PH-CCA request output == Channel Busy => XMIT FAILURE indication	Idle
Turn around transceiver	PH-ENABLE request output == ENABLE PH-DATA request {DLPDU}	Send packet
Turn around transceiver	PH-ENABLE request output == DISABLE XMIT FAILURE indication	Idle
Send Packet	PH-DATA confirm {status} == success => XMIT SUCCESS indication	Idle
Send Packet	PH-DATA confirm {status} == FAILURE => XMIT FAILURE indication	Idle

6.3.5.3.2 Idle state

When XMIT machine is started, the DLE shall start TxDelay timer. The TxDelay time shall be set so that the transmission starts at the centre of the neighbor's receive window. The device may use the TxDelay time to construct the DLPDU, run its AES-128 cipher engine and generate the MIC for the DLPDU.

The machine shall change to "Wait for Tx start" state.

6.3.5.3.3 Wait for Tx start state

While in the "Wait for Tx Start" state, the DLE shall select the channel for transmission (see 6.3.3.3.2). When TxDelay timer expires, the XMIT machine changes the state. Two transmit sequences are possible.

- If CCAEnabled is FALSE, then the DLE shall issue PH-DATA request to transmit the DLPDU and change the state to "Send packet".
- If the CCAEnabled is TRUE then the DLE shall issue PH- CCA request and change the state to "Perform CCA".

6.3.5.3.4 Perform CCA state

The DLE shall wait until the PhE has completed the clear channel assessment.

- If the DLE receives CCA request output with "Channel idle" status from the PhE then
 - the XMIT machine shall change to "Turn around transceiver" state;
 - the PhE shall be initialized using the PH-ENABLE request to place the transceiver into transmit mode.
- Else, if the DLE receives CCA request output with "Channel busy" status, then XMIT engine shall terminate, indicate that the transmission attempt failed and shall not transmit the DLPDU.

6.3.5.3.5 Turn around transceiver state

When the DLE receives PH-ENABLE request output with Enable status from the PhE, the XMIT machine shall change to "Send packet" state. Else, if the returned status is "Disable", then XMIT engine shall terminate, indicate that the transmission attempt failed and shall not transmit the DLPDU.

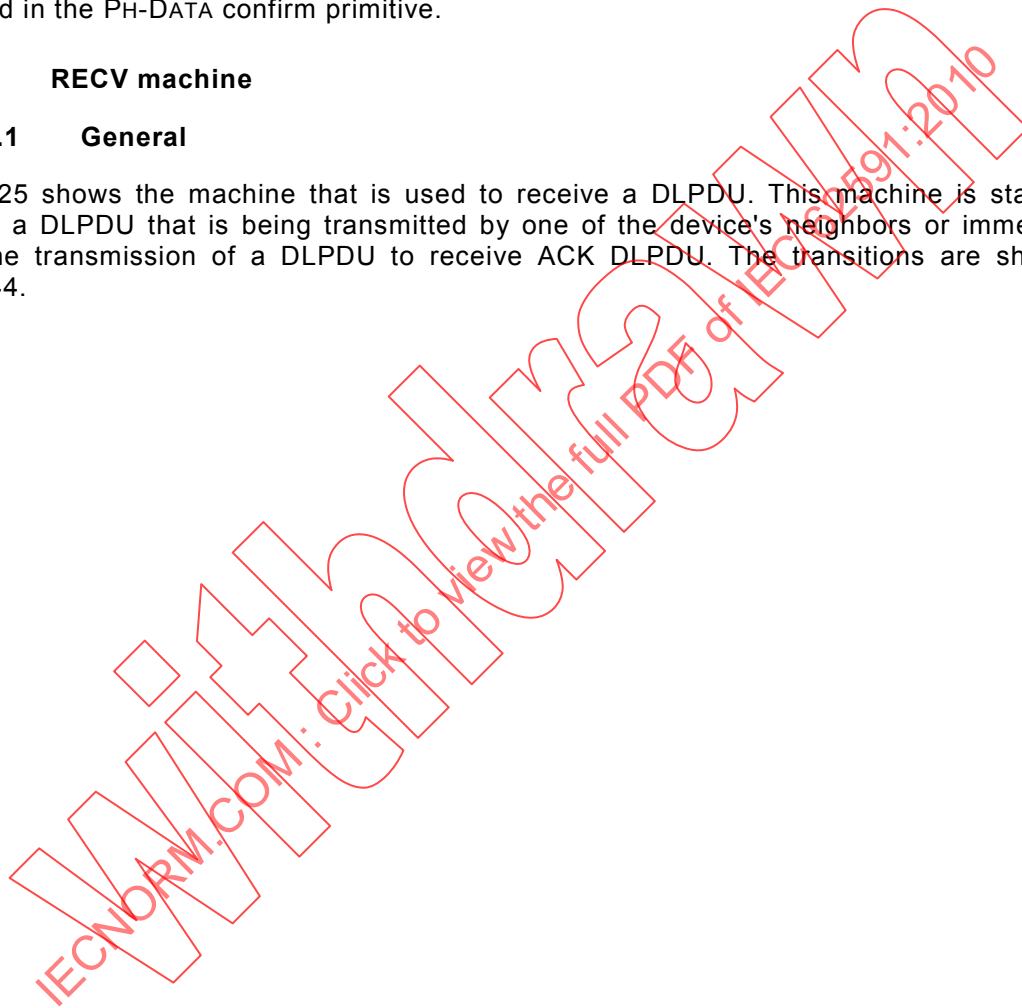
6.3.5.3.6 Send packet state

The DLE shall start the packet transmission using the PH-DATA request service of the PhE. When the DLE receives PH-DATA confirm from the PhE, the XMIT machine shall terminate and indicate that the transmission was either successful or failed, depending upon the status returned in the PH-DATA confirm primitive.

6.3.5.4 RECV machine

6.3.5.4.1 General

Figure 25 shows the machine that is used to receive a DLPDU. This machine is started to receive a DLPDU that is being transmitted by one of the device's neighbors or immediately after the transmission of a DLPDU to receive ACK DLPDU. The transitions are shown in Table 44.



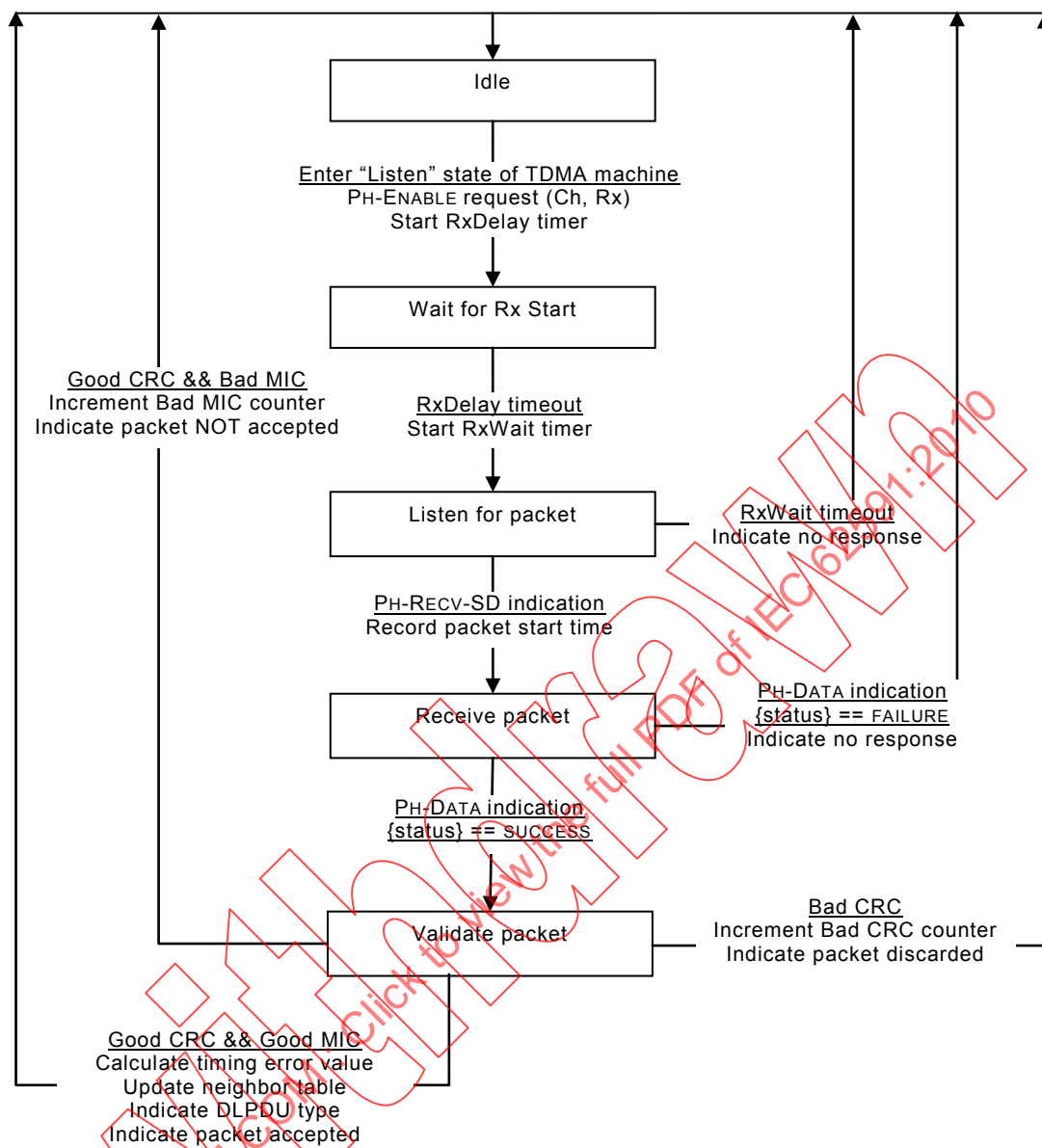


Figure 25 – RECV state machine

Table 44 – RECV state machine

Current state	Event or condition => action	Next state
Idle	Enter "Listen" state of TDMA machine => PH-ENABLE request {channel number, receive} Start RxDelay timer	Wait for Rx start
Wait for Rx start	RxDelay timeout => Start RxWait timer	Listen for packet
Listen for packet	RxWait timeout => Indicate no response	Idle
Listen for packet	PH-RECV-SD indication => Record packet start time	Receive packet
Receive packet	PH-DATA indication {status} == FAILURE => Indicate no response	Idle
Receive packet	PH-DATA indication {status} == <u>SUCCESS</u>	Validate packet
Validate packet	Bad CRC => Increment Bad CRC counter; Indicate packet discarded	Idle
Validate packet	Good CRC && Bad MIC => Increment Bad MIC counter; Indicate packet NOT accepted	Idle
Validate packet	Good CRC && Good MIC => Calculate timing error value; Update neighbor table; Indicate DLPDU type; Indicate packet accepted	Idle

6.3.5.4.2 Idle state

When RECV machine is started:

- the DLE shall select a channel based on the link's starting channel offset and current absolute slot number;
- the DLE shall place the transceiver into receive mode using the PH-ENABLE request service of the PhE;
- the DLE shall start the RxDelay timer; and
- RECV machine shall change to "Wait for Rx start" state.

During the RxDelay the transceiver settles and synchronizes to the correct channel.

6.3.5.4.3 Wait for Rx start state

When RxDelay timer expires, the RECV machine shall change to "Listen for packet" state and start RxWait timer.

6.3.5.4.4 Listen for packet state

The transceiver is instructed to begin listening to the channel by using the Ph-Recv-SD indication service of the PhE. The RECV machine stays in the "Listen for Packet" state until either 1) the start of a DLPDU as indicated by reception of the end of a PhPDU start delimiter is detected or 2) the RxWait Timer expires. If the RxWait timer expires, then receive attempt fails, the RECV engine shall terminate indicating "No Response" and the DLE shall update the communication statistics.

If the expected start delimiter is detected, its time of arrival shall be recorded and the RECV engine shall change to the "Receive Packet" state to capture the DLPDU.

6.3.5.4.5 Receive packet state

When RECV machine receives the PH-DATA indication service primitive from the PhE, it shall perform an initial evaluation of the received DLPDU. If the DLPDU addresses are not as expected for the link, the RECV machine shall terminate indicating "No Response" and the DLE shall update the communication statistics. Else, the RECV machine shall change to "Validate packet" state.

6.3.5.4.6 Validate packet state

If there are no addressing errors in the received DLPDU, then further validation shall be done. If the CRC is incorrect then the PhPDU was corrupted before or during reception; the RECV machine shall terminate indicating "packet discarded" and the DLE shall update the communication statistics.

If the received CRC is correct, the received keyed MIC shall be computed and checked. If the received MIC is not as expected, it indicates a possible attack. Therefore, reception is considered a failure and the DLE shall update both security and communication statistics. The RECV machine shall terminate indicating "packet NOT accepted".

If both the CRC and MIC are correct, then the reception is considered successful and the RECV machine shall terminate indicating the type of the DLPDU received.

6.3.5.5 ACK transmission machine

This machine is started to acknowledge the reception of a DLPDU. The DLE shall copy the measured TsError into the Time-adjustment field of that ACK DLPDU. The TDMA machine shall use the state machine shown in Figure 26 to transmit the DLPDU. The state transitions are shown in Table 45.

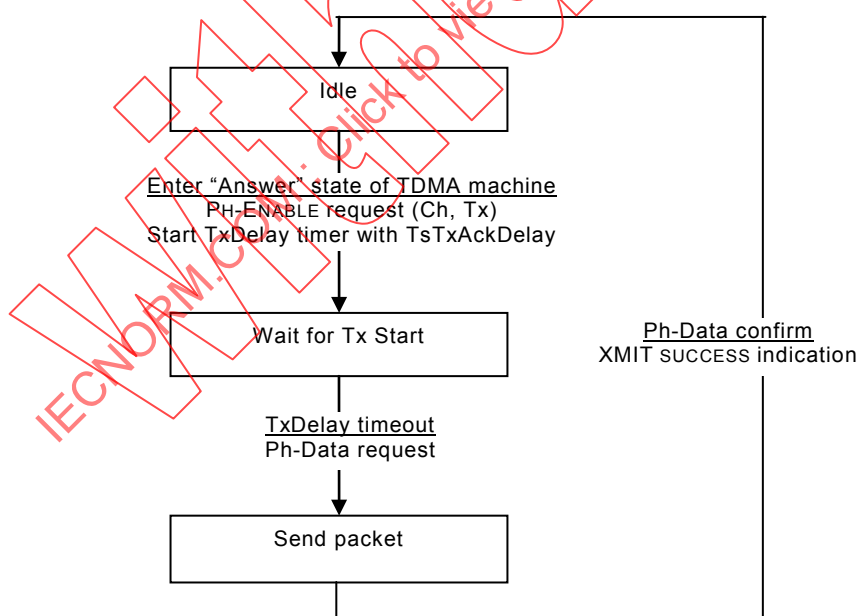


Figure 26 – ACK transmission state machine

Table 45 – ACK transmission state machine

Current state	Event or condition => action	Next state
Idle	Enter "Answer" state of TDMA machine => Ph-Enable request {channel number, transmit}; Start TxDelay timer with TsTxAckDelay;	Wait for Tx Start
Wait for Tx Start	TxDelay timeout => PH-Data request {ACK DLPDU}	Send Packet
Send Packet	PH-Data confirm => XMIT SUCCESS indication	Idle

6.4 DL-management-information

6.4.1 General

Several DL attributes shall be preconfigured before proper DLE operation is possible. These attributes are defined in 6.4.2. The DLE actions which can be requested through DL management are also summarized in 6.4.3.

6.4.2 DL attributes

Table 36 defines the superframe attributes.

Table 37 defines the link attributes.

Table 39 defines the graph attributes.

Network key used for MIC computation, see 6.2.4.2.3

Other attributes are defined in Table 46.

Table 46 – DL attributes

Attribute name	Range of values	Definition
Keep_alive_time	0 to $2^{32}-1$ in units of ms	Interval during which a node shall successfully communicate with each linked neighbor
Path_fail_time	0 to $2^{32}-1$ in units of ms	Interval of unsuccessful communication with a given neighbor, indicating a path failure
Advertise_time	0 to $2^{32}-1$ in units of ms	Interval at which Advertise DLPDUs are transmitted
Discovery_time	0 to $2^{32}-1$ in units of ms	Maximum interval between two consecutive transmission of Advertise DLPDUs on Discovery link
Priority_threshold	ALARM NORMAL PROCESS DATA COMMAND	Specifies the lowest priority DLPDU to be accepted from another device
Join_priority	0 to 15	Priority used in Advertise DLPDU sent by the DLE; lower number indicates a better choice for joining
Promiscuous_mode	TRUE FALSE	Indicates whether the DLE is in "receive all" mode i.e. the DLE accepts all DLPDUs received from the PhE.
Max_backoff_exp	{4, 5, 6, 7}	The maximum value that can be assumed for the back-off exponent used in shared slots, the default value is 4.

6.4.3 DLM actions

The DLE actions which DL management may request are defined in Table 47.

Table 47 – DL actions

Name	Description
RESET	Initialize the DLE
DISCONNECT	Disconnect the DLE from the network and cease all communications
RE-JOIN	Disconnect the DLE from the network; purge all outstanding DLS-user requests, clear all buffers and queues and clear all tables; rejoin the network
DELETE-SUPERFRAME	Delete an existing superframe and any associated links
DELETE-LINK	Delete an existing link, possibly updating the neighbor and connection tables associated with that link
DELETE-GRAPH	Deletes an existing graph connection

6.5 Network layer

6.5.1 Two level model of NL

6.5.1.1 General

The Network layer is modeled as

- a higher-level security function, and
- a lower-level routing function.

In addition the NL-management function operates with both levels.

6.5.1.2 Security function

A keyed message integrity code (MIC) is used for authentication of the NPDU, the encryption and the decryption of the NL payload. The keys are distributed by network manager.

6.5.1.3 Routing function

6.5.1.3.1 General

The network layer provides routing services. It manages sessions for end-to-end communication with correspondent devices. It receives the NPDU from the DLE. If the destination of the NPDU is the local NLS-user, then it transfers the user data to its user. Else, it forwards the NPDU destined to other devices using DLE. It also processes the NL-TRANSMIT request and prepares the NPDU for transmission.

The network layer of this standard routes a NPDU using one of the two methods.

- Graph – the network topology can be represented as all of the devices and the directed links between them. A graph is a subset of the directed links and devices that provide redundant communication routes between a source and a destination device. The route actually taken is based on current network conditions when the NPDU is conveyed from the source to the destination. The NPDU contains the Graph_ID, but does not specify the complete route. Therefore, each intermediate DLE in the route shall be configured with a fragment of the overall graph prior to using that graph route. Graph routes are redundant and highly reliable. This method of routing should be used for normal, routine communications, both upstream and downstream from the gateway.
- Source – it is a single directed route (devices and links) between a source and a destination device. The complete route is statically contained in the NPDU itself. The

intermediate DLEs require no knowledge of the route in advance. As the NPDU is routed, each intermediate DLE propagates the NPDU to the next DLE using the source route contained in the NPDU. Source Routes are not redundant and may fail at anytime. Current network conditions could break the route specified in the NPDU causing NPDU loss. This method of routing should be used only for testing routes, troubleshooting network paths or for ad-hoc communications such as routing join responses.

All devices are required to support both source and graph routing.

6.5.1.3.2 Graph routing

Graph is a collection of directed paths that connect network endpoints. The specific paths associated with each graph are explicitly configured by the network manager in all network devices. A single network instance may have multiple graphs, some of which may overlap. Each network device may have multiple graphs going through it, even to the same neighbors. Graphs are unidirectional.

Every graph in a network has a unique Graph_ID. When using graph routing, the NLE places the Graph_ID value in NPDU header. NLE receiving the NPDU then forwards it along the set of paths belonging to that graph towards the destination.

Each Graph_ID in the device should have multiple associated neighbors. In a properly configured network, all devices have at least two neighbor devices in the graph to which the NLE can send the NPDU. This provides redundancy in the route and improves the reliability. A NLE examines the Graph_ID in the received NPDU and forwards the NPDU to any of the neighbors listed in that graph.

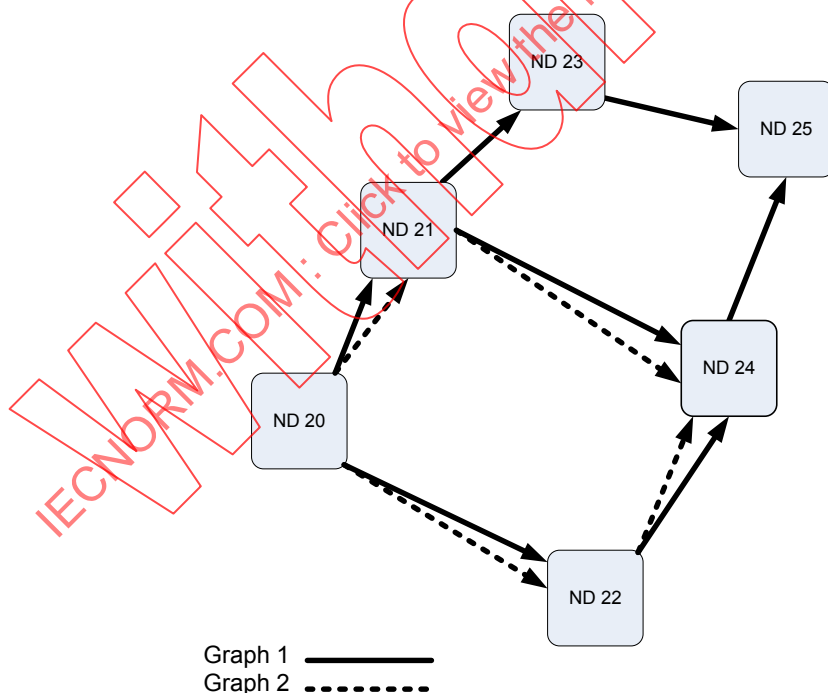


Figure 27 – Graph routing

As shown in Figure 27, for graph 1, ND 20 is the source and ND 25 is the destination; for graph 2, ND 20 is the source and ND 24 is the destination. To send a NPDU using graph 1, ND 20 may forward it to either ND 21 or ND 22. The NPDU may take several alternate routes, but either way, following graph 1 it will end up at ND 25. Thus, one NPDU from ND 20 to ND 25 could take the route through ND 21 and ND 23. Another NPDU from ND 20 to ND 25 could take the route through ND 21 and ND 24. Similarly, to communicate with ND 24, ND 20 sends NPDU using graph 2 through either ND 21 or ND 22.

A special simplified version of graph routing named superframe routing, is also supported by all devices. When superframe routing is performed, the Superframe_ID is placed in the NPDU. If the Graph_ID in the received NPDU does not match any of the Graph_ID configured in the receiving device, then the NLE looks for a superframe with the same value. If successful, the NLE forwards the packet to any neighbor with a link in that superframe. In the case of superframe routing, candidate links are selected based on the Superframe_ID number. In the case of graph routing candidate links are selected based on the graph table entries (Graph_ID & neighbor).

6.5.1.3.3 Source routing

For source routing, the complete route from the source device to the destination device is specified in the NPDU. As the NPDU is propagated, each intermediate NLE looks at the source route address list and forwards the NPDU to the next DLE in the list. The route shall be configured in the NLE, usually by the network manager. Source routing specifies a single path and, if one of the intermediate links fails, then DLPDU is lost. The source routing is less reliable than graph routing. When a source route fails, the NLE at the point of the failure notifies the network manager. It is the responsibility of the network manager to take corrective action.

As shown in Figure 28, the route from ND 20 to ND 24 goes through ND 21, ND 23 and ND 25. The routing table in the NLE of ND 20 is configured with this path. The source NLE adds the list of addresses for ND 21, ND 23 and ND 25 in the header of the NPDU and sends the NPDU to ND 21. The NLE in ND 21 receives that NPDU; examines the header and forwards the NPDU to ND 23. NLE in ND 23 forwards the NPDU to ND 25; which then forwards it to ND 24. The route from ND 20 to ND 24 could also have been configured via ND 22 and ND 25.

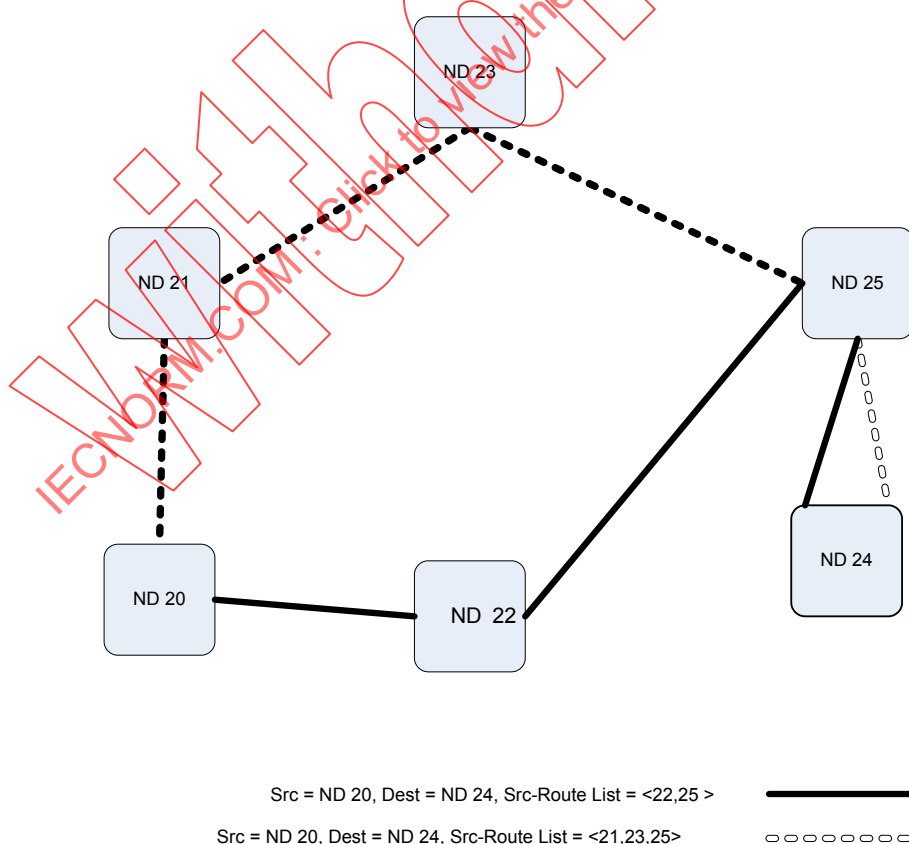


Figure 28 – Source routing

6.5.2 NPDU structure

6.5.2.1 General

Figure 29 shows the NPDU structure. Each NPDU shall consist of the fields in a sequence described below:

- The 1-octet NL Control,
- The 1-octet Hop-to-live (HTL) counter,
- The 2-octets Sequence number,
- The 2-octet Graph_ID,
- Destination and Source Addresses in that sequence, either of which can be either 2 or 8-octets long,
- Zero or more routing octets,
- The 1-octet Security control,
- Counter,
- A 4-octet message integrity code, and
- NL payload.

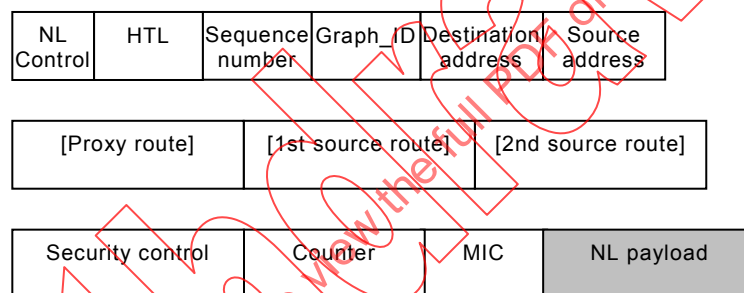


Figure 29 – NPDU Structure

6.5.2.2 NL control octet

This field specifies the length and type of destination, source addresses and presence of routing fields, as shown in Figure 30. The bit-7 specifies the destination address type; the bit-6 specifies the source address type. If the value of any of these two bits is '1', then the address in the NPDU, corresponding to that bit is EUI-64, else it is nickname (see 6.2.1.2.4). Any combination of address lengths can be used in a DLPDU.

The bits 5, 4 and 3 are reserved for future use. The NLE shall set the value of these bits to '000' before transmission. The NLE shall ignore any value of these bits in the received NPDU.

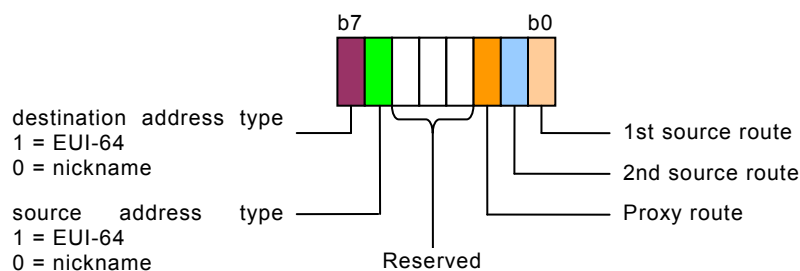


Figure 30 – NPDU control octet

If the value of any of bits 2, 1 and 0 bits is '1', then the corresponding routing field shall be present in the NPDU.

- Bit 2 specifies the presence of 2-octet long Proxy route field.
- Bit 1 specifies the presence of 8-octet long 2nd source route field.
- Bit 0 specifies the presence of 8-octet long 1st source route field.

NOTE The NPDU can specify a valid Graph_ID and source route both simultaneously.

6.5.2.3 Hop-to-live

This field keeps count of remaining number of hops that NPDU can travel before it is discarded.

6.5.2.4 Sequence number

The Sequence Number shall be set equal to the least significant octet of the Absolute Slot Number (ASN) at the time of NL-TRANSMIT request is received. This field provides coarse but critical real-time performance metrics and diagnostic information on the operation of the network. It also provides, when the full ASN is recreated, the age of the packet.

6.5.2.5 Graph_ID

The Graph_ID is used to route the packet to its final destination. The Graph_ID indirectly identifies a list of devices, any of which can be used for forwarding the NPDU towards the final destination.

6.5.2.6 Destination and source Addresses

The NL supports two types of addresses: a 2-octet nickname and an 8-octet IEEE EUI-64 address. See 6.2.1.2.4 for the definition and formats of these addresses. These two addresses are not modified during the propagation of the NPDU.

6.5.2.7 Proxy route

Proxy routing shall be used to communicate with devices that have not yet been integrated into the network by the network manager. This includes devices that are quarantined or are in the process of joining. The 2-octet nickname of the proxy parent for the final destination is indicated in the Proxy-address.

6.5.2.8 Source route

Each Source-route field contains set of four addresses that designate the route the packet shall follow from the original source device to the packet's final destination. Each address is a 2-octet nickname. Any unused address shall be set to 0xFFFF.

If the Source-route field is present, the addresses it contains shall be used to route the packet to its final destination. Source routes should only be used to test or troubleshoot network paths.

6.5.2.9 Security control

This field specifies the security strategy used for the NPDU, as shown in Figure 31. The sub-field consisting of bits 1 to 0 shall be encoded as following:

- 0: session keyed,
- 1: join keyed,
- 2: handheld keyed.

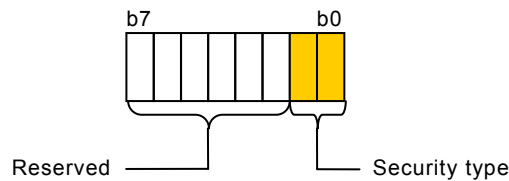


Figure 31 – Security control octet

The bits 7, 6, 5 and 4 are reserved for future use. The NLE shall set the value of these bits to '0000' in the NPDU to be transmitted. The NLE shall ignore any value of these bits in the received NPDU.

6.5.2.10 Counter

This field shall be set to full or truncated value of the nonce counter. The value and length of this field depends upon the security type as shown in Table 48.

Table 48 – Counter value

Security type	Counter value
Session	Least 8-bits of nonce counter
Join	Full 32-bits of nonce counter
Handheld	Full 32-bits of nonce counter

6.5.2.11 Message integrity code

A keyed message integrity code (MIC) is used for authentication of NPDU and the decryption of the security sublayer payload. It is 4-octet long.

6.5.2.12 NL payload

The NLS-user data is the NL payload.

6.5.3 Security function

6.5.3.1 General

The security function is provided at various layers and at overall device operation. The Annex A specifies the communications security functionality and related security aspects.

6.5.3.2 Session

6.5.3.2.1 Session types

The security is session oriented. A session enables private and secure communication between a pair of network devices. Four sessions are set up as soon as any device joins the network.

- Unicast session between the device and network manager – it is used by the network manager to manage the device;
- Broadcast session between the device and network manager – it is used to globally manage devices. For example this can be used to roll a new network key out to all network devices. All devices in the network have the same key for this session.
- Unicast between the device and the gateway – it carries normal communications (for example process data) between the gateway and the device.

- Broadcast session between the device and the gateway – it is used by the gateway to send the identical application data to all devices.

In addition, each device has join session and it cannot be deleted. The Join_key is unique; it is the only key that can be written by directly connecting to the device's maintenance port. It can also be written by network manager.

Additional sessions may be added, if a handheld or another gateway joins the network. The network manager creates all sessions and their corresponding key, except the join-session. Join sessions shall always be present in the device. Only the network manager shall create or modify all sessions, except the join-session. Other device types are not permitted.

It is possible to deploy a session that connects two arbitrary devices in the network. However, this can result in undetected and unmonitored communication between the devices and represents a security and safety threat. If the network manager supports peer-to-peer sessions between field devices, then all of the resulting communications should be routed via the gateway to allow the detection and disruption of malicious behavior.

6.5.3.2.2 Session table

Table 49 shows the attributes of the session. The combination of the correspondent addresses and Session_type shall be unique i.e. two sessions are not allowed to have both, the identical correspondent addresses and Session_type values.

When and only when a new session is created, My_nonce_counter shall be reset to zero. Each element in the Nonce_counter_history shall be set to the nonce counter value in the NLM-SET command to write values of session attributes. The device is not permitted to change Session_key value that was written by the network manager. The network device shall not allow any means to read back any session key.

Table 49 – Session attributes

Attribute name	Range of values	Definition
Session_type	UNICAST BROADCAST JOIN	Type of session
Peer_unique_ID	0 to $2^{40}-1$	Unique ID of the correspondent device as defined in 3.3.91
Peer_nickname	0 to 65 534	Nickname of the correspondent device as defined in 3.3.67
Session_key	128-bit number	Session key
Peer_nonce_counter	32-bit number	Largest nonce counter value received from the correspondent device
Nonce_counter_history	array of 32 bits	An array of bits recording the nonce counters received; most significant bit is always set and corresponds to the current Peer_nonce_counter value. The least significant bit corresponds to '1 + Peer_nonce_counter – size of Nonce_counter_history'. This creates a sliding window with each set bit recording a NPDU received with the corresponding nonce counter. The size of Nonce_counter_history equals the number of received NPDUs recorded. As the Nonce_counter_history is right shifted, history of the oldest recorded NPDUs is dropped.
My_nonce_counter	32-bit number	Nonce counter for packets sourced by the device
Transport_table	Internal reference	Reference to transport tables (normally 2) associated with correspondent
Route_table	Internal reference	Reference to Route tables associated with correspondent destination

6.5.3.2.3 Join session

Join session is unique and is created by the network device when the Join_key is first written via the maintenance port. The addresses of the 'join session' shall be set to the standard network manager addresses. The My_nonce_counter entry shall be reset to zero when the session is created. The network device's My_nonce_counter is used for both join requests from the network device and join responses from the network manager. To protect against replay attacks on the Join_key, My_nonce_counter shall be stored in non-volatile memory.

Several strategies may be used to create the 'join session' on the network manager. For example a 'join session' for a specific field device could be created in advance by using the field device's unique ID and the Join_key. No matter the strategy used to create the 'join session', when the network manager receives and validates a join request, the NPDUs counter field is copied to the session's nonce counter entries and is used to construct the join response.

6.5.3.3 Security function elements of procedure

6.5.3.3.1 Receipt of NL-Transmit request primitive

When a NL-TRANSMIT request is received, the NLE shall construct the security header part of the NPDUs for that request. A keyed MIC is used to ensure that the NPDUs arrives successfully and unmolested from the indicated source device. The MIC is generated and confirmed using CCM* mode (combined counter with CBC-MAC (corrected)) in conjunction with the AES-128 block cipher to provide authentication as specified in IEEE 802.15.4-2006, Annex B. This cipher requires four octet-strings as parameters:

- 'a', the additional data to be authenticated but not enciphered;
- 'm', is the message to be enciphered;

- 'N', the 13-octet nonce; and
- 'K', the 128-bit AES Key.

The NL payload is enciphered and is the octet-string 'm'. The NPDU header, from the NPDU control octet through the MIC, is the octet-string 'a'. The HTL, Counter and MIC fields in octet-string 'a' are set to zero while authenticating the NPDU. These are replaced with their actual values before transmitting the packet.

The security layer nonce (the 'N' octet-string) is 13-octets long. The nonce is created as summarized in Table 50 and described below.

If the Session_type is "join" and NL-TRANSMIT request primitive is join response

a) then

- 1) the LSB (octet number zero) of 'N' shall be set to '1';
- 2) the next four octets of 'N' shall be set to Peer_nonce_counter value from the corresponding join request starting with the MSB of Peer_nonce_counter;
- 3) the next eight octets of 'N' shall be set to the joining device's EUI-64 address;

NOTE The join request and response always use EUI-64 address, because the device receives its nickname only after the device has successfully joined the network. The join response can only be transmitted from the network manager.

b) else,

- 1) the LSB (octet number zero) of 'N' shall be set to '0';
- 2) the My_nonce_counter shall be pre-incremented by one;
- 3) the next four octets of 'N' shall be set to My_nonce_counter value starting with the MSB of My_nonce_counter;
- 4) the next five octets of 'N' shall be set to the source address field in the NPDU, which is either the EUI-64 address of the zero-padded nickname.

Table 50 – NPDU nonce

Octet number	Description
0	Set to 1 if join response, otherwise set to 0
1 – 4	Nonce counter starting with MSB in N[1]
For EUI-64 source address:	
5 – 12	EUI-64 address is copied (MSB to LSB) into N[5] through N[12]
For 2-octet nickname:	
5 – 10	Each octet is set to 0x00
11 – 12	The nickname is copied (MSB to LSB) into N[11] through N[12]

After the nonce is constructed, the NPDU is enciphered; and then NPDU security fields are populated.

6.5.3.3.2 Receipt of NPDU addressed to the device

6.5.3.3.2.1 General

The NLE shall process the received NPDU using the AES-128 engine to authenticate the NPDU and decipher the payload. This cipher requires four octet-strings as parameters:

- 'a', is the NPDU header, from the NPDU control octet through the NPDU MIC with the NPDU HTL, counter and NPDU MIC fields set to zero;

- 'm', is the NL payload;
- 'N', the 13-octet nonce; and
- 'K', the 128-bit AES Key.

Authentication and decipherment start by locating the correct session see 6.5.2.9 to determine the key and nonce counters.

If the received NPDU is not discarded, then the payload shall be forwarded to the NLS-user in NL-Transmit indication primitive. The NL payload shall be the NLS-user-data.

6.5.3.3.2.2 Nonce re-construction

The NLE shall re-construct the nonce for the received NPDU, as summarized in Table 50 and described below.

If the Session_type is "join"

a) then

- 1) If the received NPDU is join request, then
 - i) the LSB (octet number zero) of 'N' shall be set to '0';
 - ii) the next four octets of 'N' shall be set to Counter field of the received NPDU starting with the MSB of counter;

NOTE The join request NPDU is received only by the network manager.

2) else,

- i) the LSB (octet number zero) of 'N' shall be set to '1';
- ii) if the Counter field of the received NPDU is equal to My_nonce_counter, then the next four octets of 'N' shall be set to My_nonce_counter value starting with the MSB of My_nonce_counter;
- iii) else, the NLE shall discard the received NPDU.
- 3) If the NPDU was not discarded then the next five octets of 'N' shall be set to the source EUI-64 address from the received NPDU;

b) else,

- 1) the LSB (octet number zero) of 'N' shall be set to '0';
- 2) the next three octets of 'N' shall be set to the most-significant three octets of the corresponding Peer_nonce_counter value starting with the MSB of Peer_nonce_counter;
- 3) if the 1-octet Counter value of the received NPDU is less than the quantity (1 + LSB of Peer_nonce_counter – size of Nonce_counter_history), then the 24-bit value in N[1]-N[3] shall be incremented;
- 4) the next octet of 'N' shall be set to Counter field of the received NPDU;
- 5) the next five octets of 'N' shall be set to the received NPDU source address field, which is either the EUI-64 address or the zero-padded nickname.

6.5.3.3.2.3 Authentication

If the nonce is successfully constructed and the received NPDU is not a duplicate of a previously received NPDU, authentication and decipherment shall be performed. If this authentication fails, the packet shall be discarded.

6.5.3.3.2.4 Replay Protection

If the packet is authentic then the Nonce_counter_history shall be updated. Since NPDU can arrive out of order or be lost completely, this sliding window algorithm is used to facilitate the

communications and eliminate duplicate packets. The Peer_nonce_counter and the Nonce_counter_history together chronicle the NPDUs received by the device.

The octet number 1 to 4 of 'N' is the reconstructed nonce counter. The reconstructed nonce counter shall be compared to the Nonce_counter_history. If it corresponds to any of the bits set in the Nonce_counter_history, the received NPDu is a duplicate of previously received NPDu and the NPDu shall be discarded. If the reconstructed nonce counter is outside of the window of the Nonce_counter_history, then the DLE shall discard the received NPDu.

If the reconstructed nonce counter is less than the Peer_nonce_counter, then the appropriate bit in the Nonce_counter_history shall be set. Otherwise, the Nonce_counter_history shall be right shifted and the Peer_nonce_counter value updated. This is accomplished by subtracting the Peer_nonce_counter from the reconstructed nonce counter. The resulting difference indicates the number of times the Nonce_counter_history shall be right-shifted. After shifting, the most significant bit of the Nonce_counter_history shall be set. Finally, the reconstructed nonce counter shall be copied to the Peer_nonce_counter in the session table entry.

6.5.4 Network layer tables

6.5.4.1 General

All devices shall maintain a series of tables that control the communications performed by the device, supply routing information, support end-end acknowledgements, ensure the privacy of the communications, and collect statistics on those communications.

The tables controlling communication activities are:

- Session table – an entry in this table establishes a secure pipe between the device and a specific correspondent device. No communication to a correspondent device is possible without a security session (see 6.5.3.2.1). See 6.5.3.2.2 for session attributes.
- Transport table – it is used to support end-end acknowledge transaction with automatic retries (see 6.6.3).
- Route table – it serves as the locus for adding routing information to a new NPDu being generated as the result of a NL-TRANSMIT request (see 6.5.4.2).
- Source route table – it is attached to some routes (see 6.5.4.3).
- Service table – it indicates the Route associated with a service allocated by the network manager (see 6.5.4.4).

The relationship among these tables is shown in Figure 32. Within the device the session table is central. The NLE also uses Neighbor table (see 6.3.3.4) and Graph table (see 6.3.3.5) of the DLE. The figure shows the relationships around a route. A route is identified by a route_ID. A route is associated with a specific correspondent. A route may contain an optional source route. A route is always associated with a graph. In some cases the specialized form of graph, superframe, is used. A graph may be associated with 0 (superframe) or more neighbor. A Timetable is associated with one route. A route is associated with 0 or more Timetables. The relationship with Timetables is directed, because the Timetable knows about the route but the route does not need to know about the Timetable.

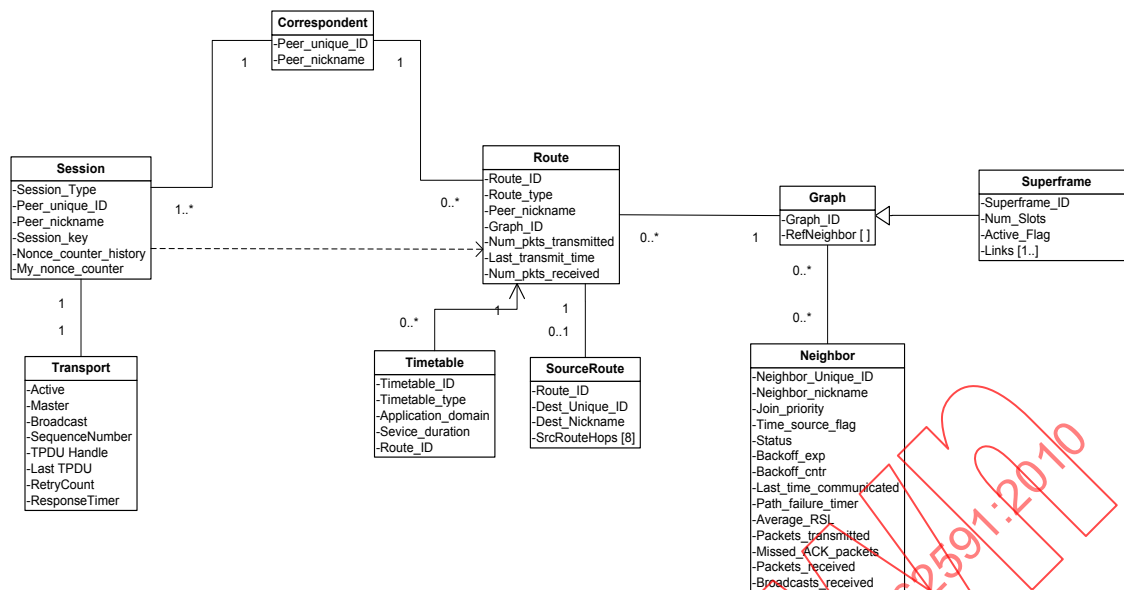


Figure 32 – NLE tables and relationship

Devices shall support the minimum number of table entries and buffer space shown in Table 51.

Table 51 – Minimum NLE Table space requirement

Table name	Required minimum entries
Session	8
Transport	16 ^a
Route	8
Source-route	2
Service	16
^a 2 per session, if session is greater than 8.	

NOTE Some of the attributes of these tables described in this standard may be calculated or derived from other information, and it is not necessarily to store these separately. Although specific implementation of data and configuration storage is left up to the designers of the device, general description of the fields contained in the data structures is important to overall operation.

6.5.4.2 Route table

Table 52 shows the route attributes. The route table is associated with a security session and a correspondent address. This is used to aid in the selection of the graph used for communication to the correspondent device. NLE shall collect statistics to allow the network manager to optimize communication resources over time.

Table 52 – Route attributes

Attribute name	Range of values	Definition
Route_ID	0 to 255	Route identifier
Route_type	MAINTENANCE PUBLISH BLOCK TRANSFER EVENT	Indicates the type of route; MAINTENANCE is default value.
Dest_nickname	0 to 65 534 – see 6.2.1.2.4	Nickname of the destination device with which the route is established
Graph_ID	256 to 65 535	Identifier for the graph associated with this route
Src_route_ID	0 to 255	If non-zero, then the identifier of the source route table for this route, else absence of the source route for this route
Num_pkts_transmitted	0 to 65 535 ^a	Number of packets Transmitted on this route
Last_transmit_time	0 to 2 ³² -1 in the units of 1/32 ms	Time last NPDU was generated for this route
Num_pkts_received	0 to 65 535 ^a	Number of packets received on this Route updated based on the Graph_ID in the NPDU
a The value of packet counters rolls over from 65 535 to 0.		

6.5.4.3 Source route table

The source route table contains a list of up to 8 device addresses tracing the route from this device to the correspondent device as shown in Table 53.

Table 53 – Source route attributes

Attribute name	Range of values	Definition
Route_ID	1 to 255	route identifier
dest_Unique_ID	0 to 2 ⁴⁰ -1	Unique ID of the destination device as defined in 3.3.91
dest_Nickname	0 to 65.534	The short address of the neighbor as defined in 3.3.67
Src_route_hops [8]	Device Nickname	Up to 8 nicknames that lead to the destination. Unused addresses shall be set to the destination's nickname

6.5.4.4 Timetable table

When a device needs to establish a communication connection with another device it sends a request to the network manager to allocate a Timetable for that communication. Each destination and Application_domain has its own Timetable. Table 54 shows the Timetable attributes. It identifies the route to the destination device and the scheduling quota allocated for this communication. When the device deletes a Timetable, the network manager is expected to de-allocate the network resources supporting that Timetable. The same route may be used by more than one Timetable.

Table 54 – TimeTable attributes

Attribute name	Range of values	Definition
Timetable_ID	0 to 255	Timetable identifier
Timetable_type	SOURCE SINK INTERMITTENT	Indicates the type of Timetable
Application_domain	MAINTENANCE PUBLISH BLOCK TRANSFER EVENT	The application domain using this Timetable
Dest_nickname	0 to 65 534	The short address of the destination for which this Timetable has been allocated
Service_duration	0 to $2^{32}-1$ in the units of 1/32 ms	For cyclical communications (for example Publish data), packets are generated with this period. Latency for cyclical communication shall not exceed 1/3 the period. For intermittent communications, this is the required maximum latency.
Route_ID	0 to 255	Route identifier for this Timetable

6.5.5 NLE machine

6.5.5.1 General operation

The NLE progresses through a series of states starting in the 'Idle' state and continuing until the device is 'Operational' and a full participant in the mesh network. There are five states and they are described in the Table 55.

Table 55 – NLE states

State	Description
Idle	The device is quiescent and its wireless transceiver is not active. It has no knowledge of the network.
Joining	The device is listening for the network, attempting to acquire an Advertise DLPDU and requesting admission to the network.
Quarantined	The device has successfully joined the network but only has a security clearance to talk with the network manager. It is not available or allowed to perform data acquisition or control functions or otherwise communicate with the gateway.
Operational	The device can be accessed by host applications via the gateway. It is integrated in the system's operation.
Suspended	The device is quiescent. All of its network tables are intact.

The state transition diagram is shown in Figure 33. This diagram depicts the events that allow the device to start-up, locate and join the network, and finally progress from being a new network member to becoming a fully operational network device. In addition, the diagram shows the device (and network) being suspended and re-synchronized. Furthermore, the device can be forced to disconnect (for example to be removed from the process for depot-level maintenance) or to re-join at any time.

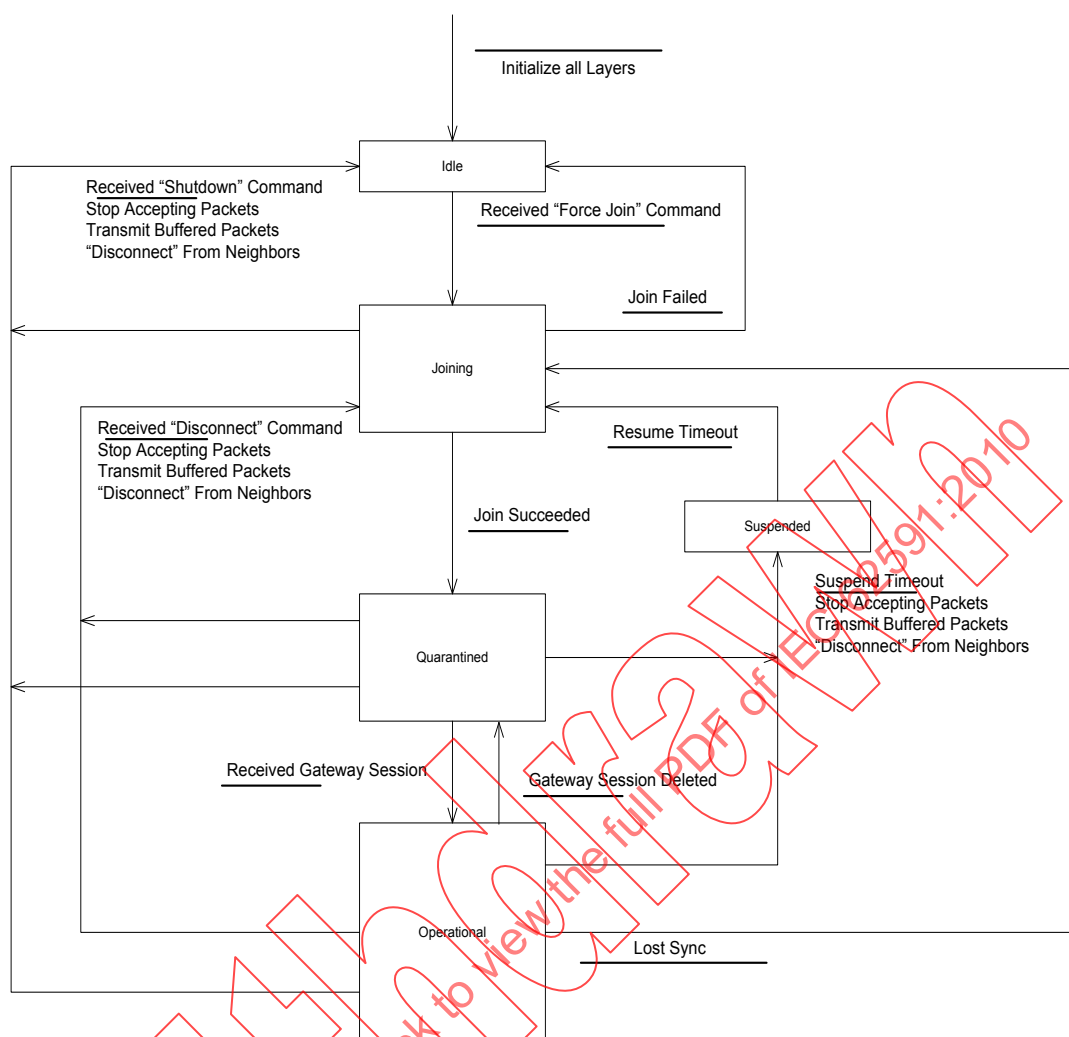


Figure 33 – NLE state machine

6.5.5.2 Idle state

After a device reset or on power up the device shall enter the 'Idle' state. While in this state the device shall not attempt to communicate wirelessly. The device shall update the Join_key and network_ID, if it is written via the maintenance port. When the device receives the Join command from the NL-management, the NLE shall change to 'Joining' state.

6.5.5.3 Joining state

The NLE shall locate the network and synchronize to the network. It shall wait to receive Advertise packets. After receiving one or more Advertise packets, the NLE shall request admission to the network, using the Join_key. The network manager is expected to send network key and network manager session to the requesting NLE. The network manager is also expected to provision the NLE with superframe, graphs and links. After the NLE has received these parameters, it shall change to 'Quarantined' state.

The Join request may be answered by the network manager immediately or the NLE may need to retry the request. A failure shall occur if the device sends number of join requests equal to max_join_retries (see Table 58), without receiving the security keys and session parameters from the network manager. In this case, the NLE shall return to the 'Idle' state.

If the NLE receives Disconnect command from the NL-management, then it shall change to 'Idle' state.

6.5.5.4 Quarantined state

Quarantined devices are limited network partners. The NLE shall remain in this state until approved for deployment into the network application. Approval may take the form of the human operator authorizing the use of the device in the system and placing the device into service. This quarantine step parallels normal practices in the process industry and adds an additional layer of security to the join process.

While quarantined the device shall only be enabled to source and sink packets, but not forward any packet. Normal network reports (for example neighbor reports) shall be generated.

After receiving the gateway session parameters, the NLE shall change to 'Operational' state.

6.5.5.5 Operational

This allows host application access to the device and begin its interaction with the device.

In the operational state, the device is a full network partner and shall begin performing the tasks designated by its configuration. It shall route packets onward through the network to their final destination. The device shall also participate in advertising and support new devices joining the network. If configured to publish data, it shall request bandwidth to do so. It shall receive user requests and responses for transmission.

6.5.5.6 Suspended state

When in 'Quarantined' or 'Operational' state, the device may be forced into 'Suspended' state for the specified time. This is normally a brief amount of time and done, for example for safety purposes during mining blast operations. In this state the device continues to track elapsed time while leaving the radio disabled. After the specified time has elapsed, the device shall change to the 'Joining' state.

6.5.6 NLE elements of procedure

6.5.6.1 Receipt of a NL-TRANSMIT request primitive

When a NL-TRANSMIT request is received from the NLS-user, the NLE shall use the Timetable_ID received in the service primitive to locate the Timetable for this request. If the destination is not the network manager and if there is insufficient remaining bandwidth in the Timetable for this request then an error shall be returned and the request shall be deleted.

the NLE shall construct the NPDU for that request. The field shall be set according the procedures indicated in Table 56. The NLE shall find the Route_ID from the Timetable attribute as shown in Table 54. Once the route has been identified the Graph_ID from rote attribute, as shown in Table 52, shall be copied into the NPDU.

The packets shall be graph routed, unless the route is being tested. However, if a source-route is attached to the route, then the source route addresses shall be included in the NPDU. Unused source route address entries in the NPDU shall be set to 0xFFFF. If source routing is used the Graph_ID shall also be valid.

Table 56 – NPDU construction

Field	Procedure
Control	Set the address bits based on source and destination address sizes. Destination addresses are indicated in the request primitive. If the destination is broadcast address, then the destination address size shall be short. Source address shall be the device's EUI-64 address until the device's nickname is initialized. Set the routing bits based on results of route selection.
HTL	Set to Default_HTL.
Sequence number	Set to current ASN value.
Graph_ID	Set based on results of route selection.
Destination address	Set to the value in the request primitive. If the destination address is broadcast address, then the value of this field shall be 0xFFFF.
Source address	Set to EUI-64 address if nickname not initialized, else set to nickname.
Proxy route	Set based on results of route selection.
Source routes(s)	Set based on results of route selection.

6.5.6.2 Receipt of a DL-RECEIVE indication primitive

The NLE shall examine the destination address in the NPDU received from the DLE. It shall route these packets to the correct destination: forward to the next DLE in the route, forward to the joining device, or deliver to the local NLS-user.

6.5.6.3 Receipt of DL-TRANSMIT confirm primitive

6.5.6.3.1 General

The NLE shall delete all data and table items for the service associated with the handle and return confirm to the NL-user.

6.5.6.3.2 NPDU addressed to the device

If the destination in the received NPDU is the local NLS-user, then the NLE shall forward it to the security function.

6.5.6.3.3 NPDU with broadcast address

If the destination address in the received NPDU is the broadcast address then the NLE shall forward the NPDU as described in 6.5.6.3.4 and also forward it to the security function.

6.5.6.3.4 Forwarding Received NPDU

The NLE shall process the received NPDU in the sequence described below.

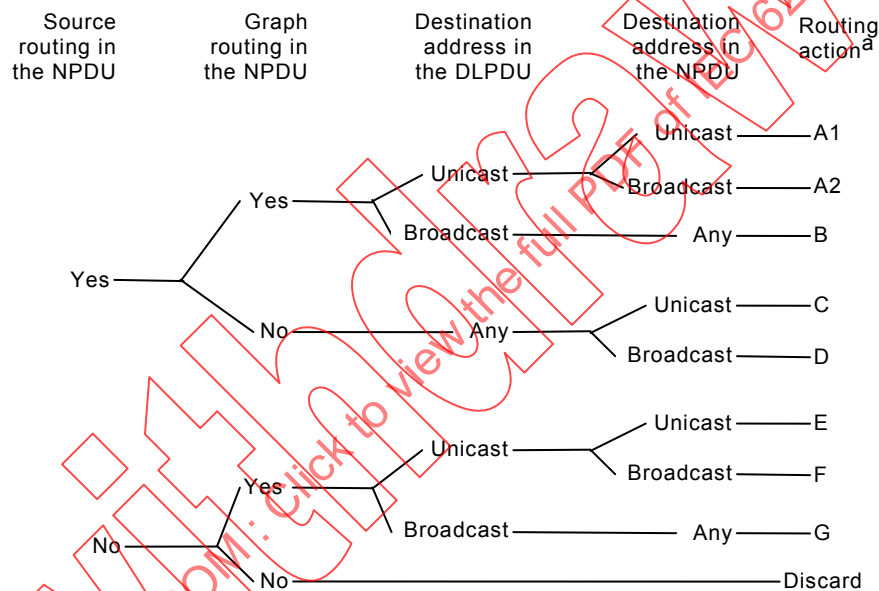
The NLE shall check and update the HTL. If the HTL count in the received NPDU is zero,

- a) then the NPDU shall be discarded;
- b) else,
 - 1) if the HTL is not 0xFF, then the NLE shall decrement it by 1 and use this value to construct the NPDU to be forwarded;
 - 2) the sequence number shall be inspected and, if the age of the received NPDU is more than max_packet_age, then it shall be discarded;
 - 3) else, the NLE shall check the destination address;
 - 4) if the NPDU contains a proxy address that matches the local NLE address then the final destination is a joining device. The NLE shall route the packet to the joining

device by forwarding the NPDU to the DLE indicating the DLL destination address that of the joining device;

- 5) else, if NPDU destination address is Unicast and it matches a neighbor, then the NPDU shall be routed directly to that neighbor. The device shall route the packet to the neighbor device by forwarding the NPDU to the DLE indicating the DLL destination address that of the neighbor device;
- 6) else the packet shall be routed onward based on the Graph_ID and the “Source route” contained in the NPDU. The routing decisions are shown in Figure 34. When analyzing a received NPDU, the columns in the figure shall be evaluated from left to right i.e., starting with “Source routing in the NPDU”. A “No” in the “Graph routing in the NPDU” column means that the received Graph_ID does not match either a Graph_ID or a Superframe_ID found in the field device that received the NPDU. The decision ends with a letter identifier for the routing decision, which is described in Table 57.

NOTE If the destination address in the received DLPDU is broadcast then the Graph_ID in that NPDU holds the Superframe_ID to identify the corresponding broadcast links.



NOTE ^a See Table 57.

Figure 34 – Routing decision of the NPDU

Table 57 – Routing actions

Action code	Action description
A1	If this was the end of the source route list in the received NPDU or if the next address in the source route is not that of a neighbor, then forward the received NPDU along the graph route in the received NPDU, else forward the received NPDU to the next address in the source route.
A2	If this was the end of the source route list in the received NPDU, then broadcast to the Superframe_ID (Graph_ID contains the Superframe_ID), else forward the received NPDU to the next address in the source route.
B	Continue broadcasting to the graph. Use broadcast link in the superframe.
C	If the received NPDU has another valid destination in the source route list then forward the NPDU to that address. Else, indicate a source route error to the NLM.
D	Forward the received NPDU to the next address in the source route.
E	Forward the received NPDU along the graph, using any normal link to neighbor on the graph; if end of the graph then indicate a graph route error to the NLM.
F	Forward the received NPDU to all neighbors in the graph; discard NPDU if at the end of the graph.
G	Graph_ID is Superframe_ID; continue broadcasting the NPDU using broadcast link in the superframe
Discard	The received NPDU is invalid; discard it.

The timeout value to be passed to the DLE shall be re-calculated using the received ASN.

The NLE shall forward the NPDU to the DLE such that it can be transmitted in the next slot i.e. the slot immediately following the slot in which the PDU was received.

6.5.7 NL-management information

6.5.7.1 General

Several NL attributes shall be configured before proper NLE operation is possible. These attributes are defined in 6.5.7.2. The NLE actions which can be requested through NL management are also summarized in 6.5.7.3.

6.5.7.2 NL attributes

Session attributes are shown in Table 49.

Route attributes are shown in Table 52.

Source-route attributes are shown in Table 53.

Other attributes are defined in Table 58.

Table 58 – NL attributes

Attribute name	Range of values	Definition
Unique ID	0 to $2^{40}-1$	Unique ID of the device as defined in 3.3.91
Nickname	0 to 65 534	Nickname of the device as defined in 3.3.67
Default_HTL	0 to 255	Packet life limit in hops; specifying the number of hops a packet can travel before being discarded; with a default value of 32
max_packet_age	0 to 65 535	Packet life limit in time using the ASN; indicating the number of slots after which a packet shall be discarded; with a default value of 30 000 ASN time (equal to 300 s)
min_ads_needed	0 to 255	Preferred number of different Advertisements before issuing join request; with a default value of 3
Ad_wait_timeout	0 to $2^{32}-1$ in units of ms	The amount of time to wait while attempting to receive additional Advertisements; with a default value of 30 s
max_join_retries	0 to 255	Join retry limit; with a default value of 5
Join_rsp_time	0 to $2^{32}-1$ in units of ms	Join response timeout duration, its default value is Keep_alive_time (see Table 46)
Bcast_reply_time	0 to $2^{32}-1$ in units of ms	Maximum amount of time to reply to a broadcast message; with a default value of 60 s
Health_report_time	0 to 65 535 in units of ms	Interval at which to publish health reports; with a default value of 15 minutes
Session_table_size	up to 255	Size of session table (i.e., maximum number of entries)
Num_session_table_entries	up to 255	Number of entries currently in the session table
Route_table_size	up to 255	Size of route table (i.e., maximum number of entries)
Num_route_table_entries	up to 255	Number of entries currently in the route table

6.5.7.3 NLM actions

The NLE actions which NL management may request are defined in Table 59.

Table 59 – NL actions

Name	Description
RESET	Reset and initialize the NLE. All network tables are cleared when this primitive is invoked. This primitive is normally invoked on device power-up or when the device is being installed in a new network
WRITE-SESSION	Adds a new session
DELETE-SESSION	Deletes an existing session
ADD-ROUTE	Adds a new route
DELETE-ROUTE	Deletes an existing route
DEFAULT-ROUTE	Sets a route as default route
ADD-TIMETABLE	Adds a new Timetable
DELETE- TIMETABLE	Deletes an existing Timetable
JOIN	Changes state from Idle to Joining
DISCONNECT	Changes state to Idle

6.5.7.4 Network management

The network manager is responsible for the overall management, scheduling, and optimization of the network. The network management function is specified in Annex C.

6.6 Transport layer

6.6.1 General

The transport layer is used to ensure that the TLS-user data is communicated to its final destination. The transport layer supports both acknowledged and unacknowledged transactions.

Unacknowledged service allows a device to send TLS-user data without requiring acknowledgement from the receiving end. The data may be delivered at the destination device in a different order than the order in which it was originally sent from the source device.

The acknowledged service is used to construct a synchronous transport pipe across the network connecting two devices. This pipe allows devices to send data and confirm its delivery. The TLS-user can send data which will be delivered without loss, duplication or misordering to the receiving TLS-user. This service is suited for user request and response traffic and event notifications. When the acknowledged service is used, the communication is synchronous. Only one outstanding transaction across the network for that transport pipe is permitted at a time.

6.6.2 TPDU structure

6.6.2.1 General

Figure 35 shows the TPDU structure. It shall consist of the fields in a sequence described below:

- A single octet transport control, and
- TL payload.



Figure 35 – TPDU Structure

6.6.2.2 Transport control

This field specifies the length and type of service as shown in Figure 36. It has four sub-fields.

- TACK: If bit-7 is set to '1' then the TPDU is for acknowledged service, else it is for unacknowledged service;
- TRESP: If bit-6 is set to '1' then the TPDU is for response as specified transport-type parameter of the TLS primitive,
- TBROAD: If bit-5 is set to '1' then the TPDU is for broadcast pipe; this bit is set to '1' for request and response both,
- TSEQN: the bits 4 – 0 form the sequence number for the TPDU to detect duplicate, missing TPDU and to correlate the response with request.

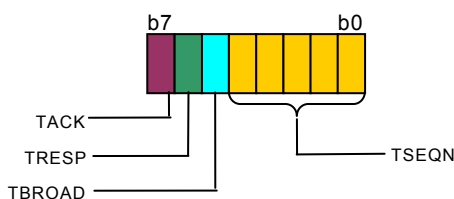


Figure 36 – TPDU control octet

6.6.3 Transport pipe table

Table 60 shows the transport pipe attributes. For each acknowledged data exchange a new entry in the transport pipe table shall be created. Unacknowledged data transfer does not require an entry in this table.

The field device is required to have capacity for following transport pipes:

- unicast data exchange slave with network manager,
- broadcast data exchange slave with network manager,
- unicast data exchange slave with gateway,
- master for event notifications to the network manager, and
- master for event notifications to the gateway.

The gateway and the network manager are required to track many transport pipes for each device and pipes for all of field devices connected to the network. When an acknowledged broadcast is generated by the network manager or gateway, the TLE shall ensure acknowledges are received from all affected devices, which can potentially be all devices.

Each entry in the Transport pipe table includes attributes indicating whether it is active (a transaction is in process), the device is performing as the master, and whether it is broadcast. The TL-Data-exchange request originates at the master end of the pipe. The TL-Data-exchange indication is provided at the slave end of the pipe.

If the transport pipe table entry is for a master end of the pipe, then the retry count and retry timer are also required.

Table 60 – Transport pipe attributes

Attribute name	Range of values	Definition
Active	TRUE, FALSE	TRUE if the transport pipe is active
Master	TRUE, FALSE	TRUE if it is the master end of the pipe
Broadcast	TRUE, FALSE	TRUE if the transport pipe is for a broadcast request
Bcast_reply_time	0 to 65 535 in units of ms	Maximum amount of time to reply to a broadcast message; default value is 60 s.
sequence_number	0 to 31	On master end of the pipe, the sequence number for the outstanding request; On slave end of the pipe, the sequence number last acknowledged
TPDU_handle	0 to 255	On master end of the pipe only, the handle from the TL-DATA-EXCHANGE request corresponding to last_TPDU; not used on slave end of the pipe
last_TPDU		On master end of the pipe, the last unacknowledged TPDU (or NULL if none); On slave end of the pipe, the last acknowledged TPDU
retry_count	0 to 255	On master end of the pipe only, number of communication attempts; not used on slave end of the pipe
max_reply_time		Amount of time to wait for a response to the outstanding request, before retrying; default value is 30 s.
max_retries	0 to 255	Number of retries used by the transport layer when performing an acknowledged packet transmission; default value is 5.

6.6.4 TLE elements of procedure

6.6.4.1 Receipt of TL-DATA-TRANSFER request primitive

The TLE shall encode transport control octet as shown in Table 61.

Table 61 – Transport control encoding for TL-DATA-TRANSFER request

Transport-type	tbroad	tack	tresp
Block-transfer request	0	0	0
Block-transfer response	0	0	1
Search-broadcast	1	0	0
Publish-broadcast	1	0	1
Response-broadcast	1	1	1
Publish-unicast	0	0	1
NOTE Other values of Transport-type are used by other services – see Table 62 and Table 63.			

On the Master side of the transport pipe, the TLE shall increment sequence_number and copy its least significant five into TSEQN sub-field. On the Slave (responder) side of the transport pipe, the TLE shall copy the least significant five bits of the Handle received in the request primitive, into TSEQN sub-field. The TLE shall forward the TPDU to NLE using NL-TRANSMIT request primitive. The Timetable_ID parameter value in this request primitive shall be used in the NL-TRANSMIT request primitive.

6.6.4.2 Receipt of TL-DATA-EXCHANGE request primitive

6.6.4.2.1 General

If this transport pipe is waiting for acknowledgement for a previous TPDU, and no more buffers are available then the service fails and TL-DATA-EXCHANGE confirm primitive shall be returned to the TLS-user to signal the error and the request shall be discarded.

Else, the TLE shall check the transport-type parameter of the request primitive and encode transport control octet as shown in Table 62. The Timetable_ID parameter value in this request primitive shall be used in the NL-TRANSMIT request primitive.

Table 62 – Transport control encoding for TL-DATA-EXCHANGE request

Transport-type	tbroad	tack	tresp
Request-unicast	0	1	0
Request-broadcast	1	1	0
NOTE Other values of Transport-type are used by other services – see Table 61 and Table 63.			

6.6.4.2.2 Acknowledged request

The request requires a response from the destination. Therefore TACK sub-field of transport control field is set to '1'. The TLE shall look for an entry in transport pipe table that corresponds to the service primitive. If no such entry exists, then the TLE shall create an entry in transport pipe table and set sequence_number for that entry to a random number.

If a transport pipe is available, and there is already a request in this transport pipe waiting for acknowledgement from the destination TLE, and no more buffers are available then the request fails. TLE shall forward the TL-DATA-EXCHANGE confirm primitive to TLS-user with error status.

If a transport pipe is available, and either there is no pending request in this transport pipe, or a buffer is available then the sequence_number from the table entry shall be incremented and the least significant five bits shall be copied to the TSEQN sub-field of transport control field. In the transport pipe entry for this service primitive, Active flag shall be set to TRUE; the

retry_count shall be reset to 0; and replyTimer shall be set to max_reply_time and started to await the response.

The TLE shall forward the TPDU to NLE using NL-TRANSMIT request primitive. The TPDU shall also be buffered for possible future retries.

6.6.4.3 Receipt of TL-DATA-EXCHANGE response primitive

The TLE shall search the transport pipe table to find the entry that corresponds to the correspondent address. There may be two such entries for each correspondent address – broadcast and unicast. The TLE shall use the transport-type parameter of the primitive to find the correct transport pipe.

The TLE shall compare the Handle (sequence_number) parameter in the response primitive to the sequence_number in the transport pipe entry. If the sequence_number is not one greater than that in the transport pipe entry, then the service fails and the response shall be discarded. The TLE shall not forward any TPDU to the NLE.

Else, the TLE shall prepare the TPDU, update the sequence_number in the transport pipe entry, encode transport control octet as shown in Table 63. If transport-type is Response-unicast, then the TPDU shall be forwarded to NLE immediately using NL-TRANSMIT request primitive. Else, the transport-type is Response-broadcast; the TLE shall wait for a random amount of time between zero and Bcast_reply_time (see Table 60) value and at the expiration of this time, the TLE shall forward the TPDU to NLE using NL-TRANSMIT request primitive. TLE shall also place the TPDU in a buffer for possible future retry.

Table 63 – Transport control encoding for TL-DATA-EXCHANGE response

Transport-type	tbroad	tack	tresp
Response-unicast	0	1	1
NOTE Other values of Transport-type are used by other services – see Table 61 and Table 62.			

6.6.4.4 Receipt of NL-Transmit indication primitive

6.6.4.4.1 General

When the TLE receives an indication primitive from the NLE, it shall inspect the transport control sub-fields TACK and TRESP, and use it to process the received TPDU.

6.6.4.4.2 Unacknowledged data transfer

If the TACK sub-field of transport control is FALSE, then the TLE shall forward the received TSDU to TLS-user in TL-DATA-TRANSFER indication primitive. The TSEQN sub-field shall be forwarded as sequence-number.

6.6.4.4.3 Acknowledged indication data transfer

If the TACK sub-field of transport control is TRUE and TRESP is FALSE, then the TLE shall validate the TSEQN sub-field. The TLE shall search the transport pipe table to find the entry that corresponds to the correspondent address. There may be two such entries for each correspondent address – broadcast and unicast. The TLE shall use the TBROAD sub-field of transport control to find the correct transport pipe.

If there is no such transport pipe table entry, then the TLE shall create a new entry. The TLE shall set Active attribute to true, Master attribute bit to false and set sequence_number attribute to one less than the TSEQN sub-field in the received TPDU. The TLE shall forward the

TL-DATA-EXCHANGE indication primitive to TLS-user. The transport-type and sequence number in the primitive shall be set to the values received in the transport control field.

If the transport pipe table entry exists, then the TSEQN sub-field in the received TPDU shall be compared to sequence_number in that transport pipe table entry. If both are identical, then the TPDU is a duplicate and the previously buffered response TLDPDU shall be forwarded to NLE in NL-TRANSMIT request primitive. Else, if the TSEQN sub-field in the received TPDU is not one greater than that in the transport pipe table entry then the TPDU shall be discarded.

Else, the TSEQN sub-field in the received TPDU is one greater than that in the transport pipe table entry. The TLE shall forward the received TL-payload as TLS-user-data in the TL-DATA-EXCHANGE indication primitive to TLS-user. The transport-type and sequence number in the primitive shall be set to the values received in the transport control field.

6.6.4.4.4 Acknowledged response data transfer

If the TACK sub-field of transport control is TRUE and TRESP is TRUE, then the TLE shall validate the TSEQN sub-field. The TLE shall search the transport pipe table to find the entry that corresponds to the correspondent address. There may be two such entries for each correspondent address – broadcast and unicast. The TLE shall use the TBROAD sub-field of transport control to find the correct transport pipe.

If there is no such transport pipe table entry, then the TLE shall discard the received TPDU.

If the transport pipe table entry exists, then the TSEQN sub-field in the received TPDU shall be compared to sequence_number in that transport pipe table entry. If both are identical, then the TPDU is a response to previously sent request. The TLE shall set Active attribute of that entry to false, disable replyTimer and delete the buffer that was used to store the TPDU for the request. The TLE shall forward the received TL-payload as TLS-user-data in TL-DATA-EXCHANGE confirm primitive to TLS-user. The Handle in the primitive shall be set to the values received in the corresponding indication primitive.

6.6.4.5 Receipt of NL-TRANSMIT confirm primitive

If the corresponding NL-TRANSMIT request was due to TL-DATA-TRANSFER request, then the TLE shall forward TL-DATA-TRANSFER confirm primitive to the TLE-user. If the corresponding NL-TRANSMIT request was due to TL-DATA-EXCHANGE request with Transport-type value of Request-broadcast, then the TLE shall forward TL-DATA-EXCHANGE confirm primitive to the TLE-user.

6.6.4.6 Expiration of replyTimer

If replyTimer expires it indicates that either the request TPDU or the corresponding response TPDU is lost. If the retry_count is not equal to max_retries, then the TLE shall send the request TPDU and increment retry_count. Else, the retry_count is at its maximum value, the request has failed and the TLE shall set Active attribute of that entry to false, disable replyTimer and delete the buffer that was used to store the TPDU for the request. The TLE shall forward the TL-DATA-EXCHANGE confirm primitive to TLS-user with error status. The Handle in the primitive shall be set to the value received in the corresponding request primitive.

6.6.5 TLE state machines

6.6.5.1 Client data transfer state machine

The states are shown in Figure 37 and the complete state machine is described in Table 64.

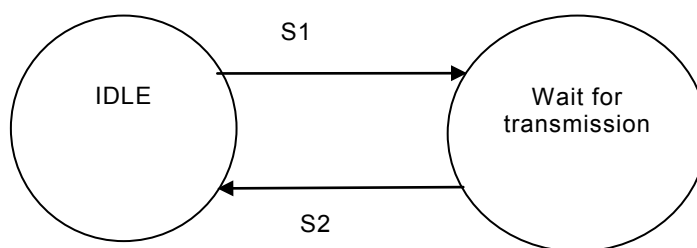


Figure 37 – Transition diagram of the client data transfer state machine

Table 64 – Client data transfer state table

#	Current state	Event or condition => action	Next state
S1	IDLE	TL-DATA-TRANSFER.request => NL-TRANSMIT.request { destination address := TL-DATA-TRANSFER.request.destination address, priority := TL-DATA-TRANSFER.request.priority, Timetable_ID := TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data := TLPDU } (see 6.6.4.1)	Wait for transmission
S2	Wait for transmission	NL-TRANSMIT.confirm => TL-DATA-TRANSFER.confirm { Status := status } (see 6.6.4.5)	IDLE

6.6.5.2 Client data exchange state machine

The states are shown in Figure 38 and the complete state machine is described in Table 65.

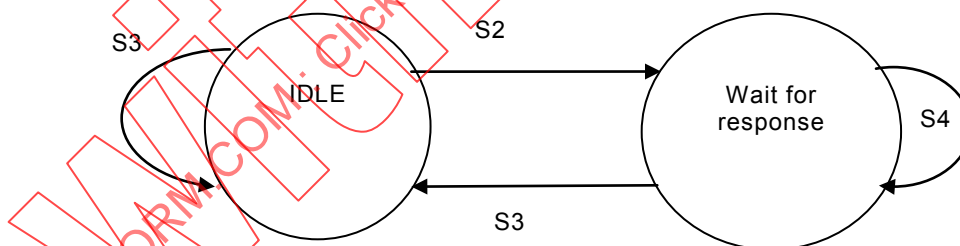


Figure 38 – Transition diagram of the client data exchange state machine

Table 65 – Client data exchange state table

#	Current state	Event or condition => action	Next state
S1	IDLE	TL-DATA-EXCHANGE.request && waiting for acknowledgement for a previous TPDU => TL-DATA-TRANSFER.confirm { Status := failure }	IDLE
S2	IDLE	TL-DATA-EXCHANGE.request && !waiting for acknowledgement for a previous TPDU => retry_count := 0, replyTimer := max_reply_time, NL-TRANSMIT.request { destination address := TL-DATA-TRANSFER.request.destination address, priority := TL-DATA-TRANSFER.request.priority, Timetable_ID := TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data := TLPDU } (see 6.6.4.2)	Wait for response
S3	Wait for response	NL-TRANSMIT.confirm && corresponding Transport-type == Request-broadcast => TL-DATA-EXCHANGE.confirm { Status := status } (see 6.6.4.5)	IDLE
S3	Wait for response	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == TRUE && transport pipe table entry exists && TLPDU.TSEQN == sequence_number in the transport pipe table entry => Disable replyTimer, Delete the buffer that was used to store the TPDU for the request, TL-DATA-EXCHANGE.confirm { NLS-user-data := TLPDU.payload, Status := status } (see 6.6.4.4.4)	IDLE
S4	Wait for response	replyTimer expires && retry_count != max_retries => Increment retry_count, NL-TRANSMIT.request { destination address := TL-DATA-TRANSFER.request.destination address, priority := TL-DATA-TRANSFER.request.priority, Timetable_ID := TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data := TLPDU } (see 6.6.4.6)	Wait for response
S3	Wait for response	replyTimer expires && retry_count == max_retries => TL-DATA-TRANSFER.confirm { Status := failure } (see 6.6.4.6)	IDLE

6.6.5.3 Server state machine

The states are shown in Figure 39 and the complete state machine is described in Table 66.

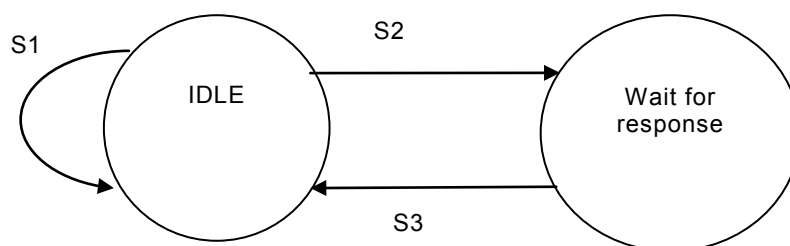
**Figure 39 – Transition diagram of the server state machine**

Table 66 – Server state table

#	Current state	Event or condition => action	Next state
S1	IDLE	TL-DATA-TRANSFER.request => NL-TRANSMIT.request { destination address := TL-DATA-TRANSFER.request.destination address, priority := TL-DATA-TRANSFER.request.priority, Timetable_ID := TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data := TLPDU } (see 6.6.4.1)	IDLE
S1	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == FALSE => TL-DATA-TRANSFER.indication { source address := NL-TRANSMIT.indication.source address, Transport-type := TLPDU.transport-type, priority := TLPDU priority, TLS-user-data := TLPDU payload } (see 6.6.4.4.2)	IDLE
S1	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == FALSE && TLPDU.TSEQN == sequence_number in the transport pipe table entry (duplicate indication, see 6.6.4.4.3) => NL-TRANSMIT.request { destination address := previously buffered destination address, priority := previously buffered priority, Timetable_ID := previously buffered Timetable_ID, NLS-user-data := previously buffered TLPDU } (see 6.6.4.4.3)	IDLE
S2	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == FALSE && transport pipe table entry does not exist => Create transport pipe table entry, sequence_number in the transport pipe table entry = TLPDU.TSEQN -1 TL-DATA-EXCHANGE.indication { source address := NL-TRANSMIT.indication.source address, sequence number := TLPDU.TSEQN, Transport-type := TLPDU.transport-type, priority := TLPDU priority, TLS-user-data := TLPDU payload } (see 6.6.4.4.3)	Wait for response
S2	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == FALSE && transport pipe table entry exists && TLPDU.TSEQN == sequence_number in the transport pipe table entry + 1 => TL-DATA-EXCHANGE.indication { source address := NL-TRANSMIT.indication.source address, sequence number := TLPDU.TSEQN, Transport-type := TLPDU.transport-type, priority := TLPDU priority, TLS-user-data := TLPDU payload } (see 6.6.4.4.3)	Wait for response
S3	Wait for response	TL-DATA-EXCHANGE.response && transport pipe table entry exists && sequence_number in the transport pipe table entry == response sequence number -1 => Increment sequence_number in the transport pipe table entry, NL-TRANSMIT.request { destination address := TL-DATA-EXCHANGE.response.destination address, priority := TL-DATA-EXCHANGE.response.priority, Timetable_ID := TL-DATA-EXCHANGE.response.Timetable_ID, NLS-user-data := TLPDU } (see 6.6.4.3), Buffer TLPDU and NL-TRANSMIT.request parameters	IDLE

6.6.6 TL-management information

Several TL attributes shall be configured before proper TLE operation is possible. These attributes are defined in Table 60.

7 Application layer service definition – Type 20 elements

7.1 Common concepts

The common concepts and templates used to describe the application layer service in this standard are detailed in IEC/TR 61158-1, Clause 9.

7.2 Data type ASE

7.2.1 Overview

7.2.1.1 General

Fieldbus data types specify the machine independent syntax for application data conveyed by FAL services. The Fieldbus application layer supports the definition and transfer of both basic and constructed data types. Encoding rules for the data types specified in this clause are provided in Clause 8.

Basic types are atomic types that cannot be decomposed into more elemental types. Constructed types are types composed of basic types and other constructed types. Their complexity and depth of nesting is not constrained by this standard.

Data types are defined as instances of the data type class, as shown in Figure 40.

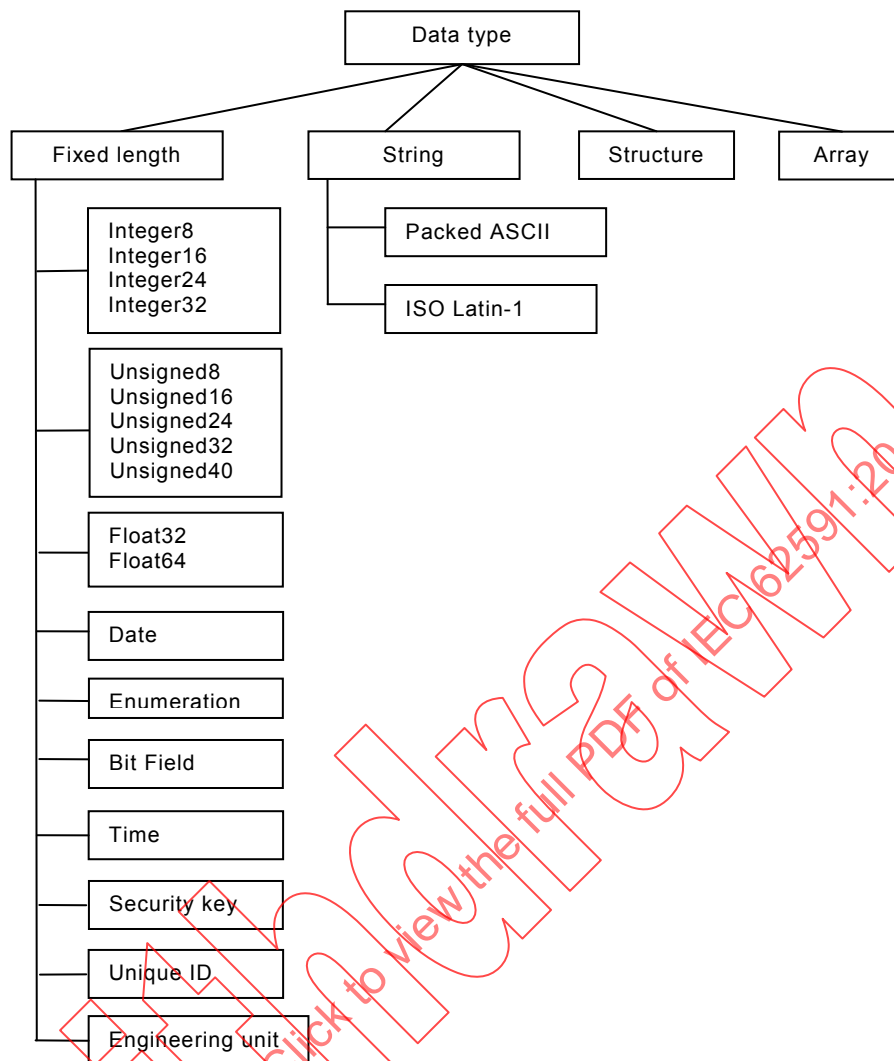


Figure 40 – Data type class hierarchy

The data type definitions are represented as a class/format/instance structure beginning with data type class entitled "Data type". The formats for data types are defined by the data type class.

The basic data classes are used to define fixed length data types. Standard types taken from ISO/IEC 8824 are referred to as *simple* data types. Other standard basic data types are defined specifically for Fieldbus applications and are referred to as specific types.

The constructed types specified in this standard are strings, arrays and structures.

7.2.1.2 Overview of basic types

Most basic types are defined from a set of ISO/IEC 8824 types (simple types). Some ISO/IEC 8824 types have been extended for Fieldbus specific use (specific types).

Simple types are ISO/IEC 8824 universal types. They are defined in this standard to provide them with Fieldbus class identifiers.

Specific types are basic types defined specifically for use in the Fieldbus environment. They are defined as simple class subtypes.

Basic types have a constant length. Two variations are defined, one for defining data types whose length is an integral number of octets, and one for defining data types whose length is bits.

NOTE Integer, Packed ASCII, ISO Latin-1 and Date are defined in this standard for the purpose of assigning Fieldbus class identifiers to them. This standard does not change their definitions as specified in ISO/IEC 8824.

7.2.1.3 Fixed length types

The length of fixed length types is an integral number of octets.

7.2.1.4 Constructed types

7.2.1.4.1 Overview

Constructed data types are needed to completely convey the variety of information present on the Fieldbus. There are three kinds of constructed types defined for this standard – string, array and structure.

7.2.1.4.2 String

A string is composed of an ordered set, variable in number of homogeneously typed fixed-length elements.

7.2.1.4.3 Structure

A structure is made of an ordered set of heterogeneously typed elements called fields. This standard does not restrict the data type of fields. However, the fields within a structure do not have to be of the same type.

7.2.1.4.4 Array

An array is composed of an ordered set of homogeneously typed elements. The data type of array elements can be fixed length basic type or structure. All elements of an array shall be of the same type.

7.2.1.4.5 Nesting level

This standard permits structures and arrays to contain structures and arrays.

7.2.1.5 Specification of user defined data types

User defined types are not supported by this standard.

7.2.1.6 Transfer of user data

User data is transferred between applications by the FAL protocol. All encoding and decoding are performed by the FAL user.

The rules for encoding user data in FAL protocol data units is data type dependent. These rules are defined in IEC 61158-6-20.

7.2.2 Formal definition of data type objects

7.2.2.1 Data type class

The data type class specifies the root of the data type class tree. Its parent class "top" indicates the top of the FAL class tree.

FAL ASE: Data type ASE

CLASS: Data type

CLASS ID: Not used

PARENT CLASS: TOP

ATTRIBUTES:

- 1 (m) Key Attribute: Data type Name
- 2 (m) Attribute: Format (FIXED LENGTH, STRING, STRUCTURE, ARRAY)
- 3 (c) Constraint: Format = FIXED LENGTH | STRING
- 3.1 (m) Attribute: Octet Length
- 4 (c) Constraint: Format = STRUCTURE
- 4.1 (m) Attribute: Number of Fields
- 4.2 (m) Attribute: List of Fields
- 4.2.1 (m) Attribute: Field Name
- 4.2.2 (m) Attribute: Field Data type
- 5 (c) Constraint: Format = ARRAY
- 5.1 (m) Attribute: Number of Array Elements
- 5.2 (m) Attribute: Array Element Data type

7.2.2.2 Attributes

Format

This attribute identifies the data type as a fixed-length, string, structure or array.

Octet Length

This conditional attribute defines the representation of the dimensions of the associated type object. It is present when the value of the format attribute is "FIXED LENGTH" or "STRING". For FIXED LENGTH data types, it represents the length in octets. For STRING data types, it represents the length in octets for a single element of a string.

Number of Fields

This conditional attribute defines the number of fields in a structure. It is present when the value of the format attribute is "STRUCTURE".

List of Fields

This conditional attribute is an ordered list of fields contained in the structure. Each field is specified by its type. Fields are described in the order in which they occur. Partial access to fields within a structure is not supported.

Field Name

This conditional attribute specifies the name of the field. It is present when the value of the format attribute is "STRUCTURE".

Field Data type

This conditional attribute specifies the data type of the field. It is present when the value of the format attribute is "STRUCTURE". This attribute may itself specify a constructed data type by referencing a constructed data type definition by embedding a constructed data type definition here.

Number of Array Elements

This conditional attribute defines the number of elements for the array type. This attribute is present when the value of the format attribute is "ARRAY".

Array Element Data type

This conditional attribute specifies the data type for the elements of an array. All elements of the array have the same data type. It is present when the value of the format attribute is "ARRAY". This attribute may itself specify a constructed data type either by referencing a constructed data type definition by its numeric id.

7.2.3 FAL defined data types

7.2.3.1 Fixed length types

7.2.3.1.1 Integer8

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Integer8
2 Format	= FIXED LENGTH
2.1 Octet Length	= 1

This integer type is a two's complement binary number with a length of one octet.

7.2.3.1.2 Integer16

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Integer16
2 Format	= FIXED LENGTH
2.1 Octet Length	= 2

This integer type is a two's complement binary number with a length of two octets.

7.2.3.1.3 Integer24

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Integer24
2 Format	= FIXED LENGTH
2.1 Octet Length	= 3

This integer type is a two's complement binary number with a length of three octets.

7.2.3.1.4 Integer32

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Integer32
2 Format	= FIXED LENGTH
2.1 Octet Length	= 4

This integer type is a two's complement binary number with a length of four octets.

7.2.3.1.5 Unsigned8

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Unsigned8
2 Format	= FIXED LENGTH
2.1 Octet Length	= 1

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This type has a length of one octet.

7.2.3.1.6 Unsigned16

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Unsigned16
2 Format	= FIXED LENGTH

2.1 Octet Length = 2

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of two octets.

7.2.3.1.7 Unsigned24

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Unsigned24
- 2 Format = FIXED LENGTH
- 2.1 Octet Length = 3

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of three octets.

7.2.3.1.8 Unsigned32

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Unsigned32
- 2 Format = FIXED LENGTH
- 2.1 Octet Length = 4

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of four octets.

7.2.3.1.9 Unsigned40

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Unsigned40
- 2 Format = FIXED LENGTH
- 2.1 Octet Length = 5

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of five octets.

7.2.3.1.10 Float32

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Float32
- 2 Format = FIXED LENGTH
- 2.1 Octet Length = 4

This type has a length of four octets. The format for Float32 is that defined by ANSI/IEEE 754 as single precision.

7.2.3.1.11 Float64

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Float64
- 2 Format = FIXED LENGTH
- 2.1 Octet Length = 8

This type has a length of eight octets. The format for Float64 is that defined by ANSI/IEEE 754 as double precision.

7.2.3.1.12 Date

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Date
2 Format	= FIXED LENGTH
2.1 Octet Length	= 3

This data type is consists of day, month and year minus 1900. This allows the representation of any date between 1 January 1900 and 31 December 2155.

7.2.3.1.13 Enumeration

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Enumeration
2 Format	= FIXED LENGTH
2.1 Octet Length	= 1

Data items that take on a single meaning from a list or table are encoded as Enumeration. This data type uses an unsigned integer of one octet length. The largest integer value is reserved and not used by any service.

7.2.3.1.14 Bit Field

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Bit Field
2 Format	= FIXED LENGTH
2.1 Octet Length	= 1 or 2

This data type is a series of eight bits numbered from 0 to 7 or sixteen bits numbered 0 to 15. Communication of information encoded as single-bit data (such as status and diagnostic information) uses this data type.

7.2.3.1.15 Time

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Time
2 Format	= FIXED LENGTH
2.1 Octet Length	= 4

This data type is an unsigned binary integer and represents time in the increments of 1/32 of a millisecond. If this data type is used to represent time of day, then it indicates number of 1/32 of milliseconds since midnight.

7.2.3.1.16 Security key

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Security key
2 Format	= FIXED LENGTH
2.1 Octet Length	= 16

This type is a string of sixteen octets. Each octet can have any value from 0 to 255. The interpretation of this data type is application dependent.

7.2.3.1.17 Unique ID

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Unique ID
2	Format	= FIXED LENGTH
2.1	Octet Length	= 5

This type is a string of five octets. Each octet can have any value from 0 to 255. The interpretation of this data type is application dependent.

7.2.3.1.18 Engineering unit

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Engineering unit
2	Format	= FIXED LENGTH
2.1	Octet Length	= 1

This type defines the measurement unit of a measured variable. The interpretation of this data type is specified by the communication profile (see 9.3.1) and beyond the scope of this standard.

7.2.3.2 String Types

7.2.3.2.1 Packed ASCII

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Packed ASCII
2	Format	= STRING
2.1	Octet Length	= 1 to n

This type is a modified subset of the ASCII character code set. This subset is shown in the Table 340. It is produced by compressing four Packed ASCII characters into three 8-bit octets,

7.2.3.2.2 ISO Latin-1

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Latin-1
2	Format	= STRING
2.1	Octet Length	= 1 to n

This type is a subset of the ISO 8859-1 (ISO Latin-1) string type. The subset is shown in the Table 341.

7.2.4 Data type ASE service specification

There are no operational services defined for the data type object.

7.2.5 Summary of data types

This clause contains a summary of the defined data types as shown in Table 67.

Table 67 – Data type summary

Data type	Clause
Integer8	7.2.3.1.1
Integer16	7.2.3.1.2
Integer24	7.2.3.1.3
Integer32	7.2.3.1.4
Unsigned8	7.2.3.1.5
Unsigned16	7.2.3.1.6
Unsigned24	7.2.3.1.7
Unsigned32	7.2.3.1.8
Unsigned40	7.2.3.1.9
Float32	7.2.3.1.10
Float64	7.2.3.1.11
Date	7.2.3.1.12
Enumeration	7.2.3.1.13
Bit Field	7.2.3.1.14
Time	7.2.3.1.15
Security Key	7.2.3.1.16
Unique ID	7.2.3.1.17
Engineering unit	7.2.3.1.18
Packed ASCII	7.2.3.2.1
ISO Latin-1	7.2.3.2.2

7.3 Communication model specification

7.3.1 Common parameters

Many services have the following parameters. Instead of defining them with each service, the following common definitions are provided.

7.3.1.1 AREP ID

This parameter specifies sufficient information to identify the AREP of the remote end of the AR. The value of this parameter can be the Unique ID of the remote end. One value of this parameter is reserved as broadcast address.

7.3.1.2 Response Code

If there is no communication error then this parameter specifies a command completion report indicating the status of the command's execution by the device. The possible values of this parameter are shown in the Table 68. Its data type is Enumeration.

Table 68 – Response code values

Value	Description
Success	Command (read or Write) was executed properly
Warning	Command (Write) was executed with the deviation as described in response (for example a value was set to its nearest legal value)
Error	Command (read or Write) was not executed properly. Response Code indicates the reason (for example the device is in Write Protect mode)

7.3.1.3 Application process status

This parameter indicates the status provided by the FAL user and it is not associated with the completion of any command. Its length is one octet.

7.3.1.4 Extended status

This parameter indicates the VFD status. It is not associated with the completion of any command. Its length is one octet.

7.3.2 ASEs

7.3.2.1 Virtual field device ASE

The virtual field device (VFD) ASE shall be according to IEC 61158-5-20:2010, 6.2.1.

7.3.2.2 Variable ASE

7.3.2.2.1 Overview

The Variable ASE provides services to read or write a variable object in Server device. The variable is identified by a number. This numeric identifier completely describes the data type and structure of the object. The object can be any of the types defined in 7.2.3.

7.3.2.2.2 Variable model – common

7.3.2.2.2.1 Simple variable

7.3.2.2.2.1.1 Formal model

The simple variable object represents a single, simple variable which is characterized by a defined Data type.

ASE:	VARIABLE ASE
CLASS:	Simple VARIABLE
PARENT CLASS:	TOP
ATTRIBUTES:	
1 (m) Key Attribute:	Numeric Identifier
2 (o) Attribute:	Variable Name
3 (m) Attribute:	Data type Name
SERVICES:	
1 (m) OpService:	Read
2 (o) OpService:	Write
3 (o) OpService:	Information report

7.3.2.2.2.1.2 Attributes

Numeric Identifier

Identifies an instance of this object class.

Variable Name

It is the name assigned to the object.

Data type Name

It is the name assigned to the data type as specified in 7.2.3.1.

7.3.2.2.1.3 Services**Read**

This confirmed service permits a client to read the value of a variable.

Write

This confirmed service permits a client to write the value of a variable.

Information report

This unconfirmed service permits a server to send value of a variable to any one or all clients.

7.3.2.2.2 Structure variable**7.3.2.2.2.1 General**

The record object consists of a collection of simple variables or an array of Structure variables of different data types. The record object is accessed completely and its individual fields cannot be accessed separately.

ASE:		VARIABLE ASE
CLASS:		Structure VARIABLE
PARENT CLASS:		TOP
ATTRIBUTES:		
1	(m) Key Attribute:	Numeric Identifier
2	(o) Attribute:	Variable Name
3	(m) Attribute:	Number of Fields
4	(m) Attribute:	List of Fields
4.1	(o) Attribute:	Field Name
4.2	(m) Attribute:	Field Data type Name
SERVICES:		
1	(m) OpsService:	Read
2	(o) OpsService:	Write
3	(o) OpsService:	Information report

7.3.2.2.2.2 Attributes**Numeric Identifier**

Identifies an instance of this object class.

Variable Name

It is the name assigned to the object.

Number of Fields

This attribute defines the number of fields in a structure.

Field Name

This optional attribute specifies the name of the field.

Field Data type Name

It is the name assigned to the data type as specified in 7.2.3.1.

7.3.2.2.2.3 Services**Read**

This confirmed service permits a client to read the value of a variable.

Write

This confirmed service permits a client to read the value of a variable.

Information report

This unconfirmed service permits a server to send value of a variable to any one or all clients.

7.3.2.2.2.3 Array of simple variable

7.3.2.2.2.3.1 Formal model

This Array Object is used to define a constructed variable in which all elements have the identical Data type and length and are of Simple Variable class. One or more elements of this object type can be accessed at a time by specifying the starting subindex and the number of elements to be accessed.

ASE:	VARIABLE ASE
CLASS:	Array of Simple VARIABLE
PARENT CLASS:	Simple variable
ATTRIBUTES:	
ATTRIBUTES:	
4 (m) Attribute:	Array size

7.3.2.2.2.3.2 Attributes

Array size

It states how many elements are contained in the Array. If the value of this attribute is "Variable" then the number of elements is not fixed.

7.3.2.2.2.4 Array of structure variable

7.3.2.2.2.4.1 Formal model

This Array Object is used to define a constructed variable in which all elements have the identical Data type and length and are of Structure Variable class. One or more elements of this object type can be accessed at a time by specifying the starting subindex and the number of elements to be accessed.

ASE:	VARIABLE ASE
CLASS:	Array of Structure VARIABLE
PARENT CLASS:	Structure variable
ATTRIBUTES:	
ATTRIBUTES:	
5 (m) Attribute:	Array size

7.3.2.2.2.4.2 Attributes

Array size

It states how many elements are contained in the Array. If the value of this attribute is "Variable" then the number of elements is not fixed.

7.3.2.2.3 Variable ASE service specification

7.3.2.2.3.1 Supported services

This subclause contains the definition of services that are unique to this ASE. The services defined for this ASE are:

- Read variable from Server device,

- Write variable to Server device, and
- Send Information report from a server device.

There are separate services for each variable, which is identified by a number. The common format of Read service is defined in 7.3.2.2.3.3. The common format of Write service is defined in 7.3.2.2.3.4. The common format of Information report service is defined in 7.3.2.2.3.5.

For every variable there is a separate read and write service. These services are defined in 7.3.2.4, 7.3.2.5, 7.3.2.6 and 7.3.2.7.

NOTE The service definitions in this subclause are not independent of the variable being accessed. This subclause only shows the format of the service. The service specific definitions include the variable model and the service specific parameters.

7.3.2.2.3.2 Primitive correlation

At the client, there is only one outstanding request for one value of AREP ID. The AREP ID is conveyed as destination address in the transport layer service primitives. Therefore, the confirm primitive is correlated with the request primitive using AREP ID. At the server, there is only one outstanding indication primitive at any time. Therefore, the response primitive is correlated with the indication primitive waiting for the response.

7.3.2.2.3.3 Read service – common format

7.3.2.2.3.3.1 Service overview

This service permits a client to read the value of a variable from the server.

7.3.2.2.3.3.2 Service primitives

The service parameters for this service are shown in Table 69.

Table 69 – Read service parameters

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP ID	M	M (=)		
Numeric Identifier	M	M (=)	M (=)	M (=)
Additional Identifier	C	M (=)	M (=)	M (=)
Subindex	C	M (=)	M (=)	M (=)
Element count	C	M (=)	CM	M (=)
Array size			C	M(=)
Result (+)				
Variable value			S	S (=)
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)
Result (–)			S	S (=)
Status				S
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)

Argument

The argument contains the parameters of the service request.

Numeric Identifier

This parameter identifies the variable. This is the key attribute assigned to the variable to be read. It is also called command number.

Additional identifier

For some of the Numeric Identifiers, this parameter provides additional information required to identify the variable(s) to be read.

Subindex

This parameter identifies the individual element in an array variable by its position within the variable. This can be either a numeric value or an Enumeration.

Element count

This parameter specifies the number of array elements to be read. This parameter is present only if the variable is an array variable and if the Subindex is a numeric value. The default value of this parameter is one (1). In the request and indication primitives, its value is the desired number of the elements to be read. In the response and confirm primitives its value is the actual number of elements read.

Array size

This parameter specifies the total number of array elements available. This parameter present only if the variable is an array variable.

Variable value

This parameter specifies the value read from the server.

Status

This parameter specifies the local reason for failure such as communication timeout. If the server sends a response, but it could not send the "Variable value, then it is shown in "Response code".

7.3.2.2.3.4 Write service – common format

7.3.2.2.3.4.1.1 Service overview

This service permits a client to write the value of a variable to the server.

7.3.2.2.3.4.1.2 Service primitives

The service parameters for this service are shown in Table 70.

Table 70 – Write service parameters

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP ID	M	M (=)		
Numeric Identifier	M	M (=)	M (=)	M (=)
Additional identifier	C	M (=)	M (=)	M (=)
Subindex	C	M (=)	M (=)	M (=)
Variable value	M	M (=)		
Result (+)				
Variable value			S	S (=)
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)
Result (–)				
Status				S
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)

Argument

The argument contains the parameters of the service request.

Numeric Identifier

This parameter identifies the variable. This is the key attribute assigned to the variable to be read. It is also called command number.

Additional identifier

For some of the Numeric Identifiers, this parameter provides additional information required to identify the variable(s) to be read.

Subindex

This parameter identifies the individual element in an array variable by its position within the variable. This can be either a numeric value or an Enumeration.

Variable value

In the request primitive, this parameter specifies the desired value of the variable to be written. In the response primitive, this parameter states the actual value of the variable that was written. In some cases, the responding server writes a value different than the value in indication primitive, for example it may round the value to the nearest allowed value or the nearest value the server can support.

NOTE The returned variable value can be used by the FAL-user to validate the write operation.

Status

This parameter specifies the local reason for failure such as communication timeout. If the server sends a response, but it could not write the “Variable value, then it is shown in “Response code”.

7.3.2.2.3.5 Information report service – common format

7.3.2.2.3.5.1 Service overview

This service permits a master or slave to send the value of a variable. It is used to publish variable data, to send event information or to acknowledge event.

7.3.2.2.3.5.2 Service primitives

The service parameters for this service are shown in Table 71.

Table 71 – Information report service parameters

Parameter name	Req	Ind
Argument		
AREP ID	M	M (=)
Numeric Identifier	M	M (=)
Subindex	C	M (=)
Variable value	M	M (=)
Report Code	M	M (=)
Application process status	M	M (=)
Extended status	M	M (=)

Argument

The argument contains the parameters of the service request.

Numeric Identifier

This parameter identifies the variable. This is the key attribute assigned to the variable to be read. It is also called command number.

Subindex

This parameter identifies the individual element in an array variable by its position within the variable. This can be either a numeric value or an Enumeration.

Variable value

This parameter specifies the value reported from the server.

7.3.2.3 Action ASE

7.3.2.3.1 Service overview

This service permits a client to command a specified action of the server. The common format of Action service is defined in this clause. For every action, there is a separate service.

NOTE The service definitions in this subclause are not independent of the action being requested. This subclause only shows the format of the service. The service specific definitions include the action behavior and the service specific parameters.

7.3.2.3.2 Primitive correlation

At the client, there is only one outstanding request for one value of AREP ID. The AREP ID is conveyed as destination address in the transport layer service primitives. Therefore, the confirm primitive is correlated with the request primitive using AREP ID. At the server, there is only one outstanding indication primitive at any time. Therefore, the response primitive is correlated with the indication primitive waiting for the response.

7.3.2.3.3 Action model

7.3.2.3.3.1 General

Some of the actions require a variable to validate the action. The model for such actions is defined in this subclause.

7.3.2.3.3.2 Formal model

ASE: Simple Action
PARENT CLASS: TOP
ATTRIBUTES:
 1 (m) Key Attribute: Numeric Identifier

ASE: Qualified Action
PARENT CLASS: TOP
ATTRIBUTES:
 1 (m) Key Attribute: Numeric Identifier
 2 (m) Attribute: Qualifier Name
 3 (m) Attribute: Data type Name

ASE: Simple Variable Action
PARENT CLASS: TOP
ATTRIBUTES:
 1 (m) Key Attribute: Numeric Identifier
 2 (m) Attribute: Variable Name
 3 (m) Attribute: Data type Name

ASE: Structure Variable Action
PARENT CLASS: TOP
ATTRIBUTES:
 1 (m) Key Attribute: Numeric Identifier
 2 (m) Attribute: Variable Name
 3 (m) Attribute: Number of Fields
 4 (m) Attribute: List of Fields
 4.1 (o) Attribute: Field Name
 4.2 (m) Attribute: Field Data type Name

7.3.2.3.3.3 Attributes

See 7.3.2.2.2 for the definition of attributes.

7.3.2.3.4 Action service primitives – common format

The service parameters for this service are shown in Table 72.

Table 72 – Action service parameters

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP ID	M	M (=)		
Numeric Identifier	M	M (=)	M (=)	M (=)
Action qualifier	C	M (=)	M (=)	M (=)
Subindex	C	M (=)	M (=)	M (=)
Variable value	C	M (=)		
Result (+)				
Variable value			S	S (=)
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)
Result (–)				
Status				S
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)

Argument

The argument contains the parameters of the service request.

Numeric identifier

This parameter identifies the action. It is also called command number.

Action qualifier

For some of the Numeric Identifiers, this parameter provides additional information required to identify action to be performed. If the action requires a variable and if the Numeric identifier is not sufficient, then this parameter identifies that variable.

Subindex

If the action requires an array variable, then this parameter identifies the individual element in that array variable. This parameter can be either a numeric value of an Enumeration.

NOTE This parameter is present if the action is to add an element in the array or delete an element from the array.

Variable value

If the action requires a variable, then this parameter specifies the value of the variable. If the action is “add” then in the request parameter, this parameter specifies the desired value of the variable to be added; in the response primitive, this parameter states the actual value of the variable that was added.

Status

This parameter specifies the local reason for failure such as communication timeout. If the server sends a response, but it could not complete the action successfully, then it is shown in “Response code”.

7.3.2.4 Device application services

7.3.2.4.1 Read primary variable

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.4.

7.3.2.4.2 Read loop current and percent of range

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.5.

7.3.2.4.3 Read dynamic variables and loop current

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.6.

7.3.2.4.4 Write loop configuration

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.7.

7.3.2.4.5 Read loop configuration

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.8.

7.3.2.4.6 Read dynamic variable families classifications

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.9.

7.3.2.4.7 Read device variables with status

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.10.

7.3.2.4.8 Read message

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.11.

7.3.2.4.9 Read tag, descriptor and date

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.12.

7.3.2.4.10 Read primary variable transducer information

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.13.

7.3.2.4.11 Read device information

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.14.

7.3.2.4.12 Read final assembly number

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.15.

7.3.2.4.13 Write message

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.16.

7.3.2.4.14 Write tag, descriptor and date

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.17.

7.3.2.4.15 Write final assembly number

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.18.

7.3.2.4.16 Read long_tag

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.19.

7.3.2.4.17 Write long_tag

This service is specified in IEC 61158-5-20:2010, 6.2.2.3.20.

7.3.2.4.18 Reset configuration changed flag

7.3.2.4.18.1 Service overview

This service resets the “Configuration changed” bit associated with the client issuing this write service.

7.3.2.4.18.2 Request parameters

CLASS: Simple Variable Action

ATTRIBUTES:

1	Numeric Identifier	= 38
2	Variable Name	= Configuration change counter
3	Data type Name	= Unsigned16

7.3.2.4.19 Perform self test

7.3.2.4.19.1 Service overview

This service initiates the self test function in the device.

7.3.2.4.19.2 Request parameters

CLASS: Simple Action

ATTRIBUTES:

1	Numeric Identifier	= 41

7.3.2.4.20 Perform device reset

7.3.2.4.20.1 Service overview

This service initiates the self test function in the device.

7.3.2.4.20.2 Request parameters

CLASS: Simple Action

ATTRIBUTES:

1	Numeric Identifier	= 42

7.3.2.4.21 Read additional device status

7.3.2.4.21.1 Service overview

This service reads device status information not included in the Response code or Device status octet.

7.3.2.4.21.2 Variable model

There are three different Variable models depending upon the type of device. The device responds with only one of these three types of variables. If the device supports more than one Analog channel, then it supports “Additional device status 2”. Some of the devices may use “Additional device status 3”. All other devices support “Additional device status 1”.

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 48
- 2 Variable name = Additional device status 1
- 3 Number of fields = variable
- 4 List of Fields
 - 4.1 (m) Field name = Device specific status 0
 - 4.2 (m) Field data type name = Bit Field
 - 4.3 (m) Field name = Extended status
 - 4.4 (m) Field data type name = Bit Field
 - 4.5 (m) Field name = Device operating mode
 - 4.6 (m) Field data type name = Bit Field
 - 4.7 (m) Field name = Standardized status 0
 - 4.8 (m) Field data type name = Bit Field

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 5 Numeric Identifier = 48
- 6 Variable name = Additional device status 2
- 7 Number of fields = variable
- 8 List of Fields
 - 8.1 (m) Field name = Device specific status 0
 - 8.2 (m) Field data type name = Bit Field
 - 8.3 (m) Field name = Extended status
 - 8.4 (m) Field data type name = Bit Field
 - 8.5 (m) Field name = Device operating mode
 - 8.6 (m) Field data type name = Bit Field
 - 8.7 (m) Field name = Standardized status 0
 - 8.8 (m) Field data type name = Bit Field
 - 8.9 (m) Field name = Standardized status 1
 - 8.10 (m) Field data type name = Bit Field
 - 8.11 (m) Field name = Analog channel saturated
 - 8.12 (m) Field data type name = Bit Field
 - 8.13 (m) Field name = Standardized status 2
 - 8.14 (m) Field data type name = Bit Field
 - 8.15 (m) Field name = Standardized status 3
 - 8.16 (m) Field data type name = Bit Field
 - 8.17 (m) Field name = Analog channel fixed
 - 8.18 (m) Field data type name = Bit Field

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 9 Numeric Identifier = 48
- 10 Variable name = Additional device status 3
- 11 Number of fields = variable
- 12 List of Fields
 - 12.1 (m) Field name = Device specific status 0
 - 12.2 (m) Field data type name = Bit Field
 - 12.3 (m) Field name = Extended status
 - 12.4 (m) Field data type name = Bit Field
 - 12.5 (m) Field name = Device operating mode
 - 12.6 (m) Field data type name = Bit Field
 - 12.7 (m) Field name = Standardized status 0
 - 12.8 (m) Field data type name = Bit Field
 - 12.9 (m) Field name = Standardized status 1

- 12.10 (m) Field data type name = Bit Field
- 12.11 (m) Field name = Analog channel saturated
- 12.12 (m) Field data type name = Bit Field
- 12.13 (m) Field name = Standardized status 2
- 12.14 (m) Field data type name = Bit Field
- 12.15 (m) Field name = Standardized status 3
- 12.16 (m) Field data type name = Bit Field
- 12.17 (m) Field name = Analog channel fixed
- 12.18 (m) Field data type name = Bit Field
- 12.19 (m) Field name = Device specific status 1
- 12.20 (m) Field Data type name = Array of Status

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Status
- 2 Format = ARRAY
- 2.1 Number of Array Elements = 1 to 11
- 2.2 Array Element Data type = Bit Field

7.3.2.4.22 Reset more status available

7.3.2.4.22.1 Service overview

This service resets the “More status available” bit corresponding to the requesting device (primary master, secondary master or gateway). The number of fields in the associated structure variable depends upon the server, and it is the minimum of 4 and the maximum of 10.

7.3.2.4.22.2 Request parameters

CLASS: Structure Variable Action

ATTRIBUTES:

- 1 Numeric Identifier = 48
- 2 Variable Name = Additional device status
- 3 Number of fields = variable
- 4 List of Fields
 - 4.1 (m) Field name = Device specific status 0
 - 4.2 (m) Field data type name = Bit Field
 - 4.3 (m) Field name = Extended status
 - 4.4 (m) Field data type name = Bit Field
 - 4.5 (m) Field name = Device operating mode
 - 4.6 (m) Field data type name = Bit Field
 - 4.7 (m) Field name = Standardized status 0
 - 4.8 (m) Field data type name = Bit Field
 - 4.9 (m) Field name = Standardized status 1
 - 4.10 (m) Field data type name = Bit Field
 - 4.11 (m) Field name = Analog channel saturated
 - 4.12 (m) Field data type name = Bit Field
 - 4.13 (m) Field name = Standardized status 2
 - 4.14 (m) Field data type name = Bit Field
 - 4.15 (m) Field name = Standardized status 3
 - 4.16 (m) Field data type name = Bit Field
 - 4.17 (m) Field name = Analog channel fixed
 - 4.18 (m) Field data type name = Bit Field
 - 4.19 (m) Field name = Device specific status 1

4.20 (m) Data type = Array of Status

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Status
- 2 Format = ARRAY
- 2.1 Number of Array Elements = 1 to 11
- 2.2 Array Element Data type = Bit Field

7.3.2.4.23 Read device variable information

7.3.2.4.23.1 Service overview

This service reads device variable related parameters.

7.3.2.4.23.2 Request parameters

- 1 Numeric Identifier = 54
- 2 Subindex
- 2.1 Subindex Name = Variable code
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.23.3 Response parameters

- 1 Numeric Identifier = 54
- 2 Subindex
- 2.1 Subindex Name = Variable code
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.23.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 54
- 2 Variable Name = Device variable information
- 3 Number of Fields = 9
- 4 List of Fields
 - 4.1 Field Name = Transducer serial number
 - 4.2 Field Data type Name = Unsigned24
 - 4.3 Field Name = Transducer unit
 - 4.4 Field Data type Name = Enumeration
 - 4.5 Field Name = Upper transducer limit
 - 4.6 Field Data type Name = Float32
 - 4.7 Field Name = Lower transducer limit
 - 4.8 Field Data type Name = Float32
 - 4.9 Field Name = Damping value
 - 4.10 Field Data type Name = Float32
 - 4.11 Field Name = Minimum span
 - 4.12 Field Data type Name = Float32
 - 4.13 Field Name = Device variable classification
 - 4.14 Field Data type Name = Enumeration
 - 4.15 Field Name = Device variable family
 - 4.16 Field Data type Name = Enumeration
 - 4.17 Field Name = Update period
 - 4.18 Field Data type Name = Time

5 Array size = Variable

7.3.2.4.24 Write device variable

7.3.2.4.24.1 Service overview

This service allows a Device variable to be temporarily forced to a fixed value.

7.3.2.4.24.2 Request parameters

1 Numeric Identifier = 79
 2 Subindex
 2.1 Subindex Name = Variable code
 2.2 Subindex Data type = Unsigned8

7.3.2.4.24.3 Response parameters

1 Numeric Identifier = 79
 2 Subindex
 2.1 Subindex Name = Variable code
 2.2 Subindex Data type = Unsigned8

7.3.2.4.24.4 Variable model

CLASS: **Array of Structure VARIABLE**

ATTRIBUTES:

1 Numeric Identifier = 79
 2 Variable Name = Device variable record
 3 Number of Fields = 4
 4 List of Fields
 4.1 Field Name = Device variable command
 4.2 Field Data type Name = Enumeration
 4.3 Field Name = Device variable unit
 4.4 Field Data type Name = Engineering unit
 4.5 Field Name = Device variable value
 4.6 Field Data type Name = Float32
 4.7 Field Name = Device variable status
 4.8 Field Data type Name = Bit Field
 5 Array size = Variable

7.3.2.4.25 Read real-time clock

7.3.2.4.25.1 Service overview

This service reads the Real-time clock including the current time as estimated by the device and the last time the clock was set.

7.3.2.4.25.2 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1 Numeric Identifier = 90
 2 Variable Name = Real-time clock
 3 Number of Fields = 5
 4 List of Fields
 4.1 Field Name = Date
 4.2 Field Data type Name = Date

- 4.3 Field Name = Current time
- 4.4 Field Data type Name = Time
- 4.5 Field Name = Date last set
- 4.6 Field Data type Name = Date
- 4.7 Field Name = Time last set
- 4.8 Field Data type Name = Time
- 4.9 Field Name = RTC flags
- 4.10 Field Data type Name = Bit Field

7.3.2.4.26 Write publish data period

7.3.2.4.26.1 Service overview

This service selects the minimum and maximum update period of a publish data message.

7.3.2.4.26.2 Request parameters

- 1 Numeric Identifier = 103
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.26.3 Response parameters

- 1 Numeric Identifier = 103
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.26.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 103
- 2 Variable Name = Publish data period
- 3 Number of Fields = 2
- 4 List of Fields
- 4.1 Field Name = Update period
- 4.2 Field Data type Name = Time
- 4.3 Field Name = Maximum update period
- 4.4 Field Data type Name = Time
- 5 Array size = Variable

7.3.2.4.27 Write publish data trigger

7.3.2.4.27.1 Service overview

This service configures the trigger that forces publishing of the Publish data message.

7.3.2.4.27.2 Request parameters

- 1 Numeric Identifier = 104
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.27.3 Response parameters

1	Numeric Identifier	= 104
2	Subindex	
2.1	Subindex Name	= Publish data message ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.27.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 104
2	Variable Name	= Publish data trigger
3	Number of Fields	= 4
4	List of Fields	
4.1	Field Name	= Trigger mode
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Device variable classification
4.4	Field Data type Name	= Enumeration
4.5	Field Name	= Device variable unit
4.6	Field Data type Name	= Engineering unit
4.7	Field Name	= Trigger level
4.8	Field Data type Name	= Float32
5	Array size	= Variable

7.3.2.4.28 Read publish data mode configuration

7.3.2.4.28.1 Service overview

This service reads the publish data mode configuration.

7.3.2.4.28.2 Request parameters

1	Numeric Identifier	= 105
2	Subindex	
2.1	Subindex Name	= Publish data message ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.28.3 Response parameters

1	Numeric Identifier	= 105
2	Subindex	
2.1	Subindex Name	= Publish data message ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.28.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 105
2	Variable Name	= Publish data mode configuration
3	Number of Fields	= 18
4	List of Fields	
4.1	Field Name	= Publish data mode control
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Command number expansion

4.4	Field Data type Name	= Unsigned8
4.5	Field Name	= Device variable code 0
4.6	Field Data type Name	= Unsigned8
4.7	Field Name	= Device variable code 1
4.8	Field Data type Name	= Unsigned8
4.9	Field Name	= Device variable code 2
4.10	Field Data type Name	= Unsigned8
4.11	Field Name	= Device variable code 3
4.12	Field Data type Name	= Unsigned8
4.13	Field Name	= Device variable code 4
4.14	Field Data type Name	= Unsigned8
4.15	Field Name	= Device variable code 5
4.16	Field Data type Name	= Unsigned8
4.17	Field Name	= Device variable code 6
4.18	Field Data type Name	= Unsigned8
4.19	Field Name	= Device variable code 7
4.20	Field Data type Name	= Unsigned8
4.21	Field Name	= Max publish count
4.22	Field Data type Name	= Unsigned8
4.23	Field Name	= Extended command number
4.24	Field Data type Name	= Unsigned16
4.25	Field Name	= Update period
4.26	Field Data type Name	= Time
4.27	Field Name	= Maximum update period
4.28	Field Data type Name	= Time
4.29	Field Name	= Trigger mode
4.30	Field Data type Name	= Enumeration
4.31	Field Name	= Device variable classification
4.32	Field Data type Name	= Enumeration
4.33	Field Name	= Device variable unit
4.34	Field Data type Name	= Engineering unit
4.35	Field Name	= Trigger level
4.36	Field Data type Name	= Float32
5	Array size	= Variable

7.3.2.4.29 Flush delayed responses

7.3.2.4.29.1 Service overview

This command clears all pending delayed responses for the Master that issued the command.

7.3.2.4.29.2 Request parameters

CLASS: Simple Action

ATTRIBUTES:

1 Numeric Identifier = 106

7.3.2.4.30 Write publish data device variables

7.3.2.4.30.1 Service overview

This service selects the Device variables that shall be used by a publishing device in Publish data mode.

7.3.2.4.30.2 Request parameters

- 1 Numeric Identifier = 107
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.30.3 Response parameters

- 1 Numeric Identifier = 107
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.30.4 Variable model**CLASS:** Array of Structure VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 107
- 2 Variable Name = Publish data device variables
- 3 Number of Fields = 8
- 4 List of Fields
- 4.1 Field Name = Device variable code 0
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Device variable code 1
- 4.4 Field Data type Name = Unsigned8
- 4.5 Field Name = Device variable code 2
- 4.6 Field Data type Name = Unsigned8
- 4.7 Field Name = Device variable code 3
- 4.8 Field Data type Name = Unsigned8
- 4.9 Field Name = Device variable code 4
- 4.10 Field Data type Name = Unsigned8
- 4.11 Field Name = Device variable code 5
- 4.12 Field Data type Name = Unsigned8
- 4.13 Field Name = Device variable code 6
- 4.14 Field Data type Name = Unsigned8
- 4.15 Field Name = Device variable code 7
- 4.16 Field Data type Name = Unsigned8
- 5 Array size = Variable

7.3.2.4.31 Write publish data mode command number**7.3.2.4.31.1 Service overview**

This service selects the response message that the publishing device transmits in Publish data mode.

7.3.2.4.31.2 Request parameters

- 1 Numeric Identifier = 108
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.31.3 Response parameters

- 1 Numeric Identifier = 108
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.31.4 Variable model**CLASS:** Array of Simple VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 108
- 2 Variable Name = Publish data mode command number
- 3 Data type Name = Unsigned16 / Unsigned8
- 4 Array size = Variable

7.3.2.4.32 Write publish data mode control**7.3.2.4.32.1 Service overview**

This service is used to enable or disable the Publish data mode for devices.

7.3.2.4.32.2 Request parameters

- 1 Numeric Identifier = 109
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.32.3 Response parameters

- 1 Numeric Identifier = 109
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.32.4 Variable model**CLASS:** Array of Simple VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 109
- 2 Variable Name = Publish data mode control
- 3 Data type Name = Enumeration
- 4 Array size = Variable

7.3.2.4.33 Read event notification summary**7.3.2.4.33.1 Service overview**

This service reads the configuration of the Event Notification.

7.3.2.4.33.2 Request parameters

- 1 Numeric Identifier = 115
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.33.3 Response parameters

- 1 Numeric Identifier = 115
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.33.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 115
- 2 Variable Name = event notification summary
- 3 Number of Fields = 8
- 4 List of Fields
 - 4.1 Field Name = Maximum event count
 - 4.2 Field Data type Name = Unsigned8
 - 4.3 Field Name = Event status
 - 4.4 Field Data type Name = Bit Field
 - 4.5 Field Name = Event notification control
 - 4.6 Field Data type Name = Bit Field
 - 4.7 Field Name = Event occurrence time
 - 4.8 Field Data type Name = Time
 - 4.9 Field Name = Event notification retry time
 - 4.10 Field Data type Name = Time
 - 4.11 Field Name = Maximum update time
 - 4.12 Field Data type Name = Time
 - 4.13 Field Name = Event de-bounce interval
 - 4.14 Field Data type Name = Time
 - 4.15 Field Name = Event mask
 - 4.16 Field Data type Name = variable size array of Bit Field
- 5 Array size = Variable

7.3.2.4.34 Write event notification bit mask

7.3.2.4.34.1 Service overview

This service sets the Event mask that is used to trigger an Event notification.

7.3.2.4.34.2 Request parameters

- 1 Numeric Identifier = 116
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.34.3 Response parameters

- 1 Numeric Identifier = 116
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.34.4 Variable model**CLASS:** Array of Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 116
2	Variable Name	= event notification bit mask
3	Data type Name	= variable size array of Bit Field
4	Array size	= Variable

7.3.2.4.35 Write event notification timing**7.3.2.4.35.1 Service overview**

This service sets the timing parameters for event notification.

7.3.2.4.35.2 Request parameters

1	Numeric Identifier	= 117
2	Subindex	
2.1	Subindex Name	= Event number ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.35.3 Response parameters

1	Numeric Identifier	= 117
2	Subindex	
2.1	Subindex Name	= Event number ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.35.4 Variable model**CLASS:** Array of Structure VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 117
2	Variable Name	= Event notification timing
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Event notification retry time
4.2	Field Data type Name	= Time
4.3	Field Name	= Maximum update time
4.4	Field Data type Name	= Time
4.5	Field Name	= Event de-bounce interval
4.6	Field Data type Name	= Time
5	Array size	= Variable

7.3.2.4.36 Write event notification control**7.3.2.4.36.1 Service overview**

This service is used to enable or disable the Event notification.

7.3.2.4.36.2 Request parameters

1	Numeric Identifier	= 118
2	Subindex	
2.1	Subindex Name	= Event number ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.36.3 Response parameters

1	Numeric Identifier	= 118
2	Subindex	
2.1	Subindex Name	= Event number ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.36.4 Variable model

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 118
2	Variable Name	= Event notification control
3	Data type Name	= Bit Field
4	Array size	= Variable

7.3.2.4.37 Write event notification acknowledgement

7.3.2.4.37.1 Service overview

This service is used to acknowledge the Event Notification.

7.3.2.4.37.2 Request parameters

1	Numeric Identifier	= 119
2	Subindex	
2.1	Subindex Name	= Event number ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.37.3 Response parameters

1	Numeric Identifier	= 119
2	Subindex	
2.1	Subindex Name	= Event number ref
2.2	Subindex Data type	= Unsigned8

7.3.2.4.37.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 119
2	Variable Name	= Event notification acknowledgement
3	Number of Fields	= 4
4	List of Fields	
4.1	Field Name	= Event occurrence time
4.2	Field Data type Name	= Time
4.3	Field Name	= Configuration change counter
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= Application process status
4.6	Field Data type Name	= Bit Field
4.7	Field Name	= Additional device status
4.8	Field Data type Name	= See Note
5	Array size	= Variable

The “Additional device status” is same as the entire Structure variable defined in 7.3.2.4.21.2.

7.3.2.5 Layer management services

7.3.2.5.1 Write Join_key

7.3.2.5.1.1 Service overview

This command allows a local maintenance tool (for example a handheld) to modify a network device's Join_key.

7.3.2.5.1.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 768
2	Variable Name	= Join_key
3	Data type Name	= Security key

7.3.2.5.2 Read join status

7.3.2.5.2.1 Service overview

This service reads the field device status as it transitions through the joining process.

7.3.2.5.2.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 769
2	Variable Name	= Join status
3	Number of Fields	= 7
4	List of Fields	
4.1	Field Name	= Wireless mode
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Join status
4.4	Field Data type Name	= Bit Field
4.5	Field Name	= Neighbors
4.6	Field Data type Name	= Unsigned8
4.7	Field Name	= Advertising packets
4.8	Field Data type Name	= Unsigned8
4.9	Field Name	= Max_join_retries
4.10	Field Data type Name	= Unsigned8
4.11	Field Name	= Join_rsp_time
4.12	Field Data type Name	= Time
4.13	Field Name	= Search time
4.14	Field Data type Name	= Time

7.3.2.5.3 Write active_search_shed_time

7.3.2.5.3.1 Service overview

This command requests faster advertising. Generally, this is received locally by a network member from a local maintenance tool (for example a handheld) and is propagated to the network manager.

7.3.2.5.3.2 Request variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 770
2	Variable Name	= Active_search_shed_time
3	Data type Name	= Time

7.3.2.5.3.3 Response variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 770
2	Variable Name	= Shed_time
3	Number of Fields	= 3
4	List of Fields	
4.1	Variable Name	= Active_search_shed_time
4.2	Data type Name	= Time
4.3	Variable Name	= Advertise_time
4.4	Data type Name	= Time
4.5	Field Name	= Neighbors
4.6	Field Data type Name	= Unsigned8

7.3.2.5.4 Write join mode configuration

7.3.2.5.4.1 Service overview

This command allows a host system or handheld device to force a field device into active join mode.

7.3.2.5.4.2 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 771
2	Variable Name	= Join mode configuration
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Join mode
4.2	Field Data type Name	= Enumeration
4.3	Variable Name	= Active_search_shed_time
4.4	Data type Name	= Time

7.3.2.5.5 Read join mode configuration

7.3.2.5.5.1 Service overview

This command reads the Join mode configuration from a network device.

7.3.2.5.5.2 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 772
2	Variable Name	= Join mode configuration
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Join mode
4.2	Field Data type Name	= Enumeration
4.3	Variable Name	= Active_search_shed_time

4.4 Data type Name = Time

7.3.2.5.6 Write network_ID

7.3.2.5.6.1 Service overview

This command configures the device to recognize the proper network_ID. This command should only be issued by the network Manager or via the maintenance port. The device receiving this command is required to reject this command from any other source.

7.3.2.5.6.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 773
2	Variable Name	= Network_ID
3	Data type Name	= Unsigned16

7.3.2.5.7 Read network_ID

7.3.2.5.7.1 Service overview

This command reads the current setting of the network_ID in a device.

7.3.2.5.7.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 774
2	Variable Name	= Network_ID
3	Data type Name	= Unsigned16

7.3.2.5.8 Write network_tag

7.3.2.5.8.1 Service overview

This command writes the tag for the network. The Network_tag is a proxy for the network_ID, so that the network can be identified in a text form at the application level.

7.3.2.5.8.2 Variable model

CLASS: String VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 775
2	Variable Name	= Network_tag
3	Data type Name	= Latin-1
3.1	Octet Length	= 32

7.3.2.5.9 Read network_tag

7.3.2.5.9.1 Service overview

This command reads the current setting of the tag of the network.

7.3.2.5.9.2 Variable model

CLASS: String VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 776
2	Variable Name	= Network_tag
3	Data type Name	= Latin-1
3.1	Octet Length	= 32

7.3.2.5.10 Read wireless device capabilities

7.3.2.5.10.1 Service overview

This service is used to determine operational characteristics of a network device. The network manager is expected to use Peak packet rate and Recovery time to accommodate rechargeable or power scavenging devices.

7.3.2.5.10.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 777
2	Variable Name	= Wireless device capabilities
3	Number of Fields	= 8
4	List of Fields	
4.1	Field Name	= Power source
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Peak packet rate
4.4	Field Data type Name	= Float32
4.5	Field Name	= Load duration
4.6	Field Data type Name	= Time
4.7	Field Name	= Recovery time
4.8	Field Data type Name	= Time
4.9	Field Name	= Good RSL
4.10	Field Data type Name	= Integer8
4.11	Field Name	= Min_Keep_alive_time
4.12	Field Data type Name	= Time
4.13	Field Name	= Neighbor count
4.14	Field Data type Name	= Unsigned16
4.15	Field Name	= Packet buffer
4.16	Field Data type Name	= Unsigned16

7.3.2.5.11 Read battery life

7.3.2.5.11.1 Service overview

This service reads the current state of the battery in a network device.

7.3.2.5.11.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 778
2	Variable Name	= Battery life
3	Data type Name	= Unsigned16

7.3.2.5.12 Report device health

7.3.2.5.12.1 Service overview

This command response is periodically published by the field device.

7.3.2.5.12.2 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 779
2	Variable Name	= Device health
3	Number of Fields	= 7
4	List of Fields	
4.1	Field Name	= Packets generated
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Packets terminated
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= DLL MIC failures
4.6	Field Data type Name	= Unsigned8
4.7	Field Name	= NL MIC failures
4.8	Field Data type Name	= Unsigned8
4.9	Field Name	= Power status
4.10	Field Data type Name	= Enumeration
4.11	Field Name	= CRC errors
4.12	Field Data type Name	= Unsigned8
4.13	Field Name	= Packets missed
4.14	Field Data type Name	= Unsigned8

7.3.2.5.13 Read neighbor health list

7.3.2.5.13.1 Service overview

This service reads the entries in the Neighbor table from the network device.

7.3.2.5.13.2 Request parameters

1	Numeric Identifier	= 780
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8
3	Element count	
3.1	Element count Name	= Entry count
3.2	Element count Data type	= Unsigned8

7.3.2.5.13.3 Response parameters

1	Numeric Identifier	= 780
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8
3	Element count	
3.1	Element count Name	= Entry count
3.2	Element count Data type	= Unsigned8
4	Array size	
4.1	Array size Name	= Neighbor count

4.2 Array size Data type = Unsigned8

7.3.2.5.13.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 780
2	Variable Name	= Neighbor health
3	Number of Fields	= 6
4	List of Fields	
4.1	Field Name	= Neighbor_nickname
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Neighbor flags
4.4	Field Data type Name	= Bit Field
4.5	Field Name	= Average_RSL
4.6	Field Data type Name	= Integer8
4.7	Field Name	= Packets_transmitted
4.8	Field Data type Name	= Unsigned16
4.9	Field Name	= Missed_ACK_packets
4.10	Field Data type Name	= Unsigned16
4.11	Field Name	= Packets_received
4.12	Field Data type Name	= Unsigned16
5	Array size	= Variable

7.3.2.5.14 Read device nickname

7.3.2.5.14.1 Service overview

This service reads a network device's nickname (short address).

7.3.2.5.14.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 781
2	Variable Name	= Device nickname
3	Data type Name	= Unsigned16

7.3.2.5.15 Read session list

7.3.2.5.15.1 Service overview

This service reads the entries in the Session table from the network device.

7.3.2.5.15.2 Request parameters

1	Numeric Identifier	= 782
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8
3	Element count	
3.1	Element count Name	= Entry count
3.2	Element count Data type	= Unsigned8

7.3.2.5.15.3 Response parameters

- 1 Numeric Identifier = 782
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Session count
 - 4.2 Array size Data type = Unsigned8

7.3.2.5.15.4 Variable model**CLASS:** Array of Structure VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 782
- 2 Variable Name = Session
- 3 Number of Fields = 5
- 4 List of Fields
 - 4.1 Field Name = Session_type
 - 4.2 Field Data type Name = Enumeration
 - 4.3 Field Name = Peer_nickname
 - 4.4 Field Data type Name = Unsigned16
 - 4.5 Field Name = Peer_unique_ID
 - 4.6 Field Data type Name = Unique ID
 - 4.7 Field Name = Peer_nonce_counter
 - 4.8 Field Data type Name = Unsigned32
 - 4.9 Field Name = My_nonce_counter
 - 4.10 Field Data type Name = Unsigned32
- 5 Array size = Variable

7.3.2.5.16 Read superframe list**7.3.2.5.16.1 Service overview**

This service reads the entries in the superframe table from the network device.

7.3.2.5.16.2 Request parameters

- 1 Numeric Identifier = 783
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8

7.3.2.5.16.3 Response parameters

- 1 Numeric Identifier = 783
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count

- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8
- 4 Array size
- 4.1 Array size Name = Superframe count
- 4.2 Array size Data type = Unsigned8

7.3.2.5.16.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 783
- 2 Variable Name = Superframe
- 3 Number of Fields = 3
- 4 List of Fields
- 4.1 Field Name = Superframe_ID
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Num_Slots
- 4.4 Field Data type Name = Unsigned16
- 4.5 Field Name = Superframe mode
- 4.6 Field Data type Name = Bit Field
- 5 Array size = Variable

7.3.2.5.17 Read link list

7.3.2.5.17.1 Service overview

This service reads the entries in the Link table from the network device.

7.3.2.5.17.2 Request parameters

- 1 Numeric Identifier = 784
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned16
- 3 Element count
- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8

7.3.2.5.17.3 Response parameters

- 1 Numeric Identifier = 784
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned16
- 3 Element count
- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8
- 4 Array size
- 4.1 Array size Name = Link count
- 4.2 Array size Data type = Unsigned16

7.3.2.5.17.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 784

2	Variable Name	= Link
3	Number of Fields	= 6
4	List of Fields	
4.1	Field Name	= Link_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Slot
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= Channel_Offset
4.6	Field Data type Name	= Unsigned8
4.7	Field Name	= Neighbor_nickname
4.8	Field Data type Name	= Unsigned16
4.9	Field Name	= Link options
4.10	Field Data type Name	= Bit Field
4.11	Field Name	= Link_type
4.12	Field Data type Name	= Enumeration
5	Array size	= Variable

7.3.2.5.18 Read graph list

7.3.2.5.18.1 Service overview

This service reads one item in the Graph table from the network device.

7.3.2.5.18.2 Request parameters

1	Numeric Identifier	= 785
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8

7.3.2.5.18.3 Response parameters

1	Numeric Identifier	= 785
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8
4	Array size	
4.1	Array size Name	= Graph count
4.2	Array size Data type	= Unsigned8

7.3.2.5.18.4 Response variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 785
2	Variable Name	= Graph
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Graph_ID
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Neighbor count
4.4	Field Data type Name	= Unsigned8
4.5	Field Name	= Neighbor list
4.6	Field Data type Name	= Array of Neighbor_nickname
5	Array size	= Variable

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Neighbor_nickname
- 2 Format = ARRAY
- 2.1 Number of Array Elements = value of Neighbor count
- 2.2 Array Element Data type = Unsigned16

7.3.2.5.19 Read neighbor attribute flag

7.3.2.5.19.1 Service overview

This command reads the attributes of a neighbor from the network device.

7.3.2.5.19.2 Request parameters

- 1 Numeric Identifier = 786
- 2 Additional Identifier
- 2.1 Name = Neighbor_nickname
- 2.2 Data type = Unsigned8

7.3.2.5.19.3 Response parameters

- 1 Numeric Identifier = 786
- 2 Additional Identifier
- 2.1 Name = Neighbor_nickname
- 2.2 Data type = Unsigned8

7.3.2.5.19.4 Response variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 786
- 2 Variable Name = Neighbor flags
- 3 Data type Name = Bit Field

7.3.2.5.20 Read neighbor signal level

7.3.2.5.20.1 Service overview

This service reads the signal level in the Neighbor table from the network device.

7.3.2.5.20.2 Request parameters

- 1 Numeric Identifier = 787
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned16
- 3 Element count
- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8

7.3.2.5.20.3 Response parameters

- 1 Numeric Identifier = 787
- 2 Subindex
- 2.1 Subindex Name = Table index

- 2.2 Subindex Data type = Unsigned16
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Neighbor count
 - 4.2 Array size Data type = Unsigned16

7.3.2.5.20.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 787
- 2 Variable Name = Neighbor signal level
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Neighbor_nickname
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = Average_RSL
 - 4.4 Field Data type Name = Integer8
- 5 Array size = Variable

7.3.2.5.21 Report path down alarm

7.3.2.5.21.1 Service overview

The device publishes the path down alarm response using this service.

7.3.2.5.21.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 788
- 2 Variable Name = Neighbor_nickname
- 3 Data type Name = Unsigned16

7.3.2.5.22 Report source route failed alarm

7.3.2.5.22.1 Service overview

The device publishes the source route failed alarm response using this service.

7.3.2.5.22.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 789
- 2 Variable Name = Source route failure
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Unreachable neighbor
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = NL MIC
 - 4.4 Field Data type Name = Unsigned32

7.3.2.5.23 Report graph route failed alarm

7.3.2.5.23.1 Service overview

The device publishes the graph route failed alarm response using this service.

7.3.2.5.23.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 790
2	Variable Name	= Graph_ID
3	Data type Name	= Unsigned16

7.3.2.5.24 Report transport layer failed alarm

7.3.2.5.24.1 Service overview

The device publishes the transport layer failed alarm response using this service.

7.3.2.5.24.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 791
2	Variable Name	= Unreachable peer
3	Data type Name	= Unsigned16

7.3.2.5.25 Write UTC time mapping

7.3.2.5.25.1 Service overview

This service sets the start of ASN zero time to the time in this request, in a network device.

7.3.2.5.25.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 793
2	Variable Name	= UTC time mapping
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Date
4.2	Field Data type Name	= Date
4.3	Variable Name	= Time of day
4.4	Data type Name	= Time

7.3.2.5.26 Read UTC time mapping

7.3.2.5.26.1 Service overview

This service reads the UTC time when the ASN zero started, as set in the network device.

7.3.2.5.26.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 794

- 2 Variable Name = UTC time mapping
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Date
 - 4.2 Field Data type Name = Date
 - 4.3 Variable Name = Time of day
 - 4.4 Data type Name = Time

7.3.2.5.27 Write timer interval

7.3.2.5.27.1 Service overview

This service sets the time interval for the timer specified by Timer_type in a network device.

7.3.2.5.27.2 Request parameters

- 1 Numeric Identifier = 795
- 2 Subindex
 - 2.1 Subindex Name = Timer_type
 - 2.2 Subindex Data type = Enumeration

7.3.2.5.27.3 Response parameters

- 1 Numeric Identifier = 795
- 2 Subindex
 - 2.1 Subindex Name = Timer_type
 - 2.2 Subindex Data type = Enumeration

7.3.2.5.27.4 Variable model

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 795
- 2 Variable Name = Interval
- 3 Data type Name = Unsigned32
- 4 Array size = 7

7.3.2.5.28 Read timer interval

7.3.2.5.28.1 Service overview

This service reads the time interval for the timer specified by Timer_type from a network device.

7.3.2.5.28.2 Request parameters

- 1 Numeric Identifier = 796
- 2 Subindex
 - 2.1 Subindex Name = Timer_type
 - 2.2 Subindex Data type = Enumeration

7.3.2.5.28.3 Response parameters

- 1 Numeric Identifier = 796
- 2 Subindex
 - 2.1 Subindex Name = Timer_type
 - 2.2 Subindex Data type = Enumeration

7.3.2.5.28.4 Variable model

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

5	Numeric Identifier	= 796
6	Variable Name	= Interval
7	Data type Name	= Unsigned32
8	Array size	= 7

7.3.2.5.29 Write radio power output

7.3.2.5.29.1 Service overview

This service sets the radio power output for a device or gateway.

7.3.2.5.29.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 797
2	Variable Name	= Output power
3	Data type Name	= Integer8

7.3.2.5.30 Read radio power output

7.3.2.5.30.1 Service overview

This service reads the radio power output from a device or gateway.

7.3.2.5.30.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 798
2	Variable Name	= Output power
3	Data type Name	= Integer8

7.3.2.5.31 Request timetable

7.3.2.5.31.1 Service overview

This service is used by a wireless device to add a service connection to another wireless device with specified bandwidth and latency characteristics.

7.3.2.5.31.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 799
2	Variable Name	= Timetable
3	Number of Fields	= 6
4	List of Fields	
4.1	Field Name	= Timetable_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Timetable_type
4.4	Field Data type Name	= Bit Field

- 4.5 Field Name = Application domain
- 4.6 Field Data type Name = Enumeration
- 4.7 Field Name = Peer_nickname
- 4.8 Field Data type Name = Unsigned16
- 4.9 Variable Name = Duration
- 4.10 Data type Name = Time
- 4.11 Variable Name = Route_ID
- 4.12 Data type Name = Unsigned8

7.3.2.5.32 Read timetable list

7.3.2.5.32.1 Service overview

This service reads the entries in the Timetable table from the network device.

7.3.2.5.32.2 Request parameters

- 1 Numeric Identifier = 800
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8

7.3.2.5.32.3 Response parameters

- 1 Numeric Identifier = 800
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Timetable count
 - 4.2 Array size Data type = Unsigned8

7.3.2.5.32.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 800
- 2 Variable Name = Timetable
- 3 Number of Fields = 6
- 4 List of Fields
 - 4.1 Field Name = Timetable_ID
 - 4.2 Field Data type Name = Unsigned8
 - 4.3 Field Name = Timetable_type
 - 4.4 Field Data type Name = Bit Field
 - 4.5 Field Name = Application domain
 - 4.6 Field Data type Name = Enumeration
 - 4.7 Field Name = Peer_nickname
 - 4.8 Field Data type Name = Unsigned16
 - 4.9 Field Name = Service_duration

- 4.10 Field Data type Name = Time
- 4.11 Field Name = Route_ID
- 4.12 Field Data type Name = Unsigned8
- 5 Array size = Variable

7.3.2.5.33 Delete timetable connection

7.3.2.5.33.1 Service overview

This service notifies a device of Timetable deletion.

7.3.2.5.33.2 Request parameters

CLASS:	Qualified Action
ATTRIBUTES:	
1 Numeric Identifier	= 801
2 Variable name	= Timetable_ID
3 Data type name	= Unsigned8
4 Qualifier Name	= Reason
5 Data type Name	= Enumeration

7.3.2.5.33.3 Response parameters

CLASS:	Qualified Action
ATTRIBUTES:	
1 Numeric Identifier	= 801
2 Variable name	= Timetable_ID
3 Data type name	= Unsigned8
4 Qualifier Name	= Reason
5 Data type Name	= Enumeration

7.3.2.5.33.4 Response variable model

CLASS:	Simple VARIABLE
ATTRIBUTES:	
1 Numeric Identifier	= 801
2 Variable Name	= Timetable count
3 Data type Name	= Unsigned8

7.3.2.5.34 Read route list

7.3.2.5.34.1 Service overview

This service reads the entries in the Route table from the network device.

7.3.2.5.34.2 Request parameters

- 1 Numeric Identifier = 802
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8

7.3.2.5.34.3 Response parameters

- 1 Numeric Identifier = 802
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Route count
 - 4.2 Array size Data type = Unsigned8

7.3.2.5.34.4 Variable model**CLASS:** Array of Structure VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 802
- 2 Variable Name = Route
- 3 Number of Fields = 5
- 4 List of Fields
 - 4.1 Field Name = Routes remaining
 - 4.2 Field Data type Name = Unsigned8
 - 4.3 Field Name = Route_ID
 - 4.4 Field Data type Name = Unsigned8
 - 4.5 Field Name = Destination nickname
 - 4.6 Field Data type Name = Unsigned16
 - 4.7 Field Name = Graph_ID
 - 4.8 Field Data type Name = Unsigned16
 - 4.9 Field Name = Source route flag
 - 4.10 Field Data type Name = Unsigned8
- 5 Array size = Variable

7.3.2.5.35 Read source route**7.3.2.5.35.1 Service overview**

This service reads the contents of a particular Source route.

7.3.2.5.35.2 Request parameters

- 1 Numeric Identifier = 803
- 2 Additional Identifier
 - 2.1 Name = Route_ID
 - 2.2 Data type = Unsigned8

7.3.2.5.35.3 Response parameters

- 3 Numeric Identifier = 803
- 4 Additional Identifier
 - 4.1 Name = Route_ID
 - 4.2 Data type = Unsigned8

7.3.2.5.35.4 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 803
2	Variable Name	= Source route
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Hop count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Hop list
4.4	Field Data type name	= Array of Hop

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Array of Hop
2	Format	= ARRAY
2.1	Number of Array Elements	= value of Hop count
2.2	Array Element Data type	= Unsigned16

7.3.2.5.36 Read CCA mode

7.3.2.5.36.1 Service overview

This service reads the Clear Channel Assessment (CCA) mode from a network device.

7.3.2.5.36.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 804
2	Variable Name	= CCA mode
3	Data type Name	= Enumeration

7.3.2.5.37 Write CCA mode

7.3.2.5.37.1 Service overview

This service writes the Clear Channel Assessment (CCA) mode in a network device.

7.3.2.5.37.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 805
2	Variable Name	= CCA mode
3	Data type Name	= Enumeration

7.3.2.5.38 Read handheld superframe

7.3.2.5.38.1 Service overview

This service is used to determine if the handheld superframe in a network device is enabled.

7.3.2.5.38.2 Variable model**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 806
2	Variable Name	= Superframe
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Superframe_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Num_Slots
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= Superframe mode
4.6	Field Data type Name	= Bit Field

7.3.2.5.39 Write handheld superframe**7.3.2.5.39.1 Service overview**

This service allows a handheld device to request that a network device enables the handheld superframe, so that maintenance can be done.

7.3.2.5.39.2 Variable model**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 807
2	Variable Name	= Superframe
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Superframe_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Superframe mode
4.4	Field Data type Name	= Bit Field

7.3.2.5.40 Read packet hop-to-live**7.3.2.5.40.1 Service overview**

This service reads the current value of Hop-to-live configured for a network device.

7.3.2.5.40.2 Variable model**CLASS: Simple VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 808
2	Variable Name	= Hop-to-live
3	Data type Name	= Unsigned8

7.3.2.5.41 Write packet hop-to-live**7.3.2.5.41.1 Service overview**

This service writes the Hop-to-live value in a network device.

7.3.2.5.41.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 809
2	Variable Name	= Hop-to-live
3	Data type Name	= Unsigned8

7.3.2.5.42 Read join_priority

7.3.2.5.42.1 Service overview

This service reads the current value of Join_priority from a network device.

7.3.2.5.42.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 810
2	Variable Name	= Join_priority
3	Data type Name	= Unsigned8

7.3.2.5.43 Write join_priority

7.3.2.5.43.1 Service overview

This service writes the Join_priority value in a network device.

7.3.2.5.43.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 811
2	Variable Name	= Join_priority
3	Data type Name	= Unsigned8

7.3.2.5.44 Read receive Priority_threshold

7.3.2.5.44.1 Service overview

This service reads the current value of Receive Priority_threshold from a network device.

7.3.2.5.44.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 812
2	Variable Name	= Priority_threshold
3	Data type Name	= Unsigned8

7.3.2.5.45 Write receive Priority_threshold

7.3.2.5.45.1 Service overview

This service writes the Receive Priority_threshold value in a network device.

7.3.2.5.45.2 Variable model**CLASS:** Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 813
2	Variable Name	= Priority_threshold
3	Data type Name	= Unsigned8

7.3.2.5.46 Read device list**7.3.2.5.46.1 Service overview**

This service reads the entries in the Device table for the type of list specified by Device code in the request.

7.3.2.5.46.2 Request parameters

1	Numeric Identifier	= 814
2	Additional Identifier	
2.1	Name	= Device code
2.2	Data type	= Enumeration
3	Subindex	
3.1	Subindex Name	= Table index
3.2	Subindex Data type	= Unsigned16
4	Element count	
4.1	Element count Name	= Entry count
4.2	Element count Data type	= Unsigned8

7.3.2.5.46.3 Response parameters

1	Numeric Identifier	= 802
2	Additional Identifier	
2.1	Name	= Device code
2.2	Data type	= Enumeration
3	Subindex	
3.1	Subindex Name	= Table index
3.2	Subindex Data type	= Unsigned16
4	Element count	
4.1	Element count Name	= Entry count
4.2	Element count Data type	= Unsigned8
5	Array size	
5.1	Array size Name	= Device count
5.2	Array size Data type	= Unsigned16

7.3.2.5.46.4 Variable model**CLASS:** Array of Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 814
2	Variable Name	= Device list
3	Data type Name	= Unique ID
4	Array size	= Variable

7.3.2.5.47 Add device list entry

7.3.2.5.47.1 Service overview

This service adds a device in the Device table for the type of list specified by Device code.

7.3.2.5.47.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 815
2	Variable name	= Device code
3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.47.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 815
2	Variable name	= Device code
3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.47.4 Response variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 815
2	Variable Name	= Device count
3	Data type Name	= Unsigned16

7.3.2.5.48 Delete device list entry

7.3.2.5.48.1 Service overview

This service deletes a device in the Device table for the type of list specified by Device code in a gateway or network manager.

7.3.2.5.48.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 816
2	Variable name	= Device code
3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.48.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 816
2	Variable name	= Device code

3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.48.4 Response variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 816
2	Variable Name	= Device count
3	Data type Name	= Unsigned16

7.3.2.5.49 Read channel blacklist

7.3.2.5.49.1 Service overview

This service reads the current channel blacklist and the blacklist that shall be used after the next restart.

7.3.2.5.49.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 817
2	Variable Name	= Channel blacklist
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Channel count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Current channel map
4.4	Field Data type Name	= Bit Field
4.5	Field Name	= Pending channel map
4.6	Field Data type Name	= Bit Field

7.3.2.5.50 Write channel blacklist

7.3.2.5.50.1 Service overview

This service writes the pending channel map, which is the blacklist to the gateway. It is used after the next restart.

7.3.2.5.50.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 818
2	Variable Name	= Channel blacklist
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Channel count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Pending channel map
4.4	Field Data type Name	= Bit Field

7.3.2.5.51 Read back-off exponent

7.3.2.5.51.1 Service overview

This service reads the current value of Max_backoff_exp from a network device.

7.3.2.5.51.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 819
2	Variable Name	= Max_backoff_exp
3	Data type Name	= Unsigned8

7.3.2.5.52 Write back-off exponent

7.3.2.5.52.1 Service overview

This service writes the Max_backoff_exp value in a network device.

7.3.2.5.52.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 820
2	Variable Name	= Max_backoff_exp
3	Data type Name	= Unsigned8

7.3.2.5.53 Write network access mode

7.3.2.5.53.1 Service overview

This service writes the network access mode value in a network device.

7.3.2.5.53.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 821
2	Variable Name	= Network access mode
3	Data type Name	= Enumeration

7.3.2.5.54 Read network access mode

7.3.2.5.54.1 Service overview

This service reads the current value of network access mode from a network device.

7.3.2.5.54.2 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 822
2	Variable Name	= Network access mode
3	Data type Name	= Enumeration

7.3.2.5.55 Request session**7.3.2.5.55.1 Service overview**

This service requests a unicast session from a handheld to a device.

7.3.2.5.55.2 Request parameters

1	Numeric Identifier	= 823
2	Additional Identifier	
2.1	Name	= Peer_nickname
2.2	Data type	= Unsigned16

7.3.2.5.55.3 Response parameters

1	Numeric Identifier	= 823
2	Additional Identifier	
2.1	Name	= Peer_nickname
2.2	Data type	= Unsigned16

7.3.2.5.55.4 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 823
2	Variable Name	= Session
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Peer_nonce_counter
4.2	Field Data type Name	= Unsigned32
4.3	Field Name	= Join_key
4.4	Field Data type Name	= Security key

7.3.2.6 Gateway and network manager services**7.3.2.6.1 Read network device identity using Unique ID****7.3.2.6.1.1 Service overview**

This service reads the identity information of the device with matching Device ID from the gateway.

7.3.2.6.1.2 Request parameters

1	Numeric Identifier	= 832
2	Additional Identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.1.3 Response parameters

1	Numeric Identifier	= 832
2	Additional Identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.1.4 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 832
- 2 Variable Name = Device identity
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Device nickname
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = Long_tag
 - 4.4 Field Data type Name = Latin-1
 - 4.4.1 Octet Length = 32

7.3.2.6.2 Read device's neighbor health

7.3.2.6.2.1 Service overview

This service reads from the gateway, information regarding the quality of the connection to each active neighbor or potential neighbor for the device specified in the request.

7.3.2.6.2.2 Request parameters

- 1 Numeric Identifier = 833
- 2 Additional Identifier
 - 2.1 Name = Unique ID
 - 2.2 Data type = Unique ID
- 3 Subindex
 - 3.1 Subindex Name = Table index
 - 3.2 Subindex Data type = Unsigned8
- 4 Element count
 - 4.1 Element count Name = Entry count
 - 4.2 Element count Data type = Unsigned8

7.3.2.6.2.3 Response parameters

- 1 Numeric Identifier = 833
- 2 Additional Identifier
 - 2.1 Name = Unique ID
 - 2.2 Data type = Unique ID
- 3 Subindex
 - 3.1 Subindex Name = Table index
 - 3.2 Subindex Data type = Unsigned8
- 4 Element count
 - 4.1 Element count Name = Entry count
 - 4.2 Element count Data type = Unsigned8

7.3.2.6.2.4 Variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 833
- 2 Variable Name = Neighbor health
- 3 Number of Fields = 5
- 4 List of Fields
 - 4.1 Field Name = Neighbor_nickname

- 4.2 Field Data type Name = Unsigned16
- 4.3 Field Name = Average_RSL
- 4.4 Field Data type Name = Integer8
- 4.5 Field Name = Packets_transmitted
- 4.6 Field Data type Name = Unsigned32
- 4.7 Field Name = Missed_ACK_packets
- 4.8 Field Data type Name = Unsigned32
- 4.9 Field Name = Packets_received
- 4.10 Field Data type Name = Unsigned32
- 5 Array size = Variable

7.3.2.6.3 Read network topology information

7.3.2.6.3.1 Service overview

This service reads one item in the Graph table for the specified network device from the gateway. By reading Command 814 the application layer can determine how many graphs a particular device is participating in by using “Read graph list” (see 7.3.2.5.18). This service can then be issued as many times as needed to determine the specific Graph_IDs that the device is participating in.

7.3.2.6.3.2 Request parameters

- 1 Numeric Identifier = 834
- 2 Additional Identifier
 - 2.1 Name = Unique ID
 - 2.2 Data type = Unique ID
- 3 Subindex
 - 3.1 Subindex Name = Table index
 - 3.2 Subindex Data type = Unsigned16

7.3.2.6.3.3 Response parameters

- 1 Numeric Identifier = 834
- 2 Additional Identifier
 - 2.1 Name = Unique ID
 - 2.2 Data type = Unique ID
- 3 Subindex
 - 3.1 Subindex Name = Table index
 - 3.2 Subindex Data type = Unsigned16
- 5 Array size
 - 5.1 Array size Name = Graph count
 - 5.2 Array size Data type = Unsigned16

7.3.2.6.3.4 Response variable model

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 834
- 2 Variable Name = Graph
- 3 Number of Fields = 3
- 4 List of Fields
 - 4.1 Field Name = Graph_ID
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = Neighbor count
 - 4.4 Field Data type Name = Unsigned16

- 4.5 Field Name = Neighbor list
- 4.6 Field Data type Name = Array of Neighbor
- 5 Array size = Variable

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Neighbor
- 2 Format = ARRAY
- 2.1 Number of Array Elements = value of Neighbor count
- 2.2 Array Element Data type = Unsigned16

7.3.2.6.4 Read publish data message list

7.3.2.6.4.1 Service overview

This service reads from the gateway, the publish data message list that the specified device participates in, including number of publish data mode commands that the specified device participates in. The requesting application can use this service to build the list of all publish data mode traffic in the system.

7.3.2.6.4.2 Request parameters

- 1 Numeric Identifier = 835
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.4.3 Response parameters

- 1 Numeric Identifier = 835
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.4.4 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 835
- 2 Variable Name = Publish data message list
- 3 Number of Fields = 2
- 4 List of Fields
- 4.1 Field Name = Publish data count
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Publish data
- 4.4 Field Data type Name = Array of Publish data

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Publish data
- 2 Format = ARRAY
- 2.1 Number of Array Elements = value of Publish data count
- 2.2 Array Element Data type = Publish data

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Publish data
- 2 Format = STRUCTURE
- 2.1 Number of Fields = 2
- 2.2 List of Fields
- 2.2.1 Field Name = Publish command
- 2.2.2 Field Data type Name = Unsigned16
- 2.2.3 Field Name = Packet count
- 2.2.4 Field Data type Name = Unsigned32

7.3.2.6.5 Flush cached responses for a device

7.3.2.6.5.1 Service overview

This service flushes the cached responses in the gateway, for the specified device.

7.3.2.6.5.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

- 1 Numeric Identifier = 836
- 2 Variable name = Unique ID
- 3 Data type name = Unique ID

7.3.2.6.6 Write update notification bit mask for a device

7.3.2.6.6.1 Service overview

This service registers a gateway client for notification updates.

7.3.2.6.6.2 Request parameters

- 1 Numeric Identifier = 837
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.6.3 Response parameters

- 1 Numeric Identifier = 837
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.6.4 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 837
- 2 Field Name = Notification mask
- 3 Field Data type Name = Bit Field

7.3.2.6.7 Read update notification bit mask for a device

7.3.2.6.7.1 Service overview

This service reads the list of update notification mask for the specified device from the gateway.

7.3.2.6.7.2 Request parameters

1	Numeric Identifier	= 838
2	Additional Identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.7.3 Response parameters

1	Numeric Identifier	= 838
2	Additional Identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.7.4 Variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

4	Numeric Identifier	= 838
5	Field Name	= Notification mask
6	Field Data type Name	= Bit Field

7.3.2.6.8 Read change notification

7.3.2.6.8.1 Service overview

This request is sent by the client application to the gateway. The response lists the changes for the device specified in the request.

7.3.2.6.8.2 Request parameters

1	Numeric Identifier	= 839
2	Additional Identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.8.3 Response parameters

1	Numeric Identifier	= 839
2	Additional Identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.8.4 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 839
2	Variable Name	= change notification list
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Change count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Change notification
4.4	Field Data type Name	= Array of Change notification

CLASS: **Data type**

ATTRIBUTES:

- 1 Data type Name = Array of Change notification
- 2 Format = ARRAY
- 2.1 Number of Array Elements = value of Change count
- 2.2 Array Element Data type = Unsigned16

7.3.2.6.9 Read network device statistics

7.3.2.6.9.1 Service overview

This service reads the statistics for the specified device from the gateway.

7.3.2.6.9.2 Request parameters

- 1 Numeric Identifier = 840
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.9.3 Response parameters

- 1 Numeric Identifier = 840
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.9.4 Response variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

- 1 Numeric Identifier = 840
- 2 Variable Name = Device statistics
- 3 Number of Fields = 13
- 4 List of Fields
 - 4.1 Field Name = Graph count
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = Frame count
 - 4.4 Field Data type Name = Unsigned16
 - 4.5 Field Name = Link count
 - 4.6 Field Data type Name = Unsigned16
 - 4.7 Field Name = Neighbor count
 - 4.8 Field Data type Name = Unsigned8
 - 4.9 Field Name = Latency
 - 4.10 Field Data type Name = Time
 - 4.11 Field Name = Join count
 - 4.12 Field Data type Name = Unsigned16
 - 4.13 Field Name = Join date
 - 4.14 Field Data type Name = Date
 - 4.15 Field Name = Join time
 - 4.16 Field Data type Name = Time
 - 4.17 Field Name = Packets generated
 - 4.18 Field Data type Name = Unsigned32
 - 4.19 Field Name = Packets terminated
 - 4.20 Field Data type Name = Unsigned32

- 4.21 Field Name = DLL MIC failures
- 4.22 Field Data type Name = Unsigned32
- 4.23 Field Name = NL MIC failures
- 4.24 Field Data type Name = Unsigned32
- 4.25 Field Name = CRC errors
- 4.26 Field Data type Name = Unsigned32
- 4.27 Field Name = Packets missed
- 4.28 Field Data type Name = Unsigned32
- 4.29 Field Name = Nonce counter missed
- 4.30 Field Data type Name = Unsigned32
- 4.31 Field Name = Latency
- 4.32 Field Data type Name = Time

7.3.2.6.10 Read network device identity using nickname

7.3.2.6.10.1 Service overview

This service reads the identity information of the device with matching Device nickname from the gateway.

7.3.2.6.10.2 Request parameters

- 1 Numeric Identifier = 841
- 2 Additional Identifier
 - 2.1 Name = Device_nickname
 - 2.2 Data type = Unsigned16

7.3.2.6.10.3 Response parameters

- 1 Numeric Identifier = 841
- 2 Additional Identifier
 - 2.1 Name = Device_nickname
 - 2.2 Data type = Unsigned16

7.3.2.6.10.4 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 841
- 2 Variable Name = Device identity
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Unique ID
 - 4.2 Field Data type Name = Unique ID
 - 4.3 Field Name = Long_tag
 - 4.4 Field Data type Name = Latin-1
 - 4.4.1 Octet Length = 32

7.3.2.6.11 Write network device's scheduling flags

7.3.2.6.11.1 Service overview

This service allows user to request special consideration for a device when the network manager is creating schedules.

7.3.2.6.11.2 Request parameters

- 1 Numeric Identifier = 842
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.11.3 Response parameters

- 1 Numeric Identifier = 842
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.11.4 Variable model**CLASS:** Simple VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 842
- 2 Variable Name = Scheduling flags
- 3 Data type Name = Bit Field

7.3.2.6.12 Read network device's scheduling flags**7.3.2.6.12.1 Service overview**

This service reads the network device attributes that a network manager may consider when creating schedules.

7.3.2.6.12.2 Request parameters

- 1 Numeric Identifier = 843
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.12.3 Response parameters

- 1 Numeric Identifier = 843
- 2 Additional Identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.12.4 Variable model**CLASS:** Simple VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 843
- 2 Variable Name = Scheduling flags
- 3 Data type Name = Bit Field

7.3.2.6.13 Read network constraints**7.3.2.6.13.1 Service overview**

This service reads the current setting of the network management strategy and the number of request / response messages per 10 s.

7.3.2.6.13.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 844
2	Variable Name	= Network constraints
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Optimization flags
4.2	Field Data type Name	= Bit Field
4.3	Field Name	= Message pair count
4.4	Field Data type Name	= Unsigned8

7.3.2.6.14 Write network constraints

7.3.2.6.14.1 Service overview

This service writes the current setting of the network management strategy and the number of request / response messages per 10 s.

7.3.2.6.14.2 Variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 845
2	Variable Name	= Network constraints
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Optimization flags
4.2	Field Data type Name	= Bit Field
4.3	Field Name	= Message pair count
4.4	Field Data type Name	= Unsigned8

7.3.2.7 Network management configuration services

7.3.2.7.1 Disconnect device

7.3.2.7.1.1 Service overview

This service allows the network manager to force a device off the network.

7.3.2.7.1.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 960
2	Variable name	= Disconnect cause
3	Data type name	= Enumeration

7.3.2.7.1.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 960
2	Variable name	= Disconnect cause
3	Data type name	= Enumeration

7.3.2.7.2 Write network key**7.3.2.7.2.1 Service overview**

This service writes the network key in a network device.

7.3.2.7.2.2 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 961
2	Variable Name	= Network key
3	Number of Fields	= 2
4	List of Fields	
4.1	Variable Name	= Join_key
4.2	Data type Name	= Security key
4.3	Field Name	= Key change time
4.4	Field Data type Name	= Unsigned40

7.3.2.7.3 Write device nickname**7.3.2.7.3.1 Service overview**

This service writes a network device's nickname.

7.3.2.7.3.2 Variable model

CLASS: **Simple VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 962
2	Variable Name	= Device nickname
3	Data type Name	= Unsigned16

7.3.2.7.4 Write session**7.3.2.7.4.1 Service overview**

This service writes the session parameters required to establish a session between the destination (of this request) device and the peer device specified in the request.

7.3.2.7.4.2 Request variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 963
2	Variable Name	= Session entry
3	Number of Fields	= 7
3.1	Field Name	= Session_type
3.2	Field Data type Name	= Enumeration
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Peer_unique_ID
3.6	Field Data type Name	= Unique ID
3.7	Field Name	= Peer_nonce_counter
3.8	Field Data type Name	= Unsigned32
3.9	Field Name	= Session_key
3.10	Field Data type Name	= Security key

- 3.11 Field Name = Reserved
- 3.12 Field Data type Name = Unsigned8
- 3.13 Field Name = Session change time
- 3.14 Field Data type Name = Unsigned40

7.3.2.7.4.3 Response variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 963
- 2 Variable Name = Session entry
- 3 Number of Fields = 7
- 3.1 Field Name = Session_type
- 3.2 Field Data type Name = Enumeration
- 3.3 Field Name = Peer_nickname
- 3.4 Field Data type Name = Unsigned16
- 3.5 Field Name = Peer_unique_ID
- 3.6 Field Data type Name = Unique ID
- 3.7 Field Name = Peer_nonce_counter
- 3.8 Field Data type Name = Unsigned32
- 3.9 Field Name = Session_key
- 3.10 Field Data type Name = Security key
- 3.11 Field Name = Session count
- 3.12 Field Data type Name = Unsigned8
- 3.13 Field Name = Session change time
- 3.14 Field Data type Name = Unsigned40

7.3.2.7.5 Delete session

7.3.2.7.5.1 Service overview

This service deletes an end-to-end session in a network device.

7.3.2.7.5.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

- 1 Numeric Identifier = 964
- 2 Variable name = Session_type
- 3 Data type name = Enumeration
- 4 Qualifier Name = Peer_nickname
- 5 Data type Name = Unsigned16

7.3.2.7.5.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

- 1 Numeric Identifier = 964
- 2 Variable name = Session_type
- 3 Data type name = Enumeration
- 4 Qualifier Name = Peer_nickname
- 5 Data type Name = Unsigned16

7.3.2.7.5.4 Response variable model**CLASS:** Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 964
2	Variable Name	= Session count
3	Data type Name	= Unsigned8

7.3.2.7.6 Write superframe**7.3.2.7.6.1 Service overview**

This service writes or modifies a Superframe in a network device.

7.3.2.7.6.2 Request variable model**CLASS:** STRUCTURE VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 965
2	Variable Name	= Superframe entry
3	Number of Fields	= 5
3.1	Field Name	= Superframe_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Num_Slots
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Superframe mode
3.6	Field Data type Name	= Bit Field
3.7	Field Name	= Reserved
3.8	Field Data type Name	= Unsigned8
3.9	Field Name	= Superframe change time
3.10	Field Data type Name	= Unsigned40

7.3.2.7.6.3 Response variable model**CLASS:** STRUCTURE VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 965
2	Variable Name	= Superframe entry
3	Number of Fields	= 5
3.1	Field Name	= Superframe_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Num_Slots
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Superframe mode
3.6	Field Data type Name	= Bit Field
3.7	Field Name	= Superframe count
3.8	Field Data type Name	= Unsigned8
3.9	Field Name	= Superframe change time
3.10	Field Data type Name	= Unsigned40

7.3.2.7.7 Delete superframe**7.3.2.7.7.1 Service overview**

This service deletes a superframe in a network device.

7.3.2.7.7.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 966
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration

7.3.2.7.7.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 966
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration

7.3.2.7.7.4 Response variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 966
2	Field Name	= Superframe count
3	Field Data type Name	= Unsigned8

7.3.2.7.8 Write link

7.3.2.7.8.1 Service overview

This service adds a link assignment to a network device.

7.3.2.7.8.2 Request variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 967
2	Variable Name	= Link entry
3	Number of Fields	= 6
3.1	Field Name	= Superframe_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Slot
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Channel_Offset
3.6	Field Data type Name	= Unsigned8
3.7	Field Name	= Neighbor
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Link options
3.10	Field Data type Name	= Bit Field
3.11	Field Name	= Link_type
3.12	Field Data type Name	= Enumeration

7.3.2.7.8.3 Response variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 967
2	Variable Name	= Link entry

3	Number of Fields	= 7
3.1	Field Name	= Superframe_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Slot
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Channel_Offset
3.6	Field Data type Name	= Unsigned8
3.7	Field Name	= Neighbor
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Link options
3.10	Field Data type Name	= Bit Field
3.11	Field Name	= Link_type
3.12	Field Data type Name	= Enumeration
3.13	Field Name	= Link count
3.14	Field Data type Name	= Unsigned16

7.3.2.7.9 Delete link

7.3.2.7.9.1 Service overview

This service deletes a Link assignment in a network device.

7.3.2.7.9.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 968
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration
4	Qualifier Name	= Slot
5	Data type Name	= Unsigned16
6	Qualifier Name	= Neighbor
7	Data type Name	= Unsigned16

7.3.2.7.9.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 968
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration
4	Qualifier Name	= Slot
5	Data type Name	= Unsigned16
6	Qualifier Name	= Neighbor
7	Data type Name	= Unsigned16

7.3.2.7.9.4 Response variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 968
2	Field Name	= Link count
3	Field Data type Name	= Unsigned16

7.3.2.7.10 Write graph connection

7.3.2.7.10.1 Service overview

This service adds a graph connection assignment to a network device.

7.3.2.7.10.2 Request variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 969
2	Variable Name	= Connection entry
3	Number of Fields	= 2
3.1	Field Name	= Graph_ID
3.2	Field Data type Name	= Unsigned16
3.3	Field Name	= Neighbor_nickname
3.4	Field Data type Name	= Unsigned16

7.3.2.7.10.3 Response variable model

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 969
2	Variable Name	= Connection entry
3	Number of Fields	= 3
3.1	Field Name	= Graph_ID
3.2	Field Data type Name	= Unsigned16
3.3	Field Name	= Neighbor_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Connection count
3.6	Field Data type Name	= Unsigned16

7.3.2.7.11 Delete graph connection

7.3.2.7.11.1 Service overview

This service deletes a graph connection assignment in a network device.

7.3.2.7.11.2 Request parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 970
2	Variable name	= Graph_ID
3	Data type name	= Unsigned16
4	Qualifier Name	= Neighbor_nickname
5	Data type Name	= Unsigned16

7.3.2.7.11.3 Response parameters

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 970
2	Variable name	= Graph_ID
3	Data type name	= Unsigned16
4	Qualifier Name	= Neighbor_nickname
5	Data type Name	= Unsigned16

7.3.2.7.11.4 Delete response variable model**CLASS:** Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 970
2	Field Name	= Connection count
3	Field Data type Name	= Unsigned16

7.3.2.7.12 Write neighbor attribute flag**7.3.2.7.12.1 Service overview**

This service sets attributes of a neighbor to a network device.

7.3.2.7.12.2 Variable model**CLASS:** STRUCTURE VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 971
2	Variable sub index	Not present
3	Variable Name	= Neighbor attribute
4	Number of Fields	= 2
4.1	Field Name	= Neighbor_nickname
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Neighbor flags
4.4	Field Data type Name	= Bit Field

7.3.2.7.13 Write network suspend**7.3.2.7.13.1 Service overview**

This service puts the entire network in a "quiet", non-transmitting state. The command may be sent to all devices on the network or an individual device based on the addressing used.

7.3.2.7.13.2 Variable model**CLASS:** STRUCTURE VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 972
2	Variable sub index	Not present
3	Variable Name	= Network suspend
4	Number of Fields	= 2
4.1	Field Name	= Suspend time
4.2	Field Data type Name	= Unsigned40
4.3	Field Name	= Resume time
4.4	Field Data type Name	= Unsigned40

7.3.2.7.14 Write timetable**7.3.2.7.14.1 Service overview**

This service adds new Timetable to a network device. The network device may start using this Service after processing this command.

7.3.2.7.14.2 Request variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 973
2	Variable Name	= Timetable entry
3	Number of Fields	= 6
3.1	Field Name	= Timetable_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Timetable_type
3.4	Field Data type Name	= Bit Field
3.5	Field Name	= Application domain
3.6	Field Data type Name	= Enumeration
3.7	Field Name	= Peer_nickname
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Service_duration
3.10	Field Data type Name	= Time
3.11	Field Name	= Route_ID
3.12	Field Data type Name	= Unsigned8

7.3.2.7.14.3 Response variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 973
2	Variable Name	= Timetable entry
3	Number of Fields	= 7
3.1	Field Name	= Timetable_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Timetable_type
3.4	Field Data type Name	= Bit Field
3.5	Field Name	= Application domain
3.6	Field Data type Name	= Enumeration
3.7	Field Name	= Peer_nickname
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Service_duration
3.10	Field Data type Name	= Time
3.11	Field Name	= Route_ID
3.12	Field Data type Name	= Unsigned8
3.13	Field Name	= Service count
3.14	Field Data type Name	= Unsigned8

7.3.2.7.15 Write route

7.3.2.7.15.1 Service overview

This service sets or modifies a route for a particular Service in a network device.

7.3.2.7.15.2 Request variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 974
2	Variable Name	= Route entry
3	Number of Fields	= 3

3.1	Field Name	= Route_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Graph_ID
3.6	Field Data type Name	= Unsigned16

7.3.2.7.15.3 Variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 974
2	Variable Name	= Route entry
3	Number of Fields	= 4
3.1	Field Name	= Route_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Graph_ID
3.6	Field Data type Name	= Unsigned16
3.7	Field Name	= Route count
3.8	Field Data type Name	= Unsigned8

7.3.2.7.16 Delete route

7.3.2.7.16.1 Service overview

This service deletes a Route entry in a network device.

7.3.2.7.16.2 Request parameters

CLASS: **Simple Variable Action**

ATTRIBUTES:

1	Numeric Identifier	= 975
2	Variable Name	= Route_ID
3	Data type Name	= Unsigned8

7.3.2.7.16.3 Response parameters

CLASS: **Simple Variable Action**

ATTRIBUTES:

1	Numeric Identifier	= 975
2	Variable Name	= Route_ID
3	Data type Name	= Unsigned8

7.3.2.7.16.4 Response variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 975
2	Field Name	= Route count
3	Field Data type Name	= Unsigned8

7.3.2.7.17 Write source route

7.3.2.7.17.1 Service overview

This service writes a Source route in a network device. Broadcast addresses are not legal address values in this request and are not be included in Source routes or the requests or responses for this command.

7.3.2.7.17.2 Request variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 976
2	Variable Name	= Source route entry
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Route_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Hop count
4.4	Field Data type Name	= Unsigned8
4.5	Field Name	= Hop list
4.6	Field Data type Name	= Array of Hop

CLASS: **Data type**

ATTRIBUTES:

1	Data type Name	= Array of Hop
2	Format	= ARRAY
2.1	Number of Array Elements	= value of Hop count
2.2	Array Element Data type	= Unsigned16

7.3.2.7.17.3 Response variable model

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 976
2	Variable Name	= Source route entry
3	Number of Fields	= 4
4	List of Fields	
4.1	Field Name	= Route_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Hop count
4.4	Field Data type Name	= Unsigned8
4.5	Field Name	= Hop list
4.6	Field Data type Name	= Array of Hop
4.7	Field Name	= Source route count
4.8	Field Data type Name	= Unsigned8

7.3.2.7.18 Delete source route

7.3.2.7.18.1 Service overview

This service deletes a source route entry in a network device.

7.3.2.7.18.2 Request parameters**CLASS:** Simple Variable Action**ATTRIBUTES:**

4	Numeric Identifier	= 977
5	Variable Name	= Route_ID
6	Data type Name	= Unsigned8

7.3.2.7.18.3 Response parameters**CLASS:** Simple Variable Action**ATTRIBUTES:**

4	Numeric Identifier	= 977
5	Variable Name	= Route_ID
6	Data type Name	= Unsigned8

7.3.3 ARs**7.3.3.1 Application relationship ASE****7.3.3.1.1 Overview**

ARs are composed of a set of endpoints of compatible classes. One endpoint of each AR has to be Master class and the other end has to be Slave class. Each device can have only one instance of a Slave class AREP.

A device can use more than one Master class AREP to communicate with several Slave class AREPs. The user at Master class AREP provides the identification of the Slave class AREP as a parameter in the request primitive.

7.3.3.1.2 Master class AREP**7.3.3.1.2.1 Formal model**

ASE: AR ASE
CLASS: Master
CLASS ID: Not used
PARENT CLASS: Top

ATTRIBUTES:

1	(m)	Attribute:	Address

SERVICES:

1	(m)	OpsService:	AR-Get Attribute
2	(m)	OpsService:	AR-Set Attribute

7.3.3.1.2.2 Attributes**Address**

This attribute specifies the network address of master device.

7.3.3.1.3 Slave class AREP**7.3.3.1.3.1 Formal model**

ASE: AR ASE
CLASS: Slave
CLASS ID: Not used
PARENT CLASS: Top

ATTRIBUTES:

1	(m)	Physical layer type	Wireless

1.1 (m) Attribute: Nickname

1.2 (m) Attribute: Unique ID

SERVICES:

1 (m) OpsService: AR-Get Attribute

2 (m) OpsService: AR-Set Attribute

7.3.3.1.3.2 Attributes

Nickname

This attribute specifies the network address of Slave device. The device responds only to this address in an Identify frame received from a Master device.

Unique ID

This attribute is used to form the network address of Slave device. The device responds only to this address in all frames received from a Master device. An all zero value of this attribute shall be reserved as Broadcast address.

7.3.3.1.4 Application relationship ASE service specifications

7.3.3.1.4.1 Overview

This subclause contains the definition of the services that are unique to this ASE. The services are

- AR Get Attribute and
- AR Set Attribute.

7.3.3.1.4.2 AR-Get attribute service

7.3.3.1.4.2.1 Overview

This confirmed service is used to read the value of attributes of an AREP locally.

7.3.3.1.4.2.2 Service parameters

The service parameters for the AR Get Attribute Service are shown in Table 73.

Table 73 – AR get attributes service parameters

Parameter name	Req	Cnf
Argument	M	
Attribute index	M	
Result (+)		S
Value		C
Result (-)		S
Status		C
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter.		

Argument

The argument carries the parameters of the service request.

Attribute index

This parameter identifies the attribute.

Result (+)

This selection type parameter indicates that the request succeeded.

Value

This parameter is returned with the requested attribute value if the request succeeded.

Result (-)

This selection type parameter indicates that the request failed.

Status

If the request failed, Status is returned indicating the reason for failure. The possible values are: Illegal attribute index, Unknown value.

7.3.3.1.4.3 AR Set attributes service**7.3.3.1.4.3.1 Overview**

This confirmed service is used to set the current value of attributes of an AREP locally.

7.3.3.1.4.3.2 Service parameters

The service parameters for the AR Set Attribute Service are shown in Table 74.

Table 74 – AR set attributes service parameters

Parameter name	Req	Cnf
Argument	M	
Attribute index	M	
Value	M	
Result		
Status		M
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter.		

Argument

The argument carries the parameters of the service request.

Attribute index

This parameter identifies the attribute to be updated.

Value

This parameter holds the new attribute value.

Status

As a result of the request, Status is returned indicating OK or the reason for failure. The possible values are: OK, Illegal attribute index, Illegal value.

7.3.4 Summary of classes

This subclause contains a summary of the defined Classes. Table 75 provides a summary of the classes.

Table 75 – Class summary

ASE	Class
Virtual field device	VFD
Data type	Fixed Length Data type
	String Data type
Variable	Simple Variable
	Structure variable

7.3.5 Permitted services by AREP role

Table 76 and Table 77 define the valid combinations of services and AREP class (which service Application protocol data units (APDU) and AREP with the specified class can send or receive).

Table 76 – Confirmed services by AREP class

Services	Master		Slave	
	req	cnf	Ind	rsp
VFD ASE				
Identify	X	X	X	X
Variable ASE				
Read service – all	X	X	X	X
Write service – all	X	X	X	X

Table 77 – Unconfirmed services by AREP class

Services	Slave	Master
	req	ind
Variable ASE		
Information report	X	X

8 Application layer protocol specification – Type 20 elements

8.1 Abstract syntax

The abstract syntax of Application protocol data unit (APDU) is combined with their transfer syntax and is specified in 8.2.

8.2 Transfer syntax

8.2.1 Common APDU fields

8.2.1.1 Command number field

The Command number is the Numeric Identifier of Read (7.3.2.2.3.3), Write (7.3.2.2.3.4), Information report (7.3.2.2.3.5) or Action (7.3.2.3.4) service. It identifies the type of service and the object in the Slave that has to be accessed or action that has to be performed. It is Unsigned16 extended Command number. Commands 0 to 255 are zero filled to form the 16-bit command number (for example Command 9 is transmitted as 0x0009). Command 31 is reserved and shall not be used in the request.

8.2.1.2 Octet count field

This field represents the number of octets in the all of fields that follow this field. The data type of this field is Unsigned8 and its value can be 0 to 255. If the value of this field is zero, then this is the last field in the APDU.

8.2.1.3 Response code field

Response code field is Enumeration data type and it indicates either the successful completion of the requested command, or completion of the requested command with a warning or an error in performing the requested command, as shown in Table 78. Some of the values depend upon the value of Command field. Each service-specific APDU structure shows the response code values assigned to that service. A device is required to use these values. The device can respond with an unused value as long it indicates a condition that is not defined in this standard.

Table 78 – Response code values

Value	Class	Definition
0	Success	Command (read or Write) was executed properly
1 – 127	Warning	Command (Write) was executed with the deviation as described in response (for example a value was set to the nearest legal value)
1 – 127	Error	Command execution was not properly completed and the Response code indicates the reason (for example the device is in Write Protect mode)
1 – 127	Reserved	A code may be reserved for future use by this standard protocol. A device is not permitted to return reserved value

The response code values that are common to all commands and the behavior of the responding device for those codes are specified in Annex E.

8.2.1.4 Application process status

This field is provided by the FAL user. Its data type is one octet long Bit Field. It is not associated with the completion of any command. This field shall have at least following status:

- a) Configuration changed – an operation was performed that changed the device's configuration, and

- b) More status available – more status information is available via another Read Command.

If “Configuration changed” is ‘1’, then the client should read other variables from the server, if necessary and then use “Reset configuration changed flag” service to reset this status bit.

If “More status available” is ‘1’, then the client should use “Read additional device status” service to get the information not included in the “Application process status”.

8.2.1.5 Extended status

This field indicates the VFD status. The possible values of this parameter are shown in the Table 79. Its data type is one octet long Bit Field. It is not associated with the completion of any command.

Table 79 – Extended status values

Code	Value	Description
0x01	Maintenance required	The VFD requires maintenance, although it has not malfunctioned.
0x02	Device variable alert	One or more of the “Device variables” are in an Alarm or Warning state.
0x04	Critical power failure	The available power is becoming critically low; used by devices that operate from stored power

8.2.1.6 Value field

The Value field is the user data which is transferred between application layer and its user. The ALE assembles it from the parameters of a service primitive or parses it into parameters of a service primitive. Its structure depends upon the type of APDU.

8.2.1.7 Formats of APDU structures

The sending application layer prepares an APDU to transfer it to the receiving application layer. It uses the parameters from the service primitives to do so. There are several formats of the APDU:

- request APDU from Master to Slave device,
- normal response from Slave to Master device, and
- command error response from Slave to Master device.

The common formats for APDU structures are specified in 8.2.2. The formats for service-specific APDU structures are specified in 8.2.3, 8.2.4, 8.2.5 and 8.2.6.

The “Command number” field is included in the request and response both. The requester does not have to store the request APDU to associate it with the response APDU.

8.2.2 Common APDU structure

8.2.2.1 Request APDU from master to slave device

All APDU have a common structure as shown in Figure 41.

Command number	Octet count	Value

Figure 41 – Request APDU format from master to slave

For a Read service, Value field is either Null or a Variable Sub Index and for an array type of variable, it may include number of elements to be read. For a Write service, Value field is

Variable Sub Index, if required followed by the value that has to be written to the Slave objects identified by the Command. The ALE assembles it from the parameters of a service primitive. Its structure depends upon the service-specific APDU.

8.2.2.2 Normal response APDU from slave to master device

Figure 42 shows the format of the application layer fields that are used for an error-free response from Slave to Master.

Application process Status	Extended Status	Command number	Octet count	Response code	Value

Figure 42 – Normal response APDU format from slave to master

The Slave sends this APDU in response to a request from Master. The value of Command number field in this APDU is identical to the Command number field in the corresponding request APDU.

The Octet count is number of octets in Value field plus one. The minimum value of Octet count field is one (1).

If the response is for a Read request, then Value field is used for the value of the objects that are read from Slave device. If the response is for a Write request, then this field is used for the value of the objects that were written in the Slave device. In the response to the Write request, it is same as the value received in the request, unless the value received was not permitted and the server wrote a different value. In that case, it is the value written in the Slave device.

The Master shall issue only one command to one Slave until that Slave sends the response or the request times out and becomes invalid (see 7.3.2.2.3.2 and 7.3.2.3.2).

8.2.2.3 Command error APDU from slave to master device

Figure 43 shows the format of the application layer fields that are used for a response from Slave to Master, when there is an error in executing the command. The Value field does not exist.

Application process Status	Extended Status	Command number	Octet count	Response code

Figure 43 – Command error response APDU format from slave to master

The responding device sends this APDU in response to a request from Master. The value of Command number field in this APDU is identical to the Command number field in the corresponding request APDU.

The Octet count field is set to one (1).

Command 31 is reserved and any response to a Command 31 request shall be an error code "Command Not Implemented".

8.2.2.4 Information report APDU from slave to master device

The Information report and data publishing uses the response APDU defined in 8.2.2.2. The publishing is setup to use one of the command numbers (see 8.2.3.32). Other information report command numbers are specified in service-specific APDU subclauses.

8.2.2.5 Command aggregation

This standard allows multiple commands and associated data to be transported in a single request and response APDU. The command aggregation is especially useful when reading device configurations. The aggregated command APDU is shown in Figure 44. The “Command #n Request” field is same as the request APDU as shown in Figure 41. The “Command #n Response” field is the same the response APDU as shown in Figure 42 and Figure 43, except that the “Application process status” and “Extended status” fields are used only once in the beginning and these are not part of “Command #1 Response” to “Command #n Response” fields.

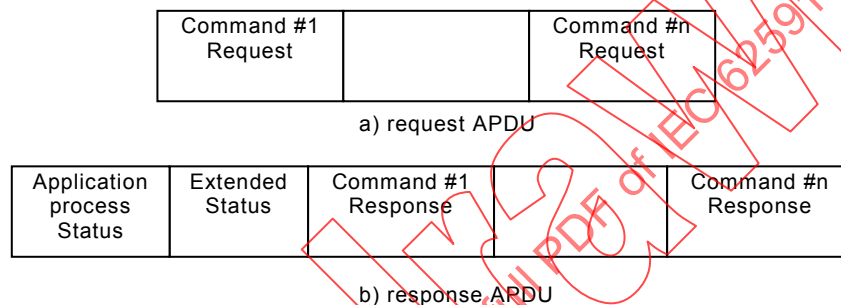


Figure 44 – Aggregated command APDU

If the device has been configured to publish more than one messages (see 8.2.3.32), it can combine these into one message using command aggregation, except that “Publish data Message 0” shall never be aggregated.

Whenever possible, the Type 20 compliant field device shall aggregate multiple pending “Publish data Message” into one transaction using Information report request. If the multiple “Publish data Message” does not fit into a single transaction, then these can be sent in separate transactions.

General requirements for aggregating commands are as following.

- If a device receives an aggregated request APDU, then it shall provide response to all “Command” fields in the request in one aggregated response APDU.
- Command with a number in the range of 0x0300 to 0x03FF shall be aggregated only with another command in this range.
- A read command can be aggregated with write command.

NOTE 1 The mixing of read and write commands in one aggregation is permitted, but it is not recommended.

- The network management configuration APDU (see 8.2.6) shall not be aggregated with any other types. Network management configuration request is acknowledged transaction containing write command that modify network behavior.

NOTE 2 The commands do not have dependency on each other. It implies that the commands can be aggregated. For example several read commands can be aggregated in one transaction. Some restrictions may be imposed in the requirements for specific commands. The protocol does not put any other restriction on the command restriction. A host application (FAL-user) for example device calibration may prevent command aggregation.

8.2.3 Device application service-specific APDU structures

8.2.3.1 Identify FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.2.

8.2.3.2 Read primary variable FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.3.

8.2.3.3 Read loop current and percent of range FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.4.

8.2.3.4 Read dynamic variables and loop current FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.5.

8.2.3.5 Write loop configuration FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.6.

8.2.3.6 Read loop configuration FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.7.

8.2.3.7 Read dynamic variable families classifications FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.8.

8.2.3.8 Read device variables with status FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.9.

8.2.3.9 Read message FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.10.

8.2.3.10 Read tag, descriptor and date FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.11.

8.2.3.11 Read primary variable transducer FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.12.

8.2.3.12 Read device information FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.13.

8.2.3.13 Read final assembly number FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.14.

8.2.3.14 Write message FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.15.

8.2.3.15 Write tag, descriptor and date FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.16.

8.2.3.16 Write final assembly number FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.17.

8.2.3.17 Read long_tag FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.18.

8.2.3.18 Write long_tag FAL PDU

This Command is specified in IEC 61158-6-20:2010, 5.3.19.

8.2.3.19 Reset configuration changed flags FAL PDU

8.2.3.19.1 Request primitive

The value of Command is thirty-eight (38); the value of Octet count field is two (2) and Value field is as shown in Table 80.

Table 80 – Reset configuration changed flags value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Configuration change counter	Total number of configuration changes

8.2.3.19.2 Response primitive

The Value field of response is shown in Table 80. The Response code values are shown in Table 81.

Table 81 – Reset configuration changed flag command specific Response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command error
7	Error	In write protect mode
9	Error	Configuration change counter mismatch
16	Error	Access restricted

8.2.3.19.3 Procedure at responding device

A field device shall contain “Configuration changed” bits separately for each Master – primary and the secondary and the gateway. All “Configuration changed” bits shall be set to ‘1’ when any configuration item in the field device is modified. This command shall reset the “Configuration changed” bit to ‘0’, which is associated with the Master device issuing this command. That bit shall stay at ‘0’ value, until another change is made to the device's configuration. The value of this bit shall be maintained even if power is removed from the device or a device reset is performed. “Configuration changed” bit of the “Application process status” octet for the Master device not issuing this command is unaffected by this command and remains set to ‘1’ until the other Master device issues this command.

Upon receiving this command the device shall compare the Configuration change counter received in this command with the current value of this parameter in the device. If they do not match then the device shall return "Configuration Change Counter Mismatch" and not reset the "Configuration changed" bit.

8.2.3.20 Perform self test FAL PDU

8.2.3.20.1 Request primitive

The value of Command is forty-one (41); the value of Octet count field is zero (0) and Value field is null.

8.2.3.20.2 Response primitive

The Value field of response is null. The Response code values are shown in Table 82.

Table 82 – Perform self test command specific Response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command error
16	Error	Access restricted
32	Error	Busy

8.2.3.20.3 Procedure at responding device

This command initiates the self test function in the device. The execution of this command may take a relatively long time to complete. During the self test the field device shall continue to communicate. However, it is allowed to respond 'Busy' to all commands received during this interval.

The Loop Current may not reflect the process while the self test is executing. A Master shall not generate spurious error messages or disconnect from a device while the self test is in progress in that device.

8.2.3.21 Perform device reset FAL PDU

8.2.3.21.1 Request primitive

The value of Command is forty-two (42); the value of Octet count field is zero (0) and Value field is null.

8.2.3.21.2 Response primitive

The Value field of response is null. The Response code values are shown in Table 83.

Table 83 – Perform device reset command specific Response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command error
16	Error	Access restricted
32	Error	Busy

8.2.3.21.3 Procedure at responding device

The device shall respond to this command immediately and within the slave time out and then reset itself. This is equivalent to cycling the power off and then back on to the device.

The execution of this command may take a relatively long period of time to complete. The device may not respond to subsequent commands until the reset is complete. The device may not respond while it is in the process of reset. This may look like communications with the device is lost. A Master shall not generate spurious error messages or disconnect from the device while the reset is in progress in that device.

8.2.3.22 Read additional device status FAL PDU

8.2.3.22.1 Request primitive

The value of Command is forty-eight (48); the value of Octet count field is zero (0) and Value field is null.

8.2.3.22.2 Response primitive

The Value field of response is shown in Table 84. The Response code values are shown in Table 85.

Table 84 – Read additional device status value field

Octet offset	Data type	Parameter name	Description
0	Bit Field (total of 6)	Device specific status 0	Manufacturer and device type specific status
6	Bit Field	Extended status	Extended operational status of the VFD
7	Bit Field	Device operating mode	Operational mode of the device
8	Bit Field	Standardized status 0	Status of the device as specified in Table F.8
9	Bit Field	Standardized status 1	Status of the device as specified in Table F.9
10	Bit Field	Analog channel saturated	Status of the analog channel as specified in Table F.6
11	Bit Field	Standardized status 2	Status of the device as specified in Table F.10
12	Bit Field	Standardized status 3	Status of the device as specified in Table F.11
13	Bit Field	Analog channel fixed	Status of the analog channel as specified in Table F.7
14	Bit Field (1 to 11)	Device specific status 1	Manufacturer and device type specific status

NOTE The Device specific status 0 and Device specific status 1 can be either Bit Field or Enumeration data types.

Table 85 – Read additional device status command specific Response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command error
8	Warning	Update in progress
16	Error	Access Restricted

8.2.3.22.3 Procedure at responding device

If the device is able to respond to the request, but the status information is pending the completion of a command that requires a relatively long time to complete, then Response

code value of “Update in progress” shall be returned. The Client should retry the request after a delay.

Else, the device shall respond indicating ‘Success’ and include all Value fields supported by the device's implementation of Command 48 in the response.

The Response Value fields returned are truncated after the last status octet supported by the device. All devices shall support at least octet numbers 0 to 8 including “Extended status”, Device operating mode and Standardized status 0. If the device supports more than one Analog channel, then octet numbers 9 to 13 shall also be returned.

8.2.3.23 Reset more status available FAL PDU

8.2.3.23.1 Request primitive

The value of Command is forty-eight (48); the value of Octet count field is nine (9) to twenty-five (25) depending upon the device implementation. The Value field is as shown in Table 86.

Table 86 – Read additional device status value field

Octet offset	Data type	Parameter name	Description
0	Bit Field (total of 6)	Device specific status 0	Manufacturer and device type specific status
6	Bit Field	Extended status	Extended operational status of the VFD
7	Bit Field	Device operating mode	Operational mode of the device
8	Bit Field	Standardized status 0	Status of the device as specified in Table F.8
9	Bit Field	Standardized status 1	Status of the device as specified in Table F.9
10	Bit Field	Analog channel saturated	Status of the analog channel as specified in Table F.6
11	Bit Field	Standardized status 2	Status of the device as specified in Table F.10
12	Bit Field	Standardized status 3	Status of the device as specified in Table F.11
13	Bit Field	Analog channel fixed	Status of the analog channel as specified in Table F.7
14	Bit Field (1 to 14)	Device specific status 1	Manufacturer and device type specific status
NOTE The Device specific status 0 and Device specific status 1 can be either Bit Field or Enumeration data types.			

8.2.3.23.2 Response primitive

The Value field of response is shown in Table 86. The Response code values are shown in Table 85.

Table 87 – Read additional device status command specific Response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command error
8	Warning	Update in progress
16	Error	Access Restricted

8.2.3.23.3 Procedure at responding device

If the device is able to respond to the request, but the status information is pending the completion of a command that requires a relatively long time to complete, then Response code value of “Update in progress” shall be returned. The Client should retry the request after a delay.

Else, it shall compare the received values to the current values contained in the device. If there is an exact match then the “More status available” bit corresponding to the requesting device (primary master, secondary master or gateway) shall be reset to ‘0’. If there is not an exact match or too few octets are received then the “More status available” bit shall remain unchanged. If the request contains more Value fields than supported by the device, then these extra fields shall be ignored. The device shall always send the current values of Value fields in the response.

The Response Value fields returned are truncated after the last status octet supported by the device. All devices shall support at least octet numbers 0 to 8 including “Extended status”, Device operating mode and Standardized status 0. If the device supports more than one Analog channel, then octet numbers 9 to 13 shall also be returned.

8.2.3.24 Read device variable information FAL PDU

8.2.3.24.1 Request primitive

The value of Command is fifty-four (54); the value of Octet count field is one (1). The Value field is as shown in Table 88.

Table 88 – Read device variable information value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Variable code	Variable for which information is requested

8.2.3.24.2 Response primitive

The Value field of response is shown in Table 89. The Response code values are shown in Table 90.

Table 89 – Read device variable information response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Variable code	Code received in request at Value field
1	Unsigned24	Transducer serial number	Value of Transducer Serial Number
4	Enumeration	Transducer unit	Code for Transducer Limits and Minimum Span Units
5	Float32	Upper transducer limit	Value of Upper Transducer Limit variable
9	Float32	Lower transducer limit	Value of Lower Transducer Limit variable
13	Float32	Damping value	Damping constant for the transducer value
17	Float32	Minimum span	Value of Minimum Span variable
21	Enumeration	Device variable classification	Function performed by the device variable as specified in Table F.5
22	Enumeration	Device variable family	Device family as specified in Table F.4
23	Time	Update period	The maximum period between Device variable updates

Table 90 – Read device variable information command specific Response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too Few data octets received
6	Error	Device-specific command error
16	Error	Access restricted
32	Error	Busy

8.2.3.24.3 Procedure at responding device

If there no transducer for the selected Deice variable, then the device shall set Transducer serial number to zero, set other Float32 parameters to "0x7F, 0xA0, 0x00, 0x00", and Transducer units parameters to "250" (not used) when they are not applicable.

If the Device variable classification is not supported by this Device variable then the device shall return "0" (not implemented). If the Device variable family is not supported by the Device variable then the device shall return "250" (not used)

The device shall update the Device variable at least once in the interval indicated by the Update period.

8.2.3.25 Write device variable FAL PDU**8.2.3.25.1 Request primitive**

The value of Command is seventy-nine (79); the value of Octet count field is eight (8) and Value field is as shown in Table 91.

Table 91 – Write device variable value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Variable code	is the Variable for which values have to be written
1	Enumeration	Device variable command	is the Command to be executed as specified in Table F.3
2	Engineering unit	Device variable unit	Measurement unit of the variable
3	Float32	Device variable value	Variable value to be written
7	Bit Field	Device variable status	Variable status to be written

8.2.3.25.2 Response primitive

The Value field of response is shown in Table 91. The Response code values are shown in Table 92.

Table 92 – Write device variable command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific Command Error
7	Error	In write protect mode
8	Warning	Device family status bit not set
10	Error	Invalid Device variable code
16	Error	Access Restricted
17	Error	Invalid Device variable index; the Device variable does not exist in the device
18	Error	Invalid Units code
19	Error	Device variable index not allowed for this command
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.3.25.3 Procedure at responding device

The Write Device variable command controls the operation of the device either forcing the engineering value, or allowing normal operation. For Device variables automatically calculated by the field device, normal operation shall be resumed when power is removed from the device and re-applied to it, or upon completing a device reset.

A field device shall start a DR (delayed response), if the change in the Device variable's value exceeds the rate-of-change limit. This DR shall continue until the new value can be reached. While the Device variable remains forced, Response code 8 – "Update failure" shall be returned, whenever the forced value is read by a client. This signals the master to detect this condition. Forcing a Device variable value may affect bits in the "Application process status" octet.

Host application should not use Device family status bits for Device variables not supporting a Device family (i.e., the field device always reset these bits in this case).

The Device variable unit code received with this command does not affect the Device variable units of the field device. The Device variable value shall be returned in the same units as received.

8.2.3.26 Read real-time clock FAL PDU

8.2.3.26.1 Request primitive

The value of Command is ninety (90); the value of Octet count field is zero (0) and Value field is null.

8.2.3.26.2 Response primitive

The Value field of response is shown in Table 93. The Response code values are shown in Table 94.

Table 93 – Read real-time clock value field

Octet offset	Data type	Parameter name	Description
0	Date	Date	Current date
3	Time	Current time	Current time of day
7	Date	Date last set	Date clock last set
10	Time	Time last set	Time clock last set
14	Bit Field	RTC flags	RTC flags as specified in Table F.18

Table 94 – Read real-time clock command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command error

8.2.3.26.3 Procedure at responding device

The device shall respond with the internal time at which the request was received. If the clock has not been set then the last time set shall be initialized to midnight (00:00) 01 January, 1900.

8.2.3.27 Write publish data period FAL PDU**8.2.3.27.1 Request primitive**

The value of Command is one hundred and three (103); the value of Octet count field is nine (9) and Value field is as shown in Table 95.

Table 95 – Write publish data period value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Publish data message ref	Reference to one set of publish data in all read and write requests and responses for that set
1	Time	Update period	The duration between two successive publication of the specified data, if the trigger conditions in Command 104 are met
5	Time	Maximum update period	The duration between two successive publication of the specified data, if the trigger conditions in Command 104 are not met
NOTE The value returned in the response data octets reflects the value actually used by the device.			

8.2.3.27.2 Response primitive

The Value field of response is shown in Table 95. The Response code values are shown in Table 96.

Table 96 – Write publish data period command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Update times adjusted
9	Error	Invalid Device variable code
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.3.27.3 Procedure at responding device

This request selects the minimum and maximum update period of a Publish data message. The minimum time shall be less than or equal to the maximum time. Only following values of Update period parameter are allowed:

0,100 s,
0,250 s,
0,500 s,
1 s,
2 s,
4 s,
8 s,
16 s,
32 s, and
any value from 60 to 3 600 s.

Field devices shall correct settings differing from these values to the nearest allowed value and indicate "Update times adjusted" in its response code. The default value of Update period parameter is 60 s.

If multiple Publish data messages are enabled and they all coincide within the same time, then these are aggregated together using Command 78 to form one APDU. The Publish data messages with the highest priority will be concatenated first up to the maximum number of octets that the respective transport layer allows. The Publish data message with the lower reference number has the higher priority over the Publish data message with the higher reference number. The "Application process status" event has lower priority than the Publish data messages and has higher priority than Command 48 event.

8.2.3.28 Write publish data trigger FAL PDU**8.2.3.28.1 Request primitive**

The value of Command is one hundred and four (104); the value of Octet count field is eight (8) and Value field is as shown in Table 97.

Table 97 – Write publish data trigger field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Publish data message ref	Reference to one set of publish data in all read and write requests and responses for that set
1	Enumeration	Trigger mode	Trigger mode selection code as specified in Table F.12
2	Enumeration	Device variable classification	Function performed by the device variable as specified in Table F.5
3	Engineering unit	Device variable unit	Measurement unit of the variable
4	Float32	Trigger level	Window or the level that is used to trigger the publishing
NOTE The value returned in the response data octets reflects the value actually used by the device.			

8.2.3.28.2 Response primitive

The Value field of response is shown in Table 97. The Response code values are shown in Table 98.

Table 98 – Write publish data trigger command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
3	Error	Parameter value to be written is larger than permitted
4	Error	Parameter value to be written is smaller than permitted
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
9	Error	Invalid Publish data message ref
11	Error	Invalid Device variable classification
12	Error	Invalid units code
13	Error	Invalid publish data trigger mode selection code
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.3.28.3 Procedure at responding device

Unless otherwise configured by this request, a Publish data message shall assume a Trigger mode of "Continuous" and a Trigger value of 0,00. Furthermore, if a device specific command is specified as the publish command then this request shall return "Access restricted" and shall use the default value of "Continuous" for Trigger mode.

These trigger modes allow the device to be configured to defer the publishing of the Publish data message beyond the Update period specified in Command 103 (see Table 95). In all cases, the Publish data message shall be triggered when the Maximum update time in Command 103 (see Table 95) is exceeded. The trigger source is specified and is normally the

first process value returned as response to the command as shown in Table 99. The command number for Publish data message is specified in 8.2.3.32.

Table 99 – Publish data message trigger source

Command	Reference	Trigger source
1	8.2.3.2	Primary Variable value
2	8.2.3.3	Primary Variable in percent of range
3	8.2.3.4	Primary Variable
9	8.2.3.8	Device variable for slot 0

When in "Windowed" mode, the Trigger value shall be a positive number and is the symmetric window around the last communicated value. Figure 45 illustrates the Publish data message being published after the window was exceeded.

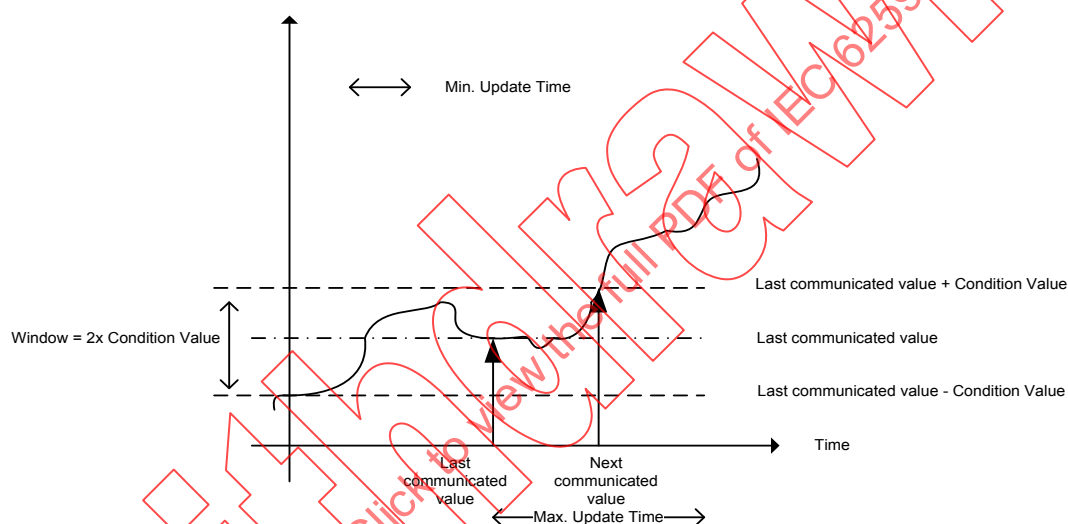


Figure 45 – Windowed trigger mode

Figure 46 illustrates that even if the value does not deviate beyond the window a Publish Data Message is still published when the Max Update Period is exceeded.

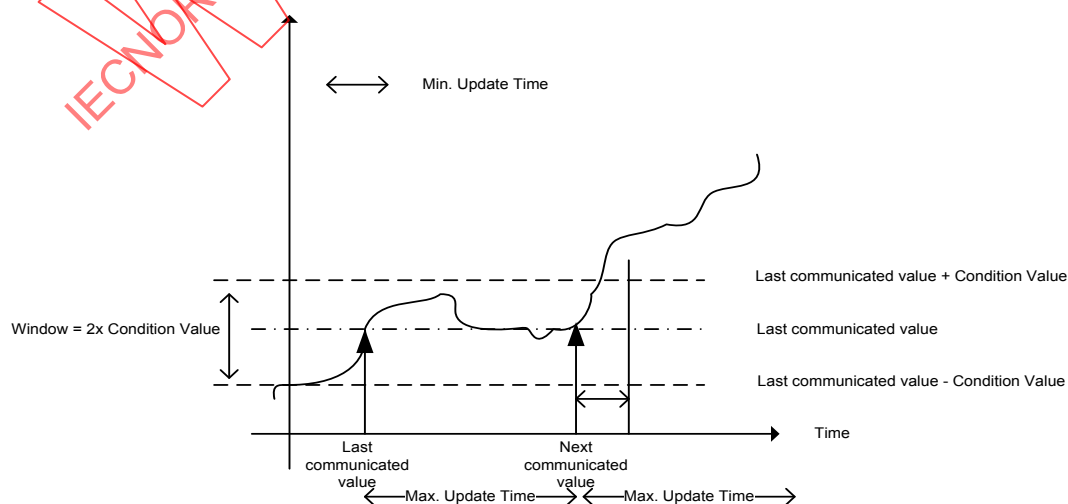


Figure 46 – Windowed trigger mode with maximum update time expired

In "Rising" mode, the Publish Data Message shall be published when the source value is above the threshold established by the trigger value. Figure 47 shows that the update time changes once the limit is exceeded. Publish Data messages are published at the rate indicated by the Update period as long as the source value remains above the trigger value. If the value falls below the trigger value, the update time shall be the Maximum update period.

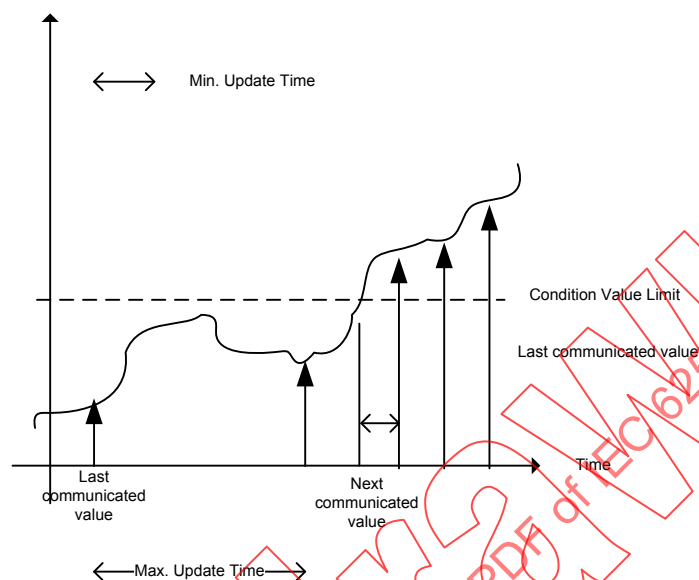


Figure 47 – Level trigger mode

In "Falling" mode is similar to "Rising" mode, the Publish Data Message shall be published when the source value is below the threshold established by the trigger value.

8.2.3.29 Read publish data mode configuration FAL PDU

8.2.3.29.1 Request primitive

The value of one hundred and five (105); the value of Octet count field is zero (0) or one (1). If the request has one octet, then the request Value field is shown in Table 100.

Table 100 – Read publish data mode configuration request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Publish data message ref	This is used to refer to one set of publish data in all read and write requests and responses for that set

8.2.3.29.2 Response primitive

The Value field of response is shown in Table 101. The Response code values are shown in Table 102.

Table 101 – Read publish data mode configuration response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Publish data mode control	Publish data mode control code as specified in Table F.2
1	Unsigned8	Command number expansion	31 (0x1F) or LSB of the Publish data command number – see Note
2	Unsigned8	Device variable code 0	Device variable code assigned to Slot 0
3	Unsigned8	Device variable code 1	Device variable code assigned to Slot 1
4	Unsigned8	Device variable code 2	Device variable code assigned to Slot 2
5	Unsigned8	Device variable code 3	Device variable code assigned to Slot 3
6	Unsigned8	Device variable code 4	Device variable code assigned to Slot 4
7	Unsigned8	Device variable code 5	Device variable code assigned to Slot 5
8	Unsigned8	Device variable code 6	Device variable code assigned to Slot 6
9	Unsigned8	Device variable code 7	Device variable code assigned to Slot 7
10	Unsigned8	Publish data message ref	Reference to one set of publish data in all read and write requests and responses for that set
11	Unsigned8	Max publish count	Maximum number of Publish data messages supported by the device
12	Unsigned16	Extended command number	Extended command number (see 8.2.2.2)
14	Time	Update period	The duration between two successive publication of the specified data, if the trigger conditions in Command 104 are met
18	Time	Maximum update period	The duration between two successive publication of the specified data, if the trigger conditions in Command 104 are not met
22	Enumeration	Trigger mode	Trigger mode selection code as specified in Table F.12
23	Enumeration	Device variable classification	Function performed by the device variable as specified in Table F.5
24	Engineering unit	Device variable unit	Measurement unit of the variable
25	Float32	Trigger level	Window or the level that is used to trigger the publishing
NOTE If the request has one octet, then this parameter is set to 31.			

Table 102 – Read publish data mode configuration command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
9	Error	Invalid Publish data message ref
32	Error	Busy – a delayed response could not be started

8.2.3.29.3 Procedure at responding device

If the device receives a Request without data octets, it shall return the configuration of Publish data message '0' and the LSB of the publish command number shall be returned in response data octet one (1).

If a slot is not configured to transmit a Device variable, that slot shall return "250" (not used) as Device variable code. If command 9 is to be published, then that slot's Device variable code shall meet the requirements according to command 9 (see 8.2.3.8).

If a device implements this Publish data mode, then it shall support at least three Publish data sets i.e. the minimum value of Max publish count shall be three (3).

8.2.3.30 Flush delayed responses FAL PDU

8.2.3.30.1 Request primitive

The value of Command is one hundred and six (106); the value of Octet count field is zero (0) and Value field is null.

8.2.3.30.2 Response primitive

The Value field of response is null. The Response code values are shown in Table 103.

Table 103 – Flush delayed responses command specific Response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command error
8	Warning	All but running delayed responses flushed
16	Error	Access restricted
32	Error	Busy

8.2.3.30.3 Procedure at responding device

This command clears all pending delayed responses for the Master that issued the command. Delayed responses currently running that cannot be interrupted or aborted may be completed. If this is the case, then the Slave shall respond with “All but running delayed responses flushed”.

Devices should support a minimum of two DR buffers (one for each Master). If only one DR buffer is supported then this command shall flush the DR buffer even if it is in use by the other Master that did not send the flush request.

8.2.3.31 Write publish data device variables FAL PDU

8.2.3.31.1 Request primitive

The value of one hundred and seven (107); the value of Octet count field is either (5), or nine (9) and Value field is as shown in Table 104.

Table 104 – Write publish data mode configuration value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Device variable code 0	Device variable code assigned to Slot 0
1	Unsigned8	Device variable code 1	Device variable code assigned to Slot 1
2	Unsigned8	Device variable code 2	Device variable code assigned to Slot 2
3	Unsigned8	Device variable code 3	Device variable code assigned to Slot 3
4	Unsigned8	Device variable code 4	Device variable code assigned to Slot 4
5	Unsigned8	Device variable code 5	Device variable code assigned to Slot 5
6	Unsigned8	Device variable code 6	Device variable code assigned to Slot 6
7	Unsigned8	Device variable code 7	Device variable code assigned to Slot 7
8	Unsigned8	Publish data message ref	Reference to one set of publish data in all read and write requests and responses for that set
NOTE The value returned in the response data octets reflects the value actually used by the device.			

8.2.3.31.2 Response primitive

The Value field of response is shown in Table 104. The Response code values are shown in Table 105.

Table 105 – Write publish data device variables command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Publish data condition conflict
9	Error	Invalid Publish data message ref

8.2.3.31.3 Procedure at responding device

The Device variables codes received in the request shall be used by a publishing device for a Command 9 or 33 in Publish data mode. The master shall always send either five (5) or nine (9) data octets in the request. If any of the Device variables codes starting with slot 1 is '250' (not used), then the Command 9 or Command 33 response shall be truncated starting with the first slot that has a value of '250' (not used). For command 33 only first four slots shall be used. If the request has only five (5) data octets, then Device variables code for slots 4 to 7 shall be set to '250' (not used).

If the Trigger mode (see Table 97) for the Publish data message ref in this request is non-zero and the trigger source's Device variable classification does not match the new Device variable code assigned to Slot 0, then the value from this request shall be accepted and Response code "Publish data condition conflict" shall be returned; the field device shall correct the classification, unit codes, set Trigger mode to publish continuously at the Update period until it receives another Command 104.

8.2.3.32 Write publish data mode command number FAL PDU

8.2.3.32.1 Request primitive

The value of one hundred and eight (108); the value of Octet count field is either three (3) as shown in Table 106 or one (1) as shown in Table 107.

Table 106 – Write publish data mode command number value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Publish command number	16-bit Command Number of the response message to be transmitted in Publish data mode
2	Unsigned8	Publish data message ref	Reference to one set of publish data in all read and write requests and responses for that set
NOTE The value returned in the response data octets reflects the value actually used by the device.			

Table 107 – Write publish data mode command number value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Publish command number	8-bit Command Number of the response message to be transmitted in Publish data mode
NOTE The value returned in the response data octets reflects the value actually used by the device.			

8.2.3.32.2 Response primitive

The Value field of response is shown in Table 106 and Table 107. The Response code values are shown in Table 108.

Table 108 – Write publish data mode command number command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Publish data condition conflict
9	Error	Invalid Publish data message ref

8.2.3.32.3 Procedure at responding device

The permitted values for Publish command number are {1, 2, 3, 9, 33, 48}. All devices shall support all of these values except 33. If the device implements the optional command 33, then it shall also support this value of Publish command number. If a field device receives only one data octet in the request, it shall assume the value of Publish data message ref is '0'; and it shall treat the request as a single octet command number and respond with only a single octet in the response.

If the Trigger mode (see Table 97) for the Publish data message ref in this request is non-zero and the trigger source's Device variable classification does not match the command number in this request, then the value from this request shall be accepted and Response code "Publish data condition conflict" shall be returned; the field device shall correct the

classification, unit codes, set Trigger mode to publish continuously at the Update period until it receives another Command 104.

After the publishing is enabled (see 8.2.3.33), the device shall publish the response to “Publish command number” using Information report service.

8.2.3.33 Write publish data mode control FAL PDU

8.2.3.33.1 Request primitive

The value of one hundred and nine (109); the value of Octet count field is either one (1) – the first field, or two (2) – both fields, as shown in Table 109.

Table 109 – Write publish data mode control value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Publish data mode control	Publish data mode control code as specified in Table F.2
1	Unsigned8	Publish data message ref	Reference to one set of publish data in all read and write requests and responses for that set

8.2.3.33.2 Response primitive

The Value field of response is shown in Table 109. The Response code values are shown in Table 110.

Table 110 – Write publish data mode control command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Update period increased
9	Error	Insufficient bandwidth
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started.
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.3.33.3 Procedure at responding device

This command is used to enable or disable the Publish data mode for devices. This command affects data-link layer operation. All data-link layer requirements for entering and exiting Publish Data mode shall be met.

If the request is for enabling the publishing, then the responding device shall perform the following procedure.

- If the responding device does not have capacity currently scheduled for publishing the data set specified for Publish data message ref, then it shall respond with “DR initiated”.
- The device shall send “Request timetable FAL PDU” request (see 8.2.4.31.1) to the network manager for publishing data. The network manager is expected to send response to “Request timetable FAL PDU” with “DR initiated” and to start processing the request. The network manager then:
 - allocates the network resources for the publish service;
 - if required, sends “Write superframe FAL PDU” request (see 8.2.6.6.1) to set up a superframe; and
 - sends “Write link FAL PDU” request (see 8.2.6.8.1) to set up a Link within the superframe.

NOTE 1 The network manager also sends request to other network devices to add superframe and Link for publishing.

- In most cases, the network manager sets up links in an existing superframe to satisfy this request. Once the network has been configured, the network manager returns a response to the “Request timetable FAL PDU” indicating that the request has been completed.
- After receiving this response the device shall send a response indicating success.
- If the response from the network manager to the “Request timetable FAL PDU” indicates that the request cannot be met, then the device shall respond indicating “Insufficient bandwidth”.
- If the responding device has sufficient capacity currently scheduled for publishing the data set specified for Publish data message ref, then the device shall send a response indicating success.

If the request is for disabling the publishing then the responding device shall perform the following procedure.

- The responding device shall respond with “Success”.
- The device shall send “Delete timetable FAL PDU” request (see 8.2.4.33.1) to the network manager. The network manager is expected to start processing the request. The network manager then:
 - de-allocates the network resources for the publish service;
 - sends “Delete link FAL PDU” request (see 8.2.6.9.1) to remove the Link from the superframe; and
 - if required, sends “Delete superframe FAL PDU” request (see 8.2.6.7.1) to remove the superframe.

NOTE 2 The network manager also sends request to other network devices to delete superframe and Link that was used for publishing.

- Once the network resources have been de-allocated, the network manager returns a response to “Request timetable FAL PDU” indicating that the request has been completed.

If the device is denied the required network capacity to publish, it shall set the bit “Capacity denied” in the Standardized status (see 8.2.3.22.2 – command 48).

If there is only one octet in the request, then:

- If the device is build according to this standard and the value of this field is neither ‘0’ nor ‘1’, then the device shall respond with “Too few data octets received”;
- Else, the device shall assume that Publish data message ‘0’ is being activated or deactivated for publishing.

8.2.3.34 Read event notification summary FAL PDU

8.2.3.34.1 Request primitive

The value of one hundred and fifteen (115); the value of Octet count field is one (1) as shown in Table 111.

Table 111 – Read event notification summary request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Event number ref	This is used to refer to one set of event data in all read and write requests and responses for that set

8.2.3.34.2 Response primitive

The Value field of response is shown in Table 112. The Response code values are shown in Table 113.

Table 112 – Read event notification summary response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Event number ref	Reference to one set of event data in all read and write requests and responses for that set
1	Unsigned8	Maximum event count	Maximum number of Event data messages supported by the device
2, bit 7	Bit Field	Event status	Event status as specified in Table F.14 – bits 7-4 at octet offset 2
2, bit 3	Bit Field	Event notification control	Event notification control code as specified in Table F.13 – bits 3-0 at octet offset 2
3	Time	Event occurrence time	Time when first unacknowledged event was triggered; 0xFFFFFFFF if no pending event
7	Time	Event notification retry time	Event notification retry time
11	Time	Maximum update time	Maximum update time
15	Time	Event de-bounce interval	Event de-bounce interval
19	Array of Bit Field (1 to 26)	Event mask	The concatenation of the mask for triggering on bits set to '1' in "Application process status" and command 48 response

Table 113 – Read event notification summary command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error

8.2.3.34.3 Procedure at responding device

This command reads the configuration of the Event notification. If the device supports fewer than maximum number of status fields described in command 48 responses (see Table 84), then the response to this request shall include Event mask for only those fields that are supported by the device.

8.2.3.35 Write event notification bit mask FAL PDU

8.2.3.35.1 Request primitive

The value of one hundred and sixteen (116); the value of Octet count field is two (2) to twenty-eight (28) as shown in Table 114.

Table 114 – Write event notification bit mask value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Event number ref	Reference to one set of event data in all read and write requests and responses for that set
1	Array of Bit Field (1 to 26)	Event mask	The concatenation of the mask for triggering on bits set to '1' in "Application process status" and command 48 response

8.2.3.35.2 Response primitive

The Value field of response is shown in Table 114. The Response code values are shown in Table 115.

Table 115 – Write event notification bit mask command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
16	Error	Access restricted

8.2.3.35.3 Procedure at responding device

This command sets the Event mask that is used to trigger an Event notification. For each bit set to '1' in the Event mask, any change in the corresponding "Application process status" or Command 48 response value triggers an event notification. If the number of Event mask fields in the request is smaller than the Event mask fields supported by the device, then unspecified mask bits shall be considered as '0'.

The device shall retain Event mask through a device reset, self test or the power being removed and reapplied.

8.2.3.36 Write event notification timing FAL PDU

8.2.3.36.1 Request primitive

The value of Command is one hundred and seventeen (117); the value of Octet count field is thirteen (13) and Value field is as shown in Table 116.

Table 116 – Write event notification timing value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Event number ref	Reference to one set of event data in all read and write requests and responses for that set
1	Time	Event notification retry time	Even notification retry time
5	Time	Maximum update time	Maximum update time
9	Time	Event de-bounce interval	Event de-bounce interval
NOTE The value returned in the response data octets reflects the value actually used by the device.			

8.2.3.36.2 Response primitive

The Value field of response is shown in Table 116. The Response code values are shown in Table 117.

Table 117 – Write event notification timing command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Update time or de-bounce interval adjusted
9	Error	Invalid event notification timing request
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started.
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.3.36.3 Procedure at responding device

This command sets the timing parameters for event notification. If the Event notification retry time is not less than or equal to the maximum update time, then the device shall adjust it. If necessary, the devices shall correct other timing values received in the request and respond with “Update time or de-bounce interval adjusted”.

The Event de-bounce interval sets the minimum time duration over which the status bit shall remain changed (i.e., the time the event shall persist) in a device before the Event notification is sent. The device shall scan for possible events at least twice during one Event de-bounce interval.

If there was no event generated during a Maximum update time, the device shall publish the Command 119 response one time to the gateway using Information report request.

The device shall retain Event notification timing parameters through a device reset, self test or the power being removed and reapplied.

8.2.3.37 Write event notification control FAL PDU

8.2.3.37.1 Request primitive

The value of one hundred and eighteen (118); the value of Octet count field is two (2) as shown in Table 118.

Table 118 – Write event notification control value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Event number ref	Reference to one set of event data in all read and write requests and responses for that set
1	Bit Field	Event notification control	Event notification control code as specified in Table F.13

8.2.3.37.2 Response primitive

The Value field of response is shown in Table 118. The Response code values are shown in Table 119.

Table 119 – Write event notification control command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Update times adjusted
14	Warning	Update time uncertain – only allowed when not connected to the network
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started.
33	Error	DR initiated – only allowed when enabling events in a device connected to a network according to this standard
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.3.37.3 Procedure at responding device

This command is used to enable or disable Event notification. If a Type 20 compliant field device joins the network and is denied the required capacity for Event notification, it shall set the bit 'Capacity denied' in the Standardized status 3, see Table F.11. Capacity can be denied due to Publish data or Event notification demands.

The device shall retain Event notification mode control through a device reset, self test or the power being removed and reapplied.

8.2.3.38 Write event notification acknowledgement FAL PDU

8.2.3.38.1 Request primitive

The value of one hundred and eighteen (119); the value of Octet count field is either one (1) or eight (8) to thirty-three (33) as shown in Table 120. If there is only one octet then it shall be Event number ref.

Table 120 – Write event notification acknowledgement value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Event number ref	Reference to one set of event data in all read and write requests and responses for that set
1	Time	Event occurrence time	Time when first unacknowledged event was triggered; '-1' when no events are pending
5	Unsigned16	Configuration change counter	Configuration change counter
7	Bit Field	Application process status	Application process status (see 8.2.1.4)
8	Array of Bit Field (1 to 25)	Additional device status	Command 48 data, see Note
NOTE Contains latched data and may not match current Command 48 response data.			

8.2.3.38.2 Response primitive

The Value field of response is shown in Table 121. The Response code values are shown in Table 121.

Table 121 – Write event notification acknowledgement command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received; generated only if the event number is not in the request data
6	Error	Device-specific command Error
8	Warning	Not all events cleared

8.2.3.38.3 Procedure at responding device

This command is used to acknowledge the Event notification. The device shall compare parameters values in the received acknowledgement to the current values for the same parameters in the device – Event occurrence time, Configuration change counter, “Application process status”, and Additional device status. If all parameter values are equal, then the device shall respond with “Success” and clear the Event Status (see Table 112) bits to ‘0’. Since there may be multiple time-stamped bit transitions, the command shall only clear the oldest one. If not all bits are cleared to ‘0’, then the device shall respond with “Not all events cleared”. Although, this is a write command, but it shall not be affected by any write protect mechanism. It is essential that Event notifications can be cleared at any time.

The command is truncated after the last octet received when setting the Event mask (see Command 116 in 8.2.3.35). If any octets beyond those provided in Command 116 are received in this command request, they shall be ignored and not returned in the command

response. The latest values for “Application process status”, the Configuration change counter and the Command 48 response octets are always included in the Command 119.

This command may be issued with only the Event number ref in the request data. If this occurs, the device shall return the current latched event data as shown in Table 120 and shall not clear the Event Status bits to ‘0’.

The device shall store at least three (3) response messages (the current event being published using Command 119 and 2 more that are buffered). It is highly recommended that five (5) transitions be buffered. It is not necessary to save these through power cycles.

8.2.4 Layer management service specific APDU structures

8.2.4.1 Write Join_key FAL PDU

8.2.4.1.1 Request primitive

The value of Command is seven hundred and sixty-eight (768); the value of Octet count field is one (1) and value field is as shown in Table 122.

Table 122 – Write Join_key value field

Octet offset	Data type	Parameter name	Description
0	Security key	Join_key	This is 128-bit network key that is used for security

8.2.4.1.2 Response primitive

The Value field of response is shown in Table 122. The Response code values are shown in Table 123.

Table 123 – Write Join_key command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict
65	Error	Key change failed

8.2.4.1.3 Procedure at responding device

This command allows a local maintenance tool (for example a handheld) to modify a network device's Join_key. If a device is connected to the network and it receives this command via the maintenance port, then the device shall respond with "Access Restricted".

8.2.4.2 Read join status FAL PDU

8.2.4.2.1 Request primitive

The value of Command is seven hundred and sixty-nine (769); the value of Octet count field is zero (0) and Value field is null.

8.2.4.2.2 Response primitive

The Value field of response is shown in Table 124. The Response code values are shown in Table 125.

Table 124 – Read join status value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Wireless mode	Wireless mode as specified in Table F.27, the upper four bits shall be set to zero.
1	Bit Field	Join status	This 2-octet enumeration indicates join status as specified in Table F.28
3	Unsigned8	Neighbors	Number of available neighbors
4	Unsigned8	Advertising packets	Number of Advertising packets received
5	Unsigned8	Max_join_retries	Number of join attempts; too many join attempts results in the device considering the join failed
6	Time	Join_rsp_time	The amount of time since the last join request was sent
10	Time	Search time	The amount of time listening for first advertisement

Table 125 – Read join status command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.2.3 Procedure at responding device

This command allows a host system or handheld device to monitor a field device as it transitions through the joining process according to A.2. If the device is not able to join the network, then the response to this command provides diagnostics assistance to an instrument technician or service person.

8.2.4.3 Write active_search_shed_time FAL PDU

8.2.4.3.1 Request primitive

The value of Command is seven hundred and seventy (770); the value of Octet count field is four (4) and value field is as shown in Table 126.

Table 126 – Write Active_search_shed_time request value field

Octet offset	Data type	Parameter name	Description
0	Time	Active_search_shed_time	Time duration for which the active search is enabled during join procedure

8.2.4.3.2 Response primitive

The Value field of response is shown in Table 127. The Response code values are shown in Table 128.

Table 127 – Write Active_search_shed_time response value field

Octet offset	Data type	Parameter name	Description
0	Time	Active_search_shed_time	Time duration for which the active search is enabled during join procedure
4	Time	Advertise_time	Interval at which Advertise DLPDUs are transmitted
8	Unsigned8	Neighbors	Number of neighbors advertising in addition to this device

Table 128 – Write Active_search_shed_time command specific response codes

Value	Class	Description
0	Success	No error
3	Error	Passed parameter larger than permitted
4	Error	Passed parameter smaller than permitted
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Set to the nearest possible value
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict
65	Error	Request declined (for example operator overridden)

8.2.4.3.3 Procedure at responding device

This command requests active and fast search – see A.2.3.2. Generally, this is received by a network device from a local maintenance tool (for example a handheld) and is propagated to the network manager. The network manager is expected to configure or turn on fast advertising in the device and one or more of its neighbors.

8.2.4.4 Write join mode configuration FAL PDU

8.2.4.4.1 Request primitive

The value of Command is seven hundred and seventy-one (771); the value of Octet count field is five (5) and value field is as shown in Table 129.

Table 129 – Write Join mode configuration value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Join mode	Join mode as specified in Table F.36
1	Time	Active_search_shed_time	Time duration for which the active search is enabled during join procedure

8.2.4.4.2 Response primitive

The Value field of response is shown in Table 129. The Response code values are shown in Table 130.

Table 130 – Write Join mode configuration command specific response codes

Value	Class	Description
0	Success	No error
3	Error	Passed parameter larger than permitted
4	Error	Passed parameter smaller than permitted
5	Error	Too few data octets received
6	Error	Device-specific command Error
8	Warning	Set to the nearest possible value
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict
65	Error	Request declined

8.2.4.4.3 Procedure at responding device

This command allows a host system or handheld device to force a field device into active join mode. The device shall stay in the active search mode for at least the Active_search_shed_time – see A.2.3.2. After this time has expired, the device may go into Deep Sleep or Ultra Low Power mode.

8.2.4.5 Read Join mode configuration FAL PDU

8.2.4.5.1 Request primitive

The value of Command is seven hundred and seventy-two (772); the value of Octet count field is zero (0) and Value field is null.

8.2.4.5.2 Response primitive

The Value field of response is shown in Table 131. The Response code values are shown in Table 132.

Table 131 – Read Join mode configuration response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Join mode	Join mode as specified in Table F.36
1	Time	Active_search_shed_time	Time duration for which the active search is enabled during join procedure

Table 132 – Read Join mode configuration command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.5.3 Procedure at responding device

This command reads the Join mode and Active_search_shed_time.

8.2.4.6 Write network_ID FAL PDU**8.2.4.6.1 Request primitive**

The value of Command is seven hundred and seventy-three (773); the value of Octet count field is two (2) and value field is as shown in Table 133.

Table 133 – Write network_ID value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	network_ID	Network identifier as defined in 3.3.65

8.2.4.6.2 Response primitive

The Value field of response is shown in Table 133. The Response code values are shown in Table 134.

Table 134 – Write network_ID command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Network_ID change pending; new value shall be used for next join
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict
65	Error	Invalid network_ID

8.2.4.6.3 Procedure at responding device

This command configures the device to recognize the proper network_ID. If this command was not issued by the network manager or via the maintenance port, then the device shall respond with "Access Restricted".

If the device is not connected to the network, then the network_ID is changed immediately. The device should enter power save mode and wait for the join mode configuration command before beginning to actively search for the network.

Else, the new network_ID shall be used for next join process.

8.2.4.7 Read network_ID FAL PDU

8.2.4.7.1 Request primitive

The value of Command is seven hundred and seventy-four (774); the value of Octet count field is zero (0) and Value field is null.

8.2.4.7.2 Response primitive

The Value field of response is shown in Table 135. The Response code values are shown in Table 136.

Table 135 – Read network_ID value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Network_ID	Network identifier as defined in 3.3.65

Table 136 – Read network_ID command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.7.3 Procedure at responding device

This command reads the current setting of the network_ID from the device.

8.2.4.8 Write network_tag FAL PDU

8.2.4.8.1 Request primitive

The value of Command is seven hundred and seventy-five (775); the value of Octet count field is thirty-two (32) and Value field is as shown in Table 137.

Table 137 – Write Network_tag value field

Octet offset	Data type	Parameter name	Description
0 – 31	Latin-1	Network_tag	Long Tag to be stored in the device

8.2.4.8.2 Response primitive

The Value field of response is shown in Table 137. The Response code values are shown in Table 138.

Table 138 – Write Network_tag command specific Response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.4.8.3 Procedure at responding device

This command writes the tag for the network. The Network_tag is a proxy for the network_ID, so that the network can be identified in a text form at the application level.

8.2.4.9 Read network_tag FAL PDU

8.2.4.9.1 Request primitive

The value of Command is seven hundred and seventy-six (776); the value of Octet count field is zero (0) and Value field is null.

8.2.4.9.2 Response primitive

The Value field of response is shown in Table 139. The Response code values are shown in Table 140.

Table 139 – Read Network_tag response value field

Octet offset	Data type	Parameter name	Description
0 – 31	Latin-1	Long_tag	Long Tag stored in the device

Table 140 – Read Network_tag command-specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy

8.2.4.10 Read wireless device capabilities FAL PDU

8.2.4.10.1 Request primitive

The value of Command is seven hundred and seventy-seven (777); the value of Octet count field is zero (0) and Value field is null.

8.2.4.10.2 Response primitive

The Value field of response is shown in Table 141. The Response code values are shown in Table 142.

Table 141 – Read wireless device capabilities value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Power source	Power Source as specified in Table F.20
1	Float32	Peak packet rate	Peak packets per second
5	Time	Load duration	Duration at peak packet load before power drained
9	Time	Recovery time	Time to recover from power drain
13	Integer8	Good RSL	RSL (Receive Signal Level in dB) threshold for 'good' connection
14	Time	Min Keep_alive_time	Minimum required Keep_alive_time
18	Unsigned16	Neighbor count	Maximum number of neighbors
20	Unsigned16	Packet buffer	Maximum number of packet buffers

Table 142 – Read wireless device capabilities command specific response codes

Value	Class	Description
0	Success	No error

8.2.4.10.3 Procedure at responding device

This PDU is used by the network manager to determine operational characteristics of a network device. The network manager is expected to use Peak packet rate and Recovery time to accommodate rechargeable or power scavenging devices.

Battery operated devices shall be able to sustain the peak packet rate for the specified lifetime of the device. The Load duration shall be at least one hour. If this parameter is not applicable, then its value shall be set to twenty-four hours.

While the device is recovering from the power drain, it shall be able to route one packet per second. If this parameter is not applicable, then its value shall be set to zero.

8.2.4.11 Read battery life FAL PDU

8.2.4.11.1 Request primitive

The value of Command is seven hundred and seventy-eight (778); the value of Octet count field is zero (0) and Value field is null.

8.2.4.11.2 Response primitive

The Value field of response is shown in Table 143. The Response code values are shown in Table 144.

Table 143 – Read battery life response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Battery life	Battery Life remaining in days

Table 144 – Read battery life command-specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy

8.2.4.11.3 Procedure at responding device

This PDU is used to determine the current state of the battery in a network device. If the device does not have a battery or other energy storage component, then the device may return 0xFFFF as the value of Battery life.

NOTE Careful design analysis of long-term battery performance is recommended to properly implement this command.

8.2.4.12 Report device health FAL PDU

8.2.4.12.1 Request primitive

There is no request primitive for this Command.

8.2.4.12.2 Response primitive

The value of Command is seven hundred and seventy-nine (779); the Value field of response is shown in Table 145. The Response code values are shown in Table 146.

Table 145 – Report device health response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Packets generated	Number of packets generated by this device
2	Unsigned16	Packets terminated	Number of packets terminated by this device
4	Unsigned8	DLL MIC failures	Number of DLL MAC MIC failures detected
5	Unsigned8	NL MIC failures	Number of NL (session) MIC failures detected
6	Enumeration	Power status	Power Status as specified by in Table F.33
7	Unsigned8	CRC errors	Number of CRC errors detected
8	Unsigned8	Packets missed	Number of packets detected as missing

Table 146 – Report device health command specific response codes

Value	Class	Description
0	Success	No error

8.2.4.12.3 Procedure at responding device

This PDU is used by the field device to publish communication related counters as Information report. These counters shall be reset only just prior to the device joining the network. After that the counters shall be incremented and roll-over to zero upon overflow. Packets missed value is incremented for each zero in Nonce counter history that underflows.

8.2.4.13 Read neighbor health list FAL PDU

8.2.4.13.1 Request primitive

The value of Command is seven hundred and eighty (780); the value of Octet count field is two (2) and Value field is shown in Table 147.

Table 147 – Read neighbor health list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Neighbor table index
1	Unsigned8	Entry count	Number of neighbor entries to be read

8.2.4.13.2 Response primitive

The Value field of response is shown in Table 148. The Response code values are shown in Table 149.

Table 148 – Read neighbor health list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Neighbor table index
1	Unsigned8	Entry count	Number of neighbor entries in the response
2	Unsigned8	Neighbor count	Total number of neighbors
3	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67
5	Bit Field	Neighbor flags	Neighbor Flags as specified in Table F.34
6	Integer8	Average_RSL	Average RSL (receive signal level in dBm) since last report
7	Unsigned16	Packets_transmitted	Number of packets transmitted to this neighbor
9	Unsigned16	Missed_ACK_packets	Failed transmission to this neighbor – number of packets expecting an ACK and none received
11	Unsigned16	Packets_received	Number of packets received from this neighbor
13..			

^a The fields from octet number 3 to 12 are repeated as many times as the value of Neighbor count.

Table 149 – Read neighbor health list command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.13.3 Procedure at responding device

This command reads the entries in the Neighbor table from the network device. Neighbors with links to device are expected at the beginning of the list and neighbors without links at the end. The neighbor table is a list with entries being deleted as a neighbor stops communication or added as a new neighbor is detected. Table index and Entry count change following addition or deletion of a neighbor. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code “Set to the nearest possible value”.

The device shall send this response to a read request. In addition, the device shall publish the Neighbor health list using “Process Data” level priority as Information report. The publishing period is set by configuring “Health report” time using Command 795 (see 8.2.4.27).

Packet counters shall be reset only just prior to the device joining the network. After that the counters shall be incremented and roll-over to zero upon overflow.

This PDU is also used by the field device to publish neighbor health as Information report.

8.2.4.14 Read device nickname FAL PDU

8.2.4.14.1 Request primitive

The value of Command is seven hundred and eight-one (781); the value of Octet count field is zero (0) and Value field is null.

8.2.4.14.2 Response primitive

The Value field of response is shown in Table 150. The Response code values are shown in Table 151.

Table 150 – Read device nickname response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Device nickname	Nickname of the device as defined in 3.3.67

Table 151 – Read device nickname command-specific response codes

Value	Class	Description
0	Success	No error

8.2.4.14.3 Procedure at responding device

This command reads a network device's Nickname (short address).

8.2.4.15 Read session list FAL PDU

8.2.4.15.1 Request primitive

The value of Command is seven hundred and eighty-two (782); the value of Octet count field is two (2) and Value field is shown in Table 152.

Table 152 – Read session list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Session table index
1	Unsigned8	Entry count	Number of session entries to be read

8.2.4.15.2 Response primitive

The Value field of response is shown in Table 153. The Response code values are shown in Table 154.

Table 153 – Read session list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Session table index
1	Unsigned8	Entry count	Number of session entries in the response
2	Unsigned8	Session count	Total number of active sessions
3	Enumeration	Session_type	Session_type as specified in Table F.24
4	Unsigned16	Peer_nickname	Nickname of the correspondent device
6	Unique ID	Peer_unique_ID	Unique ID of correspondent device as defined in 3.3.91
11	Unsigned32	Peer_nonce_counter	Largest nonce counter value received from the correspondent device
15	Unsigned32	My_nonce_counter	Nonce counter for packets sourced by the device
19..			

^a The fields from octet number 3 to 18 are repeated as many times as the value of Session count.

Table 154 – Read session list command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.15.3 Procedure at responding device

This command reads the entries in the Session table from the network device. The Session table is a list with entries being deleted or added. Session index and Entry count change following addition or deletion of a session. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code “Set to the nearest possible value”.

8.2.4.16 Read superframe list FAL PDU**8.2.4.16.1 Request primitive**

The value of Command is seven hundred and eighty-three (783); the value of Octet count field is two (2) and Value field is shown in Table 155.

Table 155 – Read superframe list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Superframe table index
1	Unsigned8	Entry count	Number of superframe entries to be read

8.2.4.16.2 Response primitive

The Value field of response is shown in Table 156. The Response code values are shown in Table 157.

Table 156 – Read superframe list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Superframe table index
1	Unsigned8	Entry count	Number of superframe entries in the response
2	Unsigned8	Superframe count	Total number of active superframes
3	Unsigned8	Superframe_ID	Identification number assigned to the superframe
4	Unsigned16	Num_Slots	Number of slots in this superframe i.e. size of superframe
6	Bit Field	Superframe mode	Superframe mode flags as specified in Table F.23
7..			
^a The fields from octet number 3 to 6 are repeated as many times as the value of Superframe count.			

Table 157 – Read superframe list command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.16.3 Procedure at responding device

This command reads the entries in the superframe table from the network device. The superframe table is a list with entries being deleted or added. Superframe index and Entry count change following addition or deletion of a superframe. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code “Set to the nearest possible value”.

8.2.4.17 Read link list FAL PDU

8.2.4.17.1 Request primitive

The value of Command is seven hundred and eighty-four (784); the value of Octet count field is three (3) and Value field is shown in Table 158.

Table 158 – Read link list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Table index	Link table index
2	Unsigned8	Entry count	Number of Link entries to be read

8.2.4.17.2 Response primitive

The Value field of response is shown in Table 159. The Response code values are shown in Table 160.

Table 159 – Read link list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned16	Table index	Link table index
2	Unsigned8	Entry count	Number of Link entries in the response
3	Unsigned16	Link count	Total number of active Links
5	Unsigned8	Link_ID	Identification number assigned to this Link
6	Unsigned16	Slot	Slot number in superframe assigned to this link
8	Unsigned8	Channel_Offset	Initial frequency hopping channel offset
9	Unsigned16	Neighbor_nickname	Nickname of neighbor for this link, or 0xFFFF if broadcast link
11	Bit Field	Link options	Link options as specified in Table F.22
12	Enumeration	Link_type	Link_type as specified in Table F.21
13..			

^a The fields from octet number 5 to 12 are repeated as many times as the value of Link count.

Table 160 – Read link list command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.17.3 Procedure at responding device

This command reads the entries in the Link table from the network device. The Link table is a list with entries being deleted or added. Link index and Entry count change following addition or deletion of a Link. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code “Set to the nearest possible value”.

8.2.4.18 Read graph list FAL PDU**8.2.4.18.1 Request primitive**

The value of Command is seven hundred and eighty-five (785); the value of Octet count field is one (1) and Value field is shown in Table 161.

Table 161 – Read graph list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Graph table index

8.2.4.18.2 Response primitive

The Value field of response is shown in Table 162. The Response code values are shown in Table 163.

Table 162 – Read graph list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Graph table index
1	Unsigned8	Graph count	Total number of graphs
2	Unsigned16	Graph_ID	Identification number assigned to this Graph
4	Unsigned8	Neighbor count	Number of neighbors
5	Unsigned16	Neighbor_nickname	Nickname of neighbor for this graph
7..			

^a The fields from octet number 5 to 6 are repeated as many times as the value of Neighbor count.

Table 163 – Read graph list command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.18.3 Procedure at responding device

This command reads one item in the Graph table from the network device. The index is for reference only and the referenced graph may change, if a graph before it on the list is deleted. Therefore, if the parameter in the request is invalid, the device shall respond with the nearest valid value of the parameter and send the response code “Set to the nearest possible value”.

8.2.4.19 Read neighbor attribute flag FAL PDU

8.2.4.19.1 Request primitive

The value of Command is seven hundred and eighty-six (786); the value of Octet count field is two (2) and Value field is shown in Table 164.

Table 164 – Read neighbor attribute flag request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67

8.2.4.19.2 Response primitive

The Value field of response is shown in Table 165. The Response code values are shown in Table 166.

Table 165 – Read neighbor attribute flag response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67
2	Bit Field	Neighbor flags	Neighbor attributes as specified in Table F.34

Table 166 – Read neighbor attribute flag command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
65	Error	Unknown Neighbor_nickname

8.2.4.19.3 Procedure at responding device

This command reads the attributes of a neighbor from the network device.

8.2.4.20 Read neighbor signal level FAL PDU

8.2.4.20.1 Request primitive

The value of Command is seven hundred and eighty-seven (787); the value of Octet count field is two (2) and Value field is shown in Table 167.

Table 167 – Read neighbor signal level request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Neighbor table index
1	Unsigned8	Entry count	Number of neighbor entries to be read

8.2.4.20.2 Response primitive

The Value field of response is shown in Table 168. The Response code values are shown in Table 169.

Table 168 – Read neighbor signal level response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Neighbor table index
1	Unsigned8	Entry count	Number of neighbor entries in the response
2	Unsigned8	Neighbor count	Total number of neighbors
3	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67
5	Integer8	Average_RSL	Average RSL (receive signal level in dBm) of neighbor
6..			

^a The fields from octet number 3 to 5 are repeated as many times as the value of Neighbor count.

Table 169 – Read neighbor signal level command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.20.3 Procedure at responding device

This command reads the signal level in the Neighbor table from the network device. The neighbor table is a list with entries being deleted as a neighbor stops communication or added as a new neighbor is detected. Table index and Entry count change following addition or deletion of a neighbor. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code "Set to the nearest possible value".

The device shall send this response to a read request. In addition, the device shall publish the Neighbor signal level using "Process Data" level priority as Information report. The publishing period is set by configuring "Health report" time using Command 795 (see 8.2.4.27).

8.2.4.21 Report path down alarm FAL PDU**8.2.4.21.1 Request primitive**

There is no request primitive for this service. The value of Command is seven hundred and eighty-eight (788).

8.2.4.21.2 Response primitive

The Value field of response is shown in Table 170. The Response code values are shown in Table 171.

Table 170 – Report path down alarm response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67

Table 171 – Report path down alarm command specific response codes

Value	Class	Description
0	Success	No error
16	Error	Access restricted

8.2.4.21.3 Procedure at responding device

If the device cannot communicate with one of its neighbor for a time exceeding Path_fail_time, it shall publish the path down alarm response using "Alarm" level priority as Information report. The Path_fail_time is configured by Command 795 (see 8.2.4.27).

8.2.4.22 Report source route failed alarm FAL PDU

8.2.4.22.1 Request primitive

There is no request primitive for this service. The value of Command is seven hundred and eighty-nine (789).

8.2.4.22.2 Response primitive

The Value field of response is shown in Table 172. The Response code values are shown in Table 173.

Table 172 – Report source route failed alarm response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Unreachable neighbor	Nickname of unreachable neighbor in the source route
2	Unsigned32	NL MIC	Network layer MIC from the NPDU that failed routing

Table 173 – Report source route failed alarm command specific response codes

Value	Class	Description
0	Success	No error
16	Error	Access restricted

8.2.4.22.3 Procedure at responding device

If the device detects a source route failure, it shall publish the source route failed alarm response using "Alarm" level priority as Information report.

8.2.4.23 Report graph route failed alarm FAL PDU

8.2.4.23.1 Request primitive

There is no request primitive for this service. The value of Command is seven hundred and eighty-ninety (790).

8.2.4.23.2 Response primitive

The Value field of response is shown in Table 174. The Response code values are shown in Table 175.

Table 174 – Report graph route failed alarm response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Graph_ID	Graph_ID of the failed route

Table 175 – Report graph route failed alarm command specific response codes

Value	Class	Description
0	Success	No error
16	Error	Access restricted

8.2.4.23.3 Procedure at responding device

If the device detects a graph route failure, it shall publish the graph route failed alarm response using "Alarm" level priority as Information report.

8.2.4.24 Report transport layer failed alarm FAL PDU

8.2.4.24.1 Request primitive

There is no request primitive for this service. The value of Command is seven hundred and ninety-one (791).

8.2.4.24.2 Response primitive

The Value field of response is shown in Table 176. The Response code values are shown in Table 177.

Table 176 – Report transport layer failed alarm response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Unreachable peer	Nickname of unreachable peer in the end-to-ends transport layer

Table 177 – Report transport layer failed alarm command specific response codes

Value	Class	Description
0	Success	No error
16	Error	Access restricted

8.2.4.24.3 Procedure at responding device

If the device detects a session failure, it shall publish the transport layer connection failed alarm response using "Alarm" level priority as Information report.

8.2.4.25 Write UTC time mapping FAL PDU

8.2.4.25.1 Request primitive

The value of Command is seven hundred and ninety-three (793); the value of Octet count field is seven (7) and value field is as shown in Table 178.

Table 178 – Write UTC time mapping value field

Octet offset	Data type	Parameter name	Description
0	Date	Date	Date
3	Time	Time of day	Time of day

8.2.4.25.2 Response primitive

The Value field of response is shown in Table 178. The Response code values are shown in Table 179.

Table 179 – Write UTC time mapping command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted

8.2.4.25.3 Procedure at responding device

This command sets the date and time received in the request as the instance in time when the ASN zero started. The date and time in the request can be UTC time or local time or any other time that the requester uses. If this command was not issued by the network manager, then the device shall respond with "Access Restricted".

8.2.4.26 Read UTC time mapping FAL PDU

8.2.4.26.1 Request primitive

The value of Command is seven hundred and ninety-four (794); the value of Octet count field is zero (0) and Value field is null.

8.2.4.26.2 Response primitive

The Value field of response is shown in Table 180. The Response code values are shown in Table 181.

Table 180 – Read UTC time mapping response value field

Octet offset	Data type	Parameter name	Description
0	Date	Date	Date
3	Time	Time of day	Time of day

Table 181 – Read UTC time mapping command specific response codes

Value	Class	Description
0	Success	No error

8.2.4.26.3 Procedure at responding device

This command reads the time when the ASN zero started, as set in the network device.

8.2.4.27 Write timer interval FAL PDU**8.2.4.27.1 Request primitive**

The value of Command is seven hundred and ninety-five (795); the value of Octet count field is five (5) and value field is as shown in Table 182.

Table 182 – Write timer interval value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Timer_type	Timer type as specified in Table F.19
1	Unsigned32	Interval	Timer interval in ms

8.2.4.27.2 Response primitive

The Value field of response is shown in Table 182. The Response code values are shown in Table 183.

Table 183 – Write timer interval command specific response codes

Value	Class	Description
0	Success	No error
3	Error	Passed parameter larger than permitted by the device
4	Error	Passed parameter smaller than permitted by the device
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Set to the nearest possible value
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict
65	Error	Invalid Timer_type
66	Error	Invalid timer interval

8.2.4.27.3 Procedure at responding device

This command sets the time interval for the timer specified by Timer_type in a network device.

8.2.4.28 Read timer interval FAL PDU

8.2.4.28.1 Request primitive

The value of Command is seven hundred and ninety-six (796); the value of Octet count field is one (1) and Value field is shown in Table 184.

Table 184 – Read timer interval request value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Timer_type	Timer type as specified in Table F.19

8.2.4.28.2 Response primitive

The Value field of response is shown in Table 185. The Response code values are shown in Table 186.

Table 185 – Read timer interval response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Timer_type	Timer type as specified in Table F.19
1	Unsigned32	Interval	Timer interval in ms

Table 186 – Read timer interval command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started
65	Error	Invalid Timer_type

8.2.4.28.3 Procedure at responding device

This command reads the time interval for the timer specified by Timer_type from a network device.

8.2.4.29 Write radio power output FAL PDU

8.2.4.29.1 Request primitive

The value of Command is seven hundred and ninety-seven (797); the value of Octet count field is one (1) and Value field is as shown in Table 187.

Table 187 – Write radio power output value field

Octet offset	Data type	Parameter name	Description
0	Integer8	Output power	Output power in dBm

8.2.4.29.2 Response primitive

The Value field of response is shown in Table 187. The Response code values are shown in Table 188.

Table 188 – Write radio power output command specific response codes

Value	Class	Description
0	Success	No error
3	Error	Passed parameter larger than permitted
4	Error	Passed parameter smaller than permitted
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Set to the nearest possible value
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.4.29.3 Procedure at responding device

This command sets the radio power output for a device or gateway. If the device cannot accept the exact value sent from the application, it shall set it to the nearest value it can and shall respond with "Set to the nearest possible value". Every device shall support at least following -10 dBm, 0 dBm and +10 dBm values. If the responding device is connected to the network, and if this command was not issued by the network manager, then the device shall respond with "Access Restricted". Else, if the responding device is not connected to the network, and if this command was not received via the maintenance port, then the device shall respond with "Access Restricted".

8.2.4.30 Read radio power output FAL PDU

8.2.4.30.1 Request primitive

The value of Command is seven hundred and ninety-eight (798); the value of Octet count field is zero (0) and Value field is null.

8.2.4.30.2 Response primitive

The Value field of response is shown in Table 189. The Response code values are shown in Table 190.

Table 189 – Read radio power output response value field

Octet offset	Data type	Parameter name	Description
0	Integer8	Output power	Output power in dBm

Table 190 – Read radio power output command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.30.3 Procedure at responding device

This command reads the radio power output from a device or gateway.

8.2.4.31 Request timetable FAL PDU**8.2.4.31.1 Request primitive**

The value of Command is seven hundred and ninety-nine (799); the value of Octet count field is ten (10) and value field is as shown in Table 191.

Table 191 – Request TimeTable value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Timetable_ID	Timetable Identifier supplied by and specific to requesting device; the value shall be in the range of 0x00 – 0x7F
1	Bit Field	Timetable_type	Timetable request flags as specified in Table F.15
2	Enumeration	Application_domain	Application domain as specified in Table F.16
3	Unsigned16	Peer_nickname	Nickname of the peer with which the Timetable is requested
5	Time	Service_duration	If Intermittent service, then latency else, (cyclical communications) period
9	Unsigned8	Route_ID	Route identifier for this Timetable

8.2.4.31.2 Response primitive

The Value field of response is shown in Table 191. The Response code values are shown in Table 192.

Table 192 – Request TimeTable command specific response codes

Value	Class	Description
0	Success	No error
4	Error	Passed parameter smaller than permitted
5	Error	Too few data octets received
6	Error	Device-specific command Error
8	Warning	Set to the nearest possible value
16	Error	Access restricted
14	Warning	Communication bandwidth supplied significantly less than requested (i.e. reduced more than setting to nearest value)
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict
65	Error	Timetable request denied
66	Error	Unknown Timetable_type
67	Error	Unknown Application_domain
68	Error	Unknown Peer_nickname
69	Error	Invalid Timetable_ID

8.2.4.31.3 Procedure at responding device

This command shall be used by a wireless device to add a Timetable to another wireless device with specified bandwidth and latency characteristics. Error free response to this command indicates that the network manager has accepted the request and it will attempt to process the request. The Timetable_ID is unique only within the requesting device.

On a heavily utilized network the network manager may decline requests for communication bandwidth or provide significantly less bandwidth than requested. When this occurs the device shall set the "Capacity Denied" status in Standardized status 3 (see Table F.11) and the "More status available" bit in the "Application process status" octet.

If the device receives a "Write Timetable" (see Command 973) on the same Timetable_ID while awaiting a response to this command, the device shall reject the new "Write Timetable" command.

8.2.4.32 Read timetable list FAL PDU**8.2.4.32.1 Request primitive**

The value of Command is eight hundred (800); the value of Octet count field is two (2) and Value field is shown in Table 193.

Table 193 – Read TimeTable list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Timetable table index
1	Unsigned8	Entry count	Number of Timetable entries to be read

8.2.4.32.2 Response primitive

The Value field of response is shown in Table 194. The Response code values are shown in Table 195.

Table 194 – Read TimeTable list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Timetable table index
1	Unsigned8	Entry count	Number of Timetable entries in the response
2	Unsigned8	Timetable count	Total number of active Timetables
3	Unsigned8	Timetable_ID	Timetable identifier of the entry
4	Bit Field	Timetable_type	Timetable request flags as specified in Table F.15
5	Enumeration	Application_domain	Application domain as specified in Table F.16
6	Unsigned16	Peer_nickname	Nickname of the peer with which the service is established
8	Time	Service_duration	If Intermittent service, then latency else, (cyclical communications) period
12	Unsigned8	Route_ID	Route identifier for this Timetable
13..			

^a The fields from octet number 3 to 12 are repeated as many times as the value of Timetable count.

Table 195 – Read TimeTable list command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid Selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
8	Warning	Set to the nearest possible value

8.2.4.32.3 Procedure at responding device

This command reads the entries in the Timetable table from the network device. The Timetable table is a list with entries being deleted or added. Timetable index and Entry count change following addition or deletion of a Timetable. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code "Set to the nearest possible value". If the Timetable index is larger than the maximum of Timetable currently in the device then the device shall return the last entry and set the response code to "Set to nearest possible value". The 'Entry count' shall be set to a valid value and the "Set to nearest possible value" response code returned.

8.2.4.33 Delete timetable FAL PDU

8.2.4.33.1 Request primitive

The value of Command is eight hundred and one (801); the value of Octet count field is two (2) and value field is as shown in Table 196.

Table 196 – Delete TimeTable request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Timetable_ID	Timetable identifier specific to the device
1	Enumeration	Reason	Reason for Timetable deletion as specified in Table F.25

8.2.4.33.2 Response primitive

The Value field of response is shown in Table 197. The Response code values are shown in Table 198.

Table 197 – Delete TimeTable response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Timetable_ID	Timetable identifier specific to the device
1	Enumeration	Reason	Reason for Timetable deletion as specified in Table F.25
2	Unsigned8	Timetable count	Total number of active Timetables remaining

Table 198 – Delete TimeTable command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Entry not found
66	Error	Invalid reason Code
67	Error	Reason code rejected, Timetable not deleted
68	Error	Delete not allowed - requestor not Timetable owner

8.2.4.33.3 Procedure at responding device

This command notifies a device of Timetable deletion. The delete operation may be caused by peer's request or because of network manager's decision. Only the creator of the Timetable can delete a Timetable. If the owner of the Timetable does not generate this command, the respondent shall reject the command and return the "Delete not allowed" response code.

8.2.4.34 Read route list FAL PDU**8.2.4.34.1 Request primitive**

The value of Command is eight hundred and two (802); the value of Octet count field is two (2) and Value field is shown in Table 199.

Table 199 – Read route list request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Table index	Route table index
1	Unsigned8	Entry count	Number of route entries to be read

8.2.4.34.2 Response primitive

The Value field of response is shown in Table 200. The Response code values are shown in Table 201.

Table 200 – Read route list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Table index	Route table index
1	Unsigned8	Entry count	Number of route entries in the response
2	Unsigned8	Route count	Total number of active routes
3	Unsigned8	Routes remaining	Number of unused Route entries in the device
4	Unsigned8	Route_ID	Route identifier for this entry
5	Unsigned16	Destination nickname	Nickname of the route destination device
7	Unsigned16	Graph_ID	Graph_ID for this route
9	Unsigned8	Source route flag	Source route attached to this route – 1: Yes, 0: No
10..			

^a The fields from octet number 3 to 9 are repeated as many times as the value of Route count.

Table 201 – Read route list command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value

8.2.4.34.3 Procedure at responding device

This command reads the entries in the Route table from the network device. The Route table is a list with entries being deleted or added. Route index and Entry count change following addition or deletion of a Route. Therefore, if the parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code “Set to the nearest possible value”.

8.2.4.35 Read source route FAL PDU

8.2.4.35.1 Request primitive

The value of Command is eight hundred and three (803); the value of Octet count field is one (1) and Value field is shown in Table 202.

Table 202 – Read source route request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for which source route has to be read

8.2.4.35.2 Response primitive

The Value field of response is shown in Table 203. The Response code values are shown in Table 204.

Table 203 – Read source route response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for which source route has to be read
1	Unsigned8	Hop count	Number of hops in this route
2	Unsigned16	Hop	Nickname for hop entry 0
4..			

^a The fields from octet number 2 to 3 are repeated as many times as the value of Hop count

Table 204 – Read source route command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
65	Error	Entry not found

8.2.4.35.3 Procedure at responding device

This command reads the contents of a particular Source route. Broadcast addresses are not legal address values in Source routes and shall not be included in any response for this command.

8.2.4.36 Read CCA mode FAL PDU**8.2.4.36.1 Request primitive**

The value of Command is eight hundred and four (804); the value of Octet count field is zero (0) and Value field is null.

8.2.4.36.2 Response primitive

The Value field of response is shown in Table 205. The Response code values are shown in Table 206.

Table 205 – Read CCA mode response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	CCA Mode	Clear Channel Assessment mode – 1: enabled, 0: disabled

Table 206 – Read CCA mode command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.36.3 Procedure at responding device

This command reads Clear Channel Assessment (CCA) mode from a network device.

8.2.4.37 Write CCA mode FAL PDU

8.2.4.37.1 Request primitive

The value of Command is eight hundred and five (805); the value of Octet count field is one (1) and Value field is as shown in Table 207.

Table 207 – Write CCA mode value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	CCA Mode	Clear Channel Assessment mode – 1: enabled, 0: disabled

8.2.4.37.2 Response primitive

The Value field of response is shown in Table 207. The Response code values are shown in Table 208.

Table 208 – Write CCA mode command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection for example, invalid CCA mode
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started.
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.4.37.3 Procedure at responding device

This command writes the Clear Channel Assessment (CCA) mode in a network device. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.4.38 Read handheld superframe FAL PDU

8.2.4.38.1 Request primitive

The value of Command is eight hundred and six (806); the value of Octet count field is zero (0) and Value field is null.

8.2.4.38.2 Response primitive

The Value field of response is shown in Table 209. The Response code values are shown in Table 210.

Table 209 – Read handheld superframe response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Num_Slots	Number of slots in this superframe i.e. size of superframe
3	Bit Field	Superframe mode	Superframe mode flags as specified in Table F.23

Table 210 – Read handheld superframe command specific response codes

Value	Class	Description
0	Success	No error
9	Error	No handheld superframe

8.2.4.38.3 Procedure at responding device

This command is used to determine if the handheld superframe in a network device is enabled.

8.2.4.39 Write handheld superframe FAL PDU**8.2.4.39.1 Request primitive**

The value of Command is eight hundred and seven (807); the value of Octet count field is two (2) and Value field is as shown in Table 211.

Table 211 – Write handheld superframe value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Bit Field	Superframe mode	Superframe mode flags as specified in Table F.23

8.2.4.39.2 Response primitive

The Value field of response is shown in Table 211. The Response code values are shown in Table 212.

Table 212 – Write handheld superframe command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection (for example invalid superframe mode)
5	Error	Too few data octets received
16	Error	Access restricted

8.2.4.39.3 Procedure at responding device

This command allows a handheld device to request that a network device enable the handheld superframe, so that maintenance can be done. The handheld superframe is used between a network device and a handheld device specifically for maintenance purposes.

8.2.4.40 Read packet hop-to-live FAL PDU

8.2.4.40.1 Request primitive

The value of Command is eight hundred and eight (808); the value of Octet count field is zero (0) and Value field is null.

8.2.4.40.2 Response primitive

The Value field of response is shown in Table 213. The Response code values are shown in Table 214.

Table 213 – Read packet hop-to-live response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Hop-to-live	Currently configured value of Hop-to-live

Table 214 – Read packet hop-to-live command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.40.3 Procedure at responding device

This command reads the current value of Hop-to-live configured for a network device.

8.2.4.41 Write packet hop-to-live FAL PDU

8.2.4.41.1 Request primitive

The value of Command is eight hundred and nine (809); the value of Octet count field is one (1) and Value field is as shown in Table 215.

Table 215 – Write packet hop-to-live value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Hop-to-live	Hop-to-live value

8.2.4.41.2 Response primitive

The Value field of response is shown in Table 215. The Response code values are shown in Table 216.

Table 216 – Write packet hop-to-live command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Set to the nearest value
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.4.41.3 Procedure at responding device

This command writes the Hop-to-live value in a network device. If the Hop-to-live value received in the request is less than eight (8), then this value shall be changed to eight (8) and the response code “Set to the nearest value” shall be returned. If this command is received from any source other than the network manager, then the device shall reject this request with the response code of “Access Restricted”.

8.2.4.42 Read Join_priority FAL PDU**8.2.4.42.1 Request primitive**

The value of Command is eight hundred and ten (810); the value of Octet count field is zero (0) and Value field is null.

8.2.4.42.2 Response primitive

The Value field of response is shown in Table 217. The Response code values are shown in Table 218.

Table 217 – Read Join_priority response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Join_priority	Join_priority at which the device shall advertise; lower number indicates a better choice for joining

Table 218 – Read Join_priority command specific response codes

Value	Class	Description
0	Success	No error

8.2.4.42.3 Procedure at responding device

This command reads the current value of Join_priority from a network device.

8.2.4.43 Write Join_priority FAL PDU

8.2.4.43.1 Request primitive

The value of Command is eight hundred and eleven (811); the value of Octet count field is one (1) and Value field is as shown in Table 219.

Table 219 – Write Join_priority value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Join_priority	Join_priority at which the device shall advertise; lower number indicates a better choice for joining

8.2.4.43.2 Response primitive

The Value field of response is shown in Table 219. The Response code values are shown in Table 220.

Table 220 – Write Join_priority command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest value
16	Error	Access restricted

8.2.4.43.3 Procedure at responding device

This command writes the Join_priority value in a network device. If the Join_priority value received in the request is more than fifteen (15), then this value shall be changed to fifteen (15) and the response code “Set to the nearest value” shall be returned. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.4.44 Read receive Priority_threshold FAL PDU

8.2.4.44.1 Request primitive

The value of Command is eight hundred and twelve (812); the value of Octet count field is zero (0) and Value field is null.

8.2.4.44.2 Response primitive

The Value field of response is shown in Table 221. The Response code values are shown in Table 222.

Table 221 – Read receive Priority_threshold response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Priority_threshold	The lowest priority DLPDU to be accepted from another device

Table 222 – Read receive Priority_threshold command specific response codes

Value	Class	Description
0	Success	No error

8.2.4.44.3 Procedure at responding device

This command reads the current value of Receive Priority_threshold from a network device.

8.2.4.45 Write receive Priority_threshold FAL PDU**8.2.4.45.1 Request primitive**

The value of Command is eight hundred and thirteen (813); the value of Octet count field is one (1) and Value field is as shown in Table 223.

Table 223 – Write receive Priority_threshold value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Priority_threshold	The lowest priority DLRDU to be accepted from another device

8.2.4.45.2 Response primitive

The Value field of response is shown in Table 224. The Response code values are shown in Table 224.

Table 224 – Write receive Priority_threshold command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Set to the nearest value
16	Error	Access restricted

8.2.4.45.3 Procedure at responding device

This command writes the Receive Priority_threshold value in a network device. The permitted values for this parameter are zero (0) to three (3). If any other value is received in the request, then the device shall set the parameter value to '3' and return the response "Set to the nearest value". If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.4.46 Read device list FAL PDU**8.2.4.46.1 Request primitive**

The value of Command is eight hundred and fourteen (814); the value of Octet count field is four (4) and Value field is shown in Table 225.

Table 225 – Read device list request value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Device code	Device list code as specified in Table F.30
1	Unsigned8	Entry count	Number of device entries to be read
2	Unsigned16	Table index	Device table starting index

8.2.4.46.2 Response primitive

The Value field of response is shown in Table 226. The Response code values are shown in Table 227.

Table 226 – Read device list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Enumeration	Device code	Device list code as specified in Table F.30
1	Unsigned8	Entry count	Number of device entries in the response
2	Unsigned16	Table index	Device table starting index
4	Unsigned16	Device count	Total number of device entries in the table
6	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
11..			

^a The fields from octet number 6 to 10 are repeated as many times as the value of Device count.

Table 227 – Read device list command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection i.e. Device list code not supported
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value
64	Error	Command not implemented

8.2.4.46.3 Procedure at responding device

A network device may maintain several Device tables with a list of devices. The currently supported Device tables are – ‘Active list’, ‘White list’ and ‘Black list’. This command reads the entries in the Device table for the type of list specified by Device code in the request. The Device table is a list of devices with entries being deleted or added. Device index and Entry count change following addition or deletion of a device. Therefore, if the Entry count or Table index parameter(s) in the request are invalid, the device shall respond with the nearest valid value of the parameter(s) and send the response code “Set to the nearest possible value”.

The gateway and network manager shall maintain Device table for ‘Active list’ and support this command. The network manager should maintain Device table for ‘White list’ and ‘Black list’. All network devices should maintain Device table for ‘Black list’. If a network device does not maintain Device table for ‘Black list’ and the “device code” in request is for ‘Black list’, then it shall respond to this command with "Command not implemented".

8.2.4.47 Add device list entry FAL PDU

8.2.4.47.1 Request primitive

The value of Command is eight hundred and fifteen (815); the value of Octet count field is six (6) and Value field is shown in Table 228.

Table 228 – Add device list entry request value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Device code	Device list code as specified in Table F.30
1	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.4.47.2 Response primitive

The Value field of response is shown in Table 229. The Response code values are shown in Table 230.

Table 229 – Add device list entry response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Device code	Device list code as specified in Table F.30
1	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
6	Unsigned16	Device count	Total number of entries remaining in the Device table

Table 230 – Add device list entry command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection i.e. Device list code not supported
5	Error	Too few data octets received
8	Warning	Device already in list
16	Error	Access restricted
65	Error	No more entries available
66	Error	Device list conflict

8.2.4.47.3 Procedure at responding device

This command adds a device in the Device table for the type of list specified by Device code. The Device table for 'Active list' cannot be modified by this command. If the request tries to add a device that is in the 'Active list' to 'Black list', then the device shall reject the request and respond with "Device list conflict". This command can be used to move a device between the 'White list' and 'Black list'.

8.2.4.48 Delete device list entry FAL PDU

8.2.4.48.1 Request primitive

The value of Command is eight hundred and sixteen (816); the value of Octet count field is six (6) and Value field is shown in Table 231.

Table 231 – Delete device list entry request value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Device code	Device list code as specified in Table F.30
1	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.4.48.2 Response primitive

The Value field of response is shown in Table 232. The Response code values are shown in Table 233.

Table 232 – Delete device list entry response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Device code	Device list code as specified in Table F.30
1	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
6	Unsigned16	Device count	Total number of entries remaining in the Device table

Table 233 – Delete device list entry command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection i.e. Device list code not supported
5	Error	Too few data octets received
16	Error	Access restricted
66	Error	Device list conflict

8.2.4.48.3 Procedure at responding device

This command deletes a device in the Device table for the type of list specified by Device code in a gateway or network manager. If the device is being deleted from 'Active list', then the network manager shall reroute the traffic and disconnect the device.

8.2.4.49 Read channel blacklist FAL PDU

8.2.4.49.1 Request primitive

The value of Command is eight hundred and seventeen (817); the value of Octet count field is zero (0) and Value field is null.

8.2.4.49.2 Response primitive

The Value field of response is shown in Table 234. The Response code values are shown in Table 235.

Table 234 – Read channel blacklist response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Channel count	Number of bits in current channel map array; this depends on the physical layer (for example for 2,4 GHz O-QPSK DSSS the array is 16-bits long i.e., 2-octets)
1 - n	Bit Field	Current channel map	Current channel map array; this is an array of bits starting with the least significant bit (bit 0 in octet 0) and adding octets as necessary until all bits are accounted for. Each bit corresponds to a channel. If the bit is '1', the channel will be used
n+1–2n	Bit Field	Pending channel map	Pending channel map array; this is an array of bits starting with the least significant bit (bit 0 in octet 0) and adding octets as necessary until all bits are accounted for. Each bit corresponds to a channel. If the bit is '1', the channel will be used after the next restart of the gateway or network manager

The number of octets in channel map depends upon the total number of channels available in the physical layer.

Table 235 – Read channel blacklist command specific response codes

Value	Class	Description
0	Success	No error

8.2.4.49.3 Procedure at responding device

This command reads the current channel blacklist and the blacklist that shall be used after the next restart.

8.2.4.50 Write channel blacklist FAL PDU

8.2.4.50.1 Request primitive

The value of Command is eight hundred and eighteen (818); the value of Octet count field is variable and Value field is shown in Table 236.

Table 236 – Write channel blacklist value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Channel count	Number of bits in new channel map array
1 - n	Bit Field	Pending channel map	Pending channel map array; this is an array of bits starting with the least significant bit (bit 0 in octet 0) and adding octets as necessary until all bits are accounted for. Each bit corresponds to a channel. If the bit is '1', the channel will be used after the next restart of the gateway or network manager

The number of octets in channel map depends upon the total number of channels available in the physical layer.

8.2.4.50.2 Response primitive

The Value field of response is shown in Table 236. The Response code values are shown in Table 237.

Table 237 – Write channel blacklist command specific response codes

Value	Class	Description
0	Success	No error
3	Error	Parameter value to be written is larger than permitted – number of bits exceeds maximum value
5	Error	Too few data octets received
6	Error	Device-specific command Error
16	Error	Access restricted
65	Error	Invalid frequency channel bits (for example channel 16 for 2,4 GHz according to IEEE 802.15.4 physical layer)

8.2.4.50.3 Procedure at responding device

This command writes the pending channel map, which is the blacklist to the gateway. This channel map shall come into effect only after the gateway and the network manager are restarted.

8.2.4.51 Read back-off exponent FAL PDU

8.2.4.51.1 Request primitive

The value of Command is eight hundred and nineteen (819); the value of Octet count field is zero (0) and Value field is null.

8.2.4.51.2 Response primitive

The Value field of response is shown in Table 238. The Response code values are shown in Table 239.

Table 238 – Read back-off exponent response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Max_backoff_exp	The maximum value that can be assumed for the back-off exponent used in shared slots

Table 239 – Read back-off exponent command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started.

8.2.4.51.3 Procedure at responding device

This command reads the current value of Max_backoff_exp from a network device.

8.2.4.52 Write back-off exponent FAL PDU

8.2.4.52.1 Request primitive

The value of Command is eight hundred and twenty (820); the value of Octet count field is one (1) and Value field is as shown in Table 240.

Table 240 – Write back-off exponent value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Max_backoff_exp	The maximum value that can be assumed for the back-off exponent used in shared slots

8.2.4.52.2 Response primitive

The Value field of response is shown in Table 240. The Response code values are shown in Table 241.

Table 241 – Write back-off exponent command specific response codes

Value	Class	Description
0	Success	No error
3	Error	Passed parameter larger than permitted
4	Error	Passed parameter smaller than permitted
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
8	Warning	Set to the nearest value
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.4.52.3 Procedure at responding device

This command writes the Max_backoff_exp value in a network device. If the Max_backoff_exp value received in the request is not one of the values in the set {4, 5, 6, 7}, then this value shall be changed to four (4) and the response code “Set to the nearest value” shall be returned. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.4.53 Write network access mode FAL PDU**8.2.4.53.1 Request primitive**

The value of Command is eight hundred and twenty-one (821); the value of Octet count field is one (1) and Value field is as shown in Table 242.

Table 242 – Write network access mode value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Network access mode	Network access mode code as specified in Table F.31

8.2.4.53.2 Response primitive

The Value field of response is shown in Table 242. The Response code values are shown in Table 243.

Table 243 – Write network access mode command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
6	Error	Device-specific command Error
7	Error	In write protect mode
16	Error	Access restricted
32	Error	Busy – a delayed response could not be started
33	Error	DR initiated
34	Error	DR running
35	Error	DR dead
36	Error	DR conflict

8.2.4.53.3 Procedure at responding device

This command writes the network access mode value in a network device. This can be used to restrict the access to the network. All devices shall support values '0' and '4' for this parameter.

NOTE The network access mode does not substitute the check for the device's credentials such as Join_key, tag or Device ID. The network access mode provides additional checks.

8.2.4.54 Read network access mode FAL PDU

8.2.4.54.1 Request primitive

The value of Command is eight hundred and twenty-two (822); the value of Octet count field is zero (0) and Value field is null.

8.2.4.54.2 Response primitive

The Value field of response is shown in Table 244. The Response code values are shown in Table 245.

Table 244 – Read network access mode response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Network access mode	Network access mode code as specified in Table F.31

Table 245 – Read network access mode command specific response codes

Value	Class	Description
0	Success	No error
6	Error	Device-specific command Error
32	Error	Busy – a delayed response could not be started

8.2.4.54.3 Procedure at responding device

This command reads the current value of network access mode from a network device.

8.2.4.55 Request session FAL PDU

8.2.4.55.1 Request primitive

The value of Command is eight hundred and twenty-three (823); the value of Octet count field is two (2) and value field is as shown in Table 246.

Table 246 – Request Session request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Peer_nickname	Nickname of the peer with which the session is requested

8.2.4.55.2 Response primitive

The Value field of response is shown in Table 247. The Response code values are shown in Table 248.

Table 247 – Request Session response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Peer_nickname	Nickname of the peer with which the session is requested
2	Unsigned32	Peer_nonce_counter	Largest nonce counter value received from the correspondent device
6	Security key	Join_key	This is 128-bit network key that is used for security

Table 248 – Request Session command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Unknown Peer
66	Error	Field device has insufficient capacity to support another session

8.2.4.55.3 Procedure at responding device

This command requests a unicast session from a handheld to a device. Only authorized devices shall be granted a session.

8.2.5 Gateway and network manager specific APDU structures

8.2.5.1 Read network device identity using unique ID FAL PDU

8.2.5.1.1 Request primitive

The value of Command is eight hundred and thirty-two (832); the value of Octet count field is five (5) and Value field is shown in Table 249.

Table 249 – Read network device identity request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.1.2 Response primitive

The Value field of response is shown in Table 250. The Response code values are shown in Table 251.

Table 250 – Read network device identity response value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned16	Device nickname	Nickname of the device as defined in 3.3.67
7 – 38	Latin-1	Long_tag	Long Tag stored in the device

Table 251 – Read network device identity command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received

8.2.5.1.3 Procedure at responding device

This command reads the identity information of the device with matching Unique ID from the gateway.

8.2.5.2 Read device's neighbor health FAL PDU

8.2.5.2.1 Request primitive

The value of Command is eight hundred and thirty-three (833); the value of Octet count field is seven (7) and Value field is shown in Table 252.

Table 252 – Read device's neighbor health request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned8	Table index	Neighbor table index
6	Unsigned8	Entry count	Number of neighbor entries to be read

8.2.5.2.2 Response primitive

The Value field of response is shown in Table 253. The Response code values are shown in Table 254.

Table 253 – Read device's neighbor health response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unique ID	Unique ID	Unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned8	Table index	Neighbor table index
6	Unsigned8	Entry count	Number of neighbor entries in the response
7	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67
9	Integer8	Average_RSL	Average RSL (receive signal level in dBm) received at this device from this neighbor
10	Unsigned32	Packets_transmitted	Number of packets transmitted to this neighbor
14	Unsigned32	Missed_ACK_packets	Failed transmission to this neighbor – number of packets expecting an ACK and none received
18	Unsigned32	Packets_received	Number of packets received from this neighbor
22..			

^a The fields from octet number 7 to 21 are repeated as many times as the value of neighbor count.

Table 254 – Read device's neighbor health command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
8	Warning	Set to the nearest possible value
65	Error	Invalid neighbor table index

8.2.5.2.3 Procedure at responding device

This command reads from the gateway, information regarding the quality of the connection to each active neighbor or potential neighbor for the device specified in the request. The gateway can find the number of neighbors for the device by sending "Read device list FAL PDU" request (see 8.2.4.46). The requesting application can use this service to build the graph and quality of the network.

8.2.5.3 Read network topology information FAL PDU**8.2.5.3.1 Request primitive**

The value of Command is eight hundred and thirty-four (834); the value of Octet count field is seven (7) and Value field is shown in Table 255.

Table 255 – Read network topology information request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned16	Table index	Graph table index

8.2.5.3.2 Response primitive

The Value field of response is shown in Table 256. The Response code values are shown in Table 257.

Table 256 – Read network topology information response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned16	Table index	Graph table index
7	Unsigned16	Graph count	Total number of graphs
9	Unsigned16	Graph_ID	Identification number assigned to this Graph
11	Unsigned16	Neighbor count	Number of neighbors
13	Unsigned16	Neighbor	Nickname of neighbor for this graph
15..			

^a The fields from octet number 13 to 14 are repeated as many times as the value of Neighbor count.

Table 257 – Read network topology information command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received
65	Error	Invalid graph table index

8.2.5.3.3 Procedure at responding device

This command reads one item in the Graph table for the specified network device from the gateway. The gateway can find the number of graphs for the device by using 'Read graph list' FAL PDU (see 8.2.4.18). The requesting application can use this service to build the graph of the network.

8.2.5.4 Read publish data message list FAL PDU**8.2.5.4.1 Request primitive**

The value of Command is eight hundred and thirty-five (835); the value of Octet count field is five (5) and Value field is shown in Table 258.

Table 258 – Read publish data message list request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.4.2 Response primitive

The Value field of response is shown in Table 259. The Response code values are shown in Table 260.

Table 259 – Read publish data message list response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned8	Publish data count	Number of different publish data commands received from this device
6	Unsigned16	Publish command	Command number being published
8	Unsigned32	Packet count	Number of publish data packets received
12..			
^a The fields from octet number 6 to 11 are repeated as many times as the value of Publish data count.			

Table 260 – Read publish data message list command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received

8.2.5.4.3 Procedure at responding device

This command reads from the gateway, the publish data message list that the specified device participates in, including number of publish data mode commands that the specified device participates in. The requesting application can use this service to build the list of all publish data mode traffic in the system.

8.2.5.5 Flush cached responses for a device FAL PDU**8.2.5.5.1 Request primitive**

The value of Command is eight hundred and thirty-six (836); the value of Octet count field is five (5) and Value field is shown in Table 261.

Table 261 – Flush cached value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.5.2 Response primitive

The Value field of response is shown in Table 261. The Response code values are shown in Table 262.

Table 262 – Flush cached responses command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted

8.2.5.5.3 Procedure at responding device

This command flushes the cached responses in the gateway, for the specified device.

8.2.5.6 Write update notification bit mask for a device FAL PDU

8.2.5.6.1 Request primitive

The value of Command is eight hundred and thirty-seven (837); the value of Octet count field is seven (7) and Value field is shown in Table 263.

Table 263 – Write update notification bit mask for a device value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Bit Field	Notification mask	2-octet change notification mask as specified in Table F.35

8.2.5.6.2 Response primitive

The Value field of response is shown in Table 263. The Response code values are shown in Table 264.

Table 264 – Write update notification bit mask for a device command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Unknown Unique ID
66	Error	Unknown Notification mask

8.2.5.6.3 Procedure at responding device

This command registers a gateway client for notification updates.

8.2.5.7 Read update notification bit mask for a device FAL PDU

The value of Command is eight hundred and thirty-eight (838); the value of Octet count field is five (5) and Value field is shown in Table 265.

Table 265 – Read update notification bit mask for a device request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.7.1 Response primitive

The Value field of response is shown in Table 266. The Response code values are shown in Table 267.

Table 266 – Read update notification bit mask for a device response value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Bit Field	Notification mask	2-octet change notification mask as specified in Table F.35

Table 267 – Read update notification bit mask for a device command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
65	Error	Unknown Unique ID

8.2.5.7.2 Procedure at responding device

This command reads the list of update notification mask for the specified device from the gateway.

8.2.5.8 Read change notification FAL PDU**8.2.5.8.1 Request primitive**

The value of Command is eight hundred and thirty-nine (839); the value of Octet count field is five (5) and Value field is shown in Table 268.

Table 268 – Report change notification request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.8.2 Response primitive

The Value field of response is shown in Table 269. The Response code values are shown in Table 270.

Table 269 – Report change notification response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned8	Change count	Total number of change notifications
6	Unsigned16	Change notification	Change notification (cached command number)
8..			

^a The fields from octet number 6 to 7 are repeated as many times as the value of Change count.

Table 270 – Report change notification command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
65	Error	Unknown Unique ID

8.2.5.8.3 Procedure at responding device

This request is sent by the client application to the gateway. The response lists the changes for the specified device. The maximum permitted value of Change count is ten (10).

8.2.5.9 Read network device statistics FAL PDU

8.2.5.9.1 Request primitive

The value of Command is eight hundred and forty (840); the value of Octet count field is five (5) and Value field is shown in Table 271.

Table 271 – Read network device statistics request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.9.2 Response primitive

The Value field of response is shown in Table 272. The Response code values are shown in Table 273.

Table 272 – Read network device statistics response value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Unsigned16	Graph count	Number of active graphs
7	Unsigned16	Frame count	Number of active frames
9	Unsigned16	Link count	Number of active links
11	Unsigned8	Neighbor count	Number of neighbors
12	Time	Latency	Average communication latency from the gateway to this device
16	Unsigned16	Join count	Number of joins
18	Date	Join date	Date of most recent join
21	Time	Join time	Time in 1/32 of a ms unit from midnight of the date when the device most recently joined the network
25	Unsigned32	Packets generated	Number of packets generated by this device
29	Unsigned32	Packets terminated	Number of packets terminated by this device
33	Unsigned32	DLL MIC failures	Number of DLL MAC MIC failures detected
37	Unsigned32	NL MIC failures	Number of NL (session) MIC failures detected
41	Unsigned32	CRC errors	Number of CRC errors detected
45	Unsigned32	Packets missed	Number of packets detected as missing
49	Unsigned32	Nonce counter missed	Number of Nonce Counter Values not received from the device
53	Time	Latency	Standard deviation of latency

Table 273 – Read network device statistics command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received

8.2.5.9.3 Procedure at responding device

This command reads the statistics for the specified device from the gateway. The gateway is expected have this information available and shall respond immediately. Packets missed value is incremented for each zero in Nonce counter history that underflows.

8.2.5.10 Read network device identity using nickname FAL PDU**8.2.5.10.1 Request primitive**

The value of Command is eight hundred and forty-one (841); the value of Octet count field is two (2) and Value field is shown in Table 274.

Table 274 – Read network device identity request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Device_nickname	Nickname of the device as defined in 3.3.67

8.2.5.10.2 Response primitive

The Value field of response is shown in Table 275. The Response code values are shown in Table 276.

Table 275 – Read network device identity response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Device_nickname	Nickname of the device as defined in 3.3.67
2	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
7 – 38	Latin-1	Long_tag	Long Tag stored in the device

Table 276 – Read network device identity command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received

8.2.5.10.3 Procedure at responding device

This command reads the identity information of the device with matching Device nickname from the gateway.

8.2.5.11 Write network device's scheduling flags FAL PDU

8.2.5.11.1 Request primitive

The value of Command is eight hundred and forty-two (842); the value of Octet count field is six (6) and Value field is shown in Table 277.

Table 277 – Write network device's scheduling flags value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Bit Field	Scheduling flags	Device scheduling flags as specified in Table F.37

8.2.5.11.2 Response primitive

The Value field of response is shown in Table 277. The Response code values are shown in Table 278.

Table 278 – Write network device's scheduling flags command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
7	Error	In write protect mode
8	Warning	Unsupported attribute flag detected, Device attribute Flags set to the nearest possible value
9	Error	Invalid attribute flag
65	Error	Unknown Unique ID

8.2.5.11.3 Procedure at responding device

This command allows user to request special consideration for a device when the network manager is creating schedules. This command shall only be supported by gateways and network managers. This information shall not be written to the network device.

8.2.5.12 Read network device's scheduling flags FAL PDU**8.2.5.12.1 Request primitive**

The value of Command is eight hundred and forty-three (843), the value of Octet count field is five (5) and Value field is shown in Table 279.

Table 279 – Read network device's scheduling flags request value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique_ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4

8.2.5.12.2 Response primitive

The Value field of response is shown in Table 280. The Response code values are shown in Table 281.

Table 280 – Read network device's scheduling flags response value field

Octet offset	Data type	Parameter name	Description
0	Unique ID	Unique_ID	The unique identifier assigned to the device as defined in 3.3.91 and described in 6.2.1.2.4
5	Bit Field	Scheduling flags	Device scheduling flags as specified in Table F.37

Table 281 – Read network device's scheduling flags command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection
5	Error	Too few data octets received

8.2.5.12.3 Procedure at responding device

This command reads the network device attributes that a network manager may consider when creating schedules. This command shall only be supported by gateways and network managers.

8.2.5.13 Read network constraints FAL PDU

8.2.5.13.1 Request primitive

The value of Command is eight hundred and forty-four (844); the value of Octet count field is zero (0) and Value field is null.

8.2.5.13.2 Response primitive

The Value field of response is shown in Table 282. The Response code values are shown in Table 283.

Table 282 – Read network constraints value field

Octet offset	Data type	Parameter name	Description
0	Bit Field	Optimization flags	Network optimization flags as specified in Table F.38
1	Unsigned8	Message pair count	Number of request / response message pairs that the network manager is permitted to send in 10 s duration

Table 283 – Read network constraints command specific response codes

Value	Class	Description
0	Success	No error

8.2.5.13.3 Procedure at responding device

This command reads the current setting of the network management strategy and the number of request / response messages per 10 s.

8.2.5.14 Write network constraints FAL PDU

8.2.5.14.1 Request primitive

The value of Command is eight hundred and forty-five (845); the value of Octet count field is two (2) and Value field is shown in Table 284.

Table 284 – Write network constraints value field

Octet offset	Data type	Parameter name	Description
0	Bit Field	Optimization flags	Network optimization flags as specified in Table F.38
1	Unsigned8	Message pair count	Number of request / response message pairs that the network manager is permitted to send in 10 s duration

8.2.5.14.2 Response primitive

The Value field of response is shown in Table 284. The Response code values are shown in Table 285.

Table 285 – Write network constraints command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid selection (illegal or unsupported strategy)
5	Error	Too few data octets received
7	Error	In write protect mode
8	Warning	Set to the nearest possible value
16	Error	Access restricted

8.2.5.14.3 Procedure at responding device

This command writes the current setting of the network management strategy and the number of request / response messages per 10 s. Minimum value for Number of request / response messages per 10 s is one (1).

8.2.6 Network management configuration specific APDU structures**8.2.6.1 Disconnect device FAL PDU****8.2.6.1.1 Request primitive**

The value of Command is nine hundred and sixty (960); the value of Octet count field is one (1) and value field is as shown in Table 286.

Table 286 – Disconnect device value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Disconnect cause	Disconnect cause code as specified in Table F.26

8.2.6.1.2 Response primitive

The Value field of response is shown in Table 286. The Response code values are shown in Table 287.

Table 287 – Disconnect device command specific response codes

Value	Class	Description
0	Success	No error
16	Error	Access restricted

8.2.6.1.3 Procedure at responding device

This service allows the network manager to force a device off the network. The receiving device shall go off the network, issue TL-FLUSH request, clear all its network information and rejoin the network. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.2 Write network key FAL PDU**8.2.6.2.1 Request primitive**

The value of Command is nine hundred and sixty-one (961); the value of Octet count field is twenty-one (21) and value field is as shown in Table 288.

Table 288 – Write network key value field

Octet offset	Data type	Parameter name	Description
0	Security key	Network key	This is 128-bit network key that is used for security
16	Unsigned40	Key change time	Execution ASN time for command; 0 - execute immediately

The transport type for this request shall be Request-broadcast.

8.2.6.2.2 Response primitive

The Value field of response is shown in Table 288. The Response code values are shown in Table 289.

Table 289 – Write network key command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Key change failed
66	Error	Invalid execution time

8.2.6.2.3 Procedure at responding device

This command writes the network key in a network device. The value field in the request can be truncated after the Key value. If truncated, then the network Key shall become effective immediately. Else, the new key shall become effective at the Key change time received in the request. If the Key change time received in the request is less than the current ASN time the "Invalid execution time" response code shall be returned. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.6.3 Write device nickname FAL PDU

8.2.6.3.1 Request primitive

The value of Command is nine hundred and sixty-two (962); the value of Octet count field is two (2) and Value field is as shown in Table 290.

Table 290 – Write device nickname value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Device nickname	Nickname of the device as defined in 3.3.67

8.2.6.3.2 Response primitive

The Value field of response is shown in Table 290. The Response code values are shown in Table 291.

Table 291 – Write device nickname command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Invalid device nickname

8.2.6.3.3 Procedure at responding device

This command writes a network device's nickname. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.6.4 Write session FAL PDU**8.2.6.4.1 Request primitive**

The value of Command is nine hundred and sixty-three (963); the value of Octet count field is thirty-two (32) and Value field is shown in Table 292.

Table 292 – Write session request value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Session_type	Session_type as specified in Table F.24
1	Unsigned16	Peer_nickname	Nickname of the correspondent device
3	Unique ID	Peer_unique_ID	Unique ID of correspondent device as defined in 3.3.91
8	Unsigned32	Peer_nonce_counter	Nonce counter value received from the correspondent device
12	Security key	Session_key	This is 128-bit network key that is used for security.
28	Unsigned8	Reserved	Always zero (0) in request
29	Unsigned40	Session change time	Execution ASN time for command; 0 - execute immediately

8.2.6.4.2 Response primitive

The Value field of response is shown in Table 293. The Response code values are shown in Table 294.

Table 293 – Write session response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Session_type	Session_type as specified in Table F.24
1	Unsigned16	Peer_nickname	Nickname of the correspondent device
3	Unique ID	Peer_unique_ID	Unique ID of correspondent device as defined in 3.3.91
8	Unsigned32	Peer_nonce_counter	Nonce counter value received from the correspondent device
12	Security key	Session_key	This is 128-bit session key that is used for security
28	Unsigned8	Session count	Number of sessions remaining
29	Unsigned40	Session change time	Execution ASN time for command; 0 - execute immediately

Table 294 – Write session command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	No more entries available
66	Error	Invalid execution time
67	Error	Session_type invalid

8.2.6.4.3 Procedure at responding device

This command writes the session parameters required to establish a session between the destination (of this request) device and the peer device specified in the request. Session is identified by Session_type and Peer_nickname. If this command is modifying an existing session then the Peer_nonce_counter is ignored. The response always includes the values actually used by the device. When a new key is written to the session the old key shall be retained to allow for out-of-order packet reception. The old key shall be discarded after 2 * Max_PDU_age duration from the reception of the request. The value fields in the request can be truncated after the Session_key value. If truncated, then the Session_key shall become effective immediately. If the Session change time received in the request is less than the current ASN time the "Invalid execution time" response code shall be returned. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.6.5 Delete session FAL PDU

8.2.6.5.1 Request primitive

The value of Command is nine hundred and sixty-four (964); the value of Octet count field is three (3) and Value field is shown in Table 295.

Table 295 – Delete session request value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Session_type	Session_type as specified in Table F.24
1	Unsigned16	Peer_nickname	Nickname of the correspondent device

8.2.6.5.2 Response primitive

The Value field of response is shown in Table 296. The Response code values are shown in Table 297.

Table 296 – Delete session response value field

Octet offset	Data type	Parameter name	Description
0	Enumeration	Session_type	Session_type as specified in Table F.24
1	Unsigned16	Peer_nickname	Nickname of the correspondent device
3	Unsigned8	Session count	Number of sessions remaining

Table 297 – Delete session command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Session with given peer device does not exist

8.2.6.5.3 Procedure at responding device

This command deletes an end-to-end session in a network device. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.6 Write superframe FAL PDU**8.2.6.6.1 Request primitive**

The value of Command is nine hundred and sixty-five (965); the value of Octet count field is ten (10) and Value field is shown in Table 298.

Table 298 – Write superframe request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Num_Slots	Number of slots in this superframe i.e. size of superframe
3	Bit Field	Superframe mode	Superframe mode flags as specified in Table F.23
4	Unsigned8	Reserved	Always zero (0) in request
5	Unsigned40	Superframe change time	Execution ASN time for command; 0 - execute immediately

8.2.6.6.2 Response primitive

The Value field of response is shown in Table 299. The Response code values are shown in Table 300.

Table 299 – Write superframe response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Num_Slots	Number of slots in this superframe i.e. size of superframe
3	Bit Field	Superframe mode	Superframe mode flags as specified in Table F.23
4	Unsigned8	Superframe count	Number of superframes remaining
5	Unsigned40	Superframe change time	Execution ASN time for command; 0 - execute immediately

Table 300 – Write superframe command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	No more entries available
66	Error	Invalid execution time
67	Error	Invalid number of slots
68	Error	Invalid superframe mode

8.2.6.6.3 Procedure at responding device

This command writes or modifies a superframe in a network device. If the device has a superframe with its ID matching the Superframe_ID in the request, then it shall modify the existing superframe. Else, it shall add a new superframe. The device shall make the superframe effective at the superframe change time received in the request. This allows synchronous network-wide operation. If this request is received while a superframe change is pending then the pending modifications shall be overwritten by the new values received in this request.

The value fields in the request can be truncated after the superframe mode value. If truncated, then the superframe shall become effective immediately. If the Superframe change time received in the request is less than the current ASN time the "Invalid execution time" response code shall be returned. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.6.7 Delete superframe FAL PDU

8.2.6.7.1 Request primitive

The value of Command is nine hundred and sixty-six (966); the value of Octet count field is one (1) and Value field is shown in Table 301.

Table 301 – Delete superframe request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe

8.2.6.7.2 Response primitive

The Value field of response is shown in Table 302. The Response code values are shown in Table 303.

Table 302 – Delete superframe response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned8	Superframe count	Number of superframes remaining

Table 303 – Delete superframe command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Superframe entry not found

8.2.6.7.3 Procedure at responding device

This command deletes a superframe in a network device. If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.8 Write link FAL PDU**8.2.6.8.1 Request primitive**

The value of Command is nine hundred and sixty-seven (967); the value of Octet count field is eight (8) and Value field is shown in Table 304.

Table 304 – Write link request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Slot	Slot number in superframe assigned to this link
3	Unsigned8	Channel_Offset	Initial frequency hopping channel offset
4	Unsigned16	Neighbor	Nickname of neighbor for this link, or 0xFFFF if broadcast link
6	Bit Field	Link options	Link options as specified in Table F.22
7	Enumeration	Link_type	Link_type as specified in Table F.21

8.2.6.8.2 Response primitive

The Value field of response is shown in Table 305. The Response code values are shown in Table 306.

Table 305 – Write link response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Slot	Slot number in superframe assigned to this link
3	Unsigned8	Channel_Offset	Initial frequency hopping channel offset
4	Unsigned16	Neighbor	Nickname of neighbor for this link, or 0xFFFF if broadcast link
6	Bit Field	Link options	Link options as specified in Table F.22
7	Enumeration	Link_type	Link_type as specified in Table F.21
8	Unsigned16	Link count	Total number of Link entries remaining

Table 306 – Write link command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid Selection e.g., nickname = 0x0000
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	No more Link entries available
66	Error	Link already exists
67	Error	Unknown Superframe_ID
68	Error	Invalid Slot
69	Error	Invalid Link options (for example neither transmit nor receive set)
70	Error	Invalid Channel_Offset
71	Error	Invalid Link_type
72	Error	No more neighbors available

8.2.6.8.3 Procedure at responding device

This command adds a link assignment to a network device. The link is uniquely identified by the combination of following key parameters:

Superframe_ID, Slot, Neighbor.

If the request specifies a Link that already exists, then the device shall reject the request and respond with “Link already exists”. The network manager has to delete an existing link and then write a new link to accomplish a modify operation.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.9 Delete link FAL PDU

8.2.6.9.1 Request primitive

The value of Command is nine hundred and sixty-eight (968); the value of Octet count field is five (5) and Value field is shown in Table 307.

Table 307 – Delete link request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Slot	Slot number in superframe assigned to this link
3	Unsigned16	Neighbor	Nickname of neighbor for this link, or 0xFFFF if broadcast link

8.2.6.9.2 Response primitive

The Value field of response is shown in Table 308. The Response code values are shown in Table 309.

Table 308 – Delete link response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Superframe_ID	Identification number assigned to the superframe
1	Unsigned16	Slot	Slot number in superframe assigned to this link
3	Unsigned16	Neighbor	Nickname of neighbor for this link, or 0xFFFF if broadcast link
5	Unsigned16	Link count	Total number of Link entries remaining

Table 309 – Delete link command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Link does not exist

8.2.6.9.3 Procedure at responding device

This command deletes a Link assignment in a network device. The link is uniquely identified by the combination of following key parameters:

Superframe_ID, Slot, Neighbor.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.10 Write graph connection FAL PDU**8.2.6.10.1 Request primitive**

The value of Command is nine hundred and sixty-nine (969); the value of Octet count field is four (4) and Value field is shown in Table 310.

Table 310 – Write graph connection request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Graph_ID	Identification number assigned to this Graph
2	Unsigned16	Neighbor_nickname	Nickname of neighbor for this graph

8.2.6.10.2 Response primitive

The Value field of response is shown in Table 311. The Response code values are shown in Table 312.

Table 311 – Write graph connection response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Graph_ID	Identification number assigned to this graph
2	Unsigned16	Neighbor_nickname	Nickname of neighbor for this graph
4	Unsigned16	Connection count	Total number of connection entries remaining

Table 312 – Write graph connection command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid Selection e.g., nickname = 0x0000
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	No more Edge entries available
66	Error	Unknown neighbor
67	Error	Can't add connection because superframe is used as graph equivalent

8.2.6.10.3 Procedure at responding device

This command adds a graph connection assignment to a network device. A graph and its connections with an ID less than 256 is specified by a superframe and its transmit links. In that case connections shall be added to the graph by writing a transmit link to new neighbor for that superframe-graph.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.6.11 Delete graph connection FAL PDU

8.2.6.11.1 Request primitive

The value of Command is nine hundred and seventy (970); the value of Octet count field is four (4) and Value field is shown in Table 313.

Table 313 – Delete graph connection request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Graph_ID	Identification number assigned to this Graph
2	Unsigned16	Neighbor_nickname	Nickname of neighbor for this graph

8.2.6.11.2 Response primitive

The Value field of response is shown in Table 314. The Response code values are shown in Table 315.

Table 314 – Delete graph connection response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Graph_ID	Identification number assigned to this Graph
2	Unsigned16	Neighbor_nickname	Nickname of neighbor for this graph
4	Unsigned16	Connection count	Total number of connection entries remaining

Table 315 – Delete graph connection command specific response codes

Value	Class	Description
0	Success	No error
2	Error	Invalid Selection e.g., nickname = 0x0000
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Graph connection entry does not exist

8.2.6.11.3 Procedure at responding device

This command deletes a graph connection assignment in a network device.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.12 Write neighbor attribute flag FAL PDU**8.2.6.12.1 Request primitive**

The value of Command is nine hundred and seventy-one (971); the value of Octet count field is three (3) and Value field is shown in Table 316.

Table 316 – Write neighbor attribute flag value field

Octet offset	Data type	Parameter name	Description
0	Unsigned16	Neighbor_nickname	Nickname of neighbor as defined in 3.3.67
2	Bit Field	Neighbor flags	Neighbor attributes as specified in Table F.34

8.2.6.12.2 Response primitive

The Value field of response is shown in Table 316. The Response code values are shown in Table 317.

Table 317 – Write neighbor attribute command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Unknown neighbor
66	Error	Invalid neighbor attribute

8.2.6.12.3 Procedure at responding device

This command sets attributes of a neighbor to a network device.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.13 Write network suspend FAL PDU

8.2.6.13.1 Request primitive

The value of Command is nine hundred and seventy-two (972); the value of Octet count field is ten (10) and Value field is shown in Table 318.

Table 318 – Write network suspend value field

Octet offset	Data type	Parameter name	Description
0	Unsigned40	Suspend time	ASN time at which to suspend network
5	Unsigned40	Resume time	ASN time at which to resume network

8.2.6.13.2 Response primitive

The Value field of response is shown in Table 318. The Response code values are shown in Table 319.

Table 319 – Write network suspend command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Invalid Suspend time
66	Error	Invalid Resume time

8.2.6.13.3 Procedure at responding device

This command is used by the network manager to suspend the operation of one or more devices. The command may be sent to all devices on the network or an individual device based on the addressing used. The receiving device shall go off the network, delete all of the pending transmission requests and clear all its network information. If this command is received while a suspend command is pending then the pending suspend setting shall be overwritten by the values received in this request.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.14 Write timetable FAL PDU

8.2.6.14.1 Request primitive

The value of Command is nine hundred and seventy-three (973); the value of Octet count field is ten (10) and Value field is shown in Table 320.

Table 320 – Write TimeTable request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Timetable_ID	Timetable identifier of the entry; if the network manager is creating a new Timetable then the Timetable_ID shall be in the range of 0x80-0xFF
1	Bit Field	Timetable_type	Timetable request flags as specified in Table F.15
2	Enumeration	Application_domain	Application domain as specified in Table F.16
3	Unsigned16	Peer_nickname	Nickname of the peer with which the Timetable is established
5	Time	Service_duration	If Intermittent service, then latency else, (cyclical communications) period
9	Unsigned8	Route_ID	Route identifier for this Timetable

8.2.6.14.2 Response primitive

The Value field of response is shown in Table 321. The Response code values are shown in Table 322.

Table 321 – Write TimeTable response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Timetable_ID	Timetable identifier of the entry
1	Bit Field	Timetable_type	Timetable request flags as specified in Table F.15
2	Enumeration	Application_domain	Application domain as specified in Table F.16
3	Unsigned16	Peer_nickname	Nickname of the peer with which the Timetable is established
5	Time	Service_duration	If Intermittent service, then latency else, (cyclical communications) period
9	Unsigned8	Route_ID	Route identifier for this Timetable
10	Unsigned8	Timetable count	Total number of Timetable entries remaining

Table 322 – Write TimeTable command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	No more Timetable entries available
66	Error	Invalid Timetable_ID
67	Error	Open transaction pending
68	Error	Invalid application domain
69	Error	Unknown correspondent nickname (no session exists)
70	Error	Invalid route ID
71	Error	Correspondent nickname and route correspondent mismatch

8.2.6.14.3 Procedure at responding device

This command adds new Timetable to a network device. The network device may start using this Timetable after processing this command. The network manager may issue this command to restrict or reduce previously allocated bandwidth (for example during a network disturbance). When this command is used to reduce previously allocated bandwidth the

recipient device shall set the "Capacity Denied" status and the "More status available" bit in the "Application process status" octet, if the bandwidth restriction can result in a process upset/disturbance.

If this command is received while the device is awaiting a response to a "Request Timetable (see Command 799)" on the same Timetable_ID, the device shall respond to this command with an "Open Transaction Pending" error.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of "Access Restricted".

8.2.6.15 Write route FAL PDU

8.2.6.15.1 Request primitive

The value of Command is nine hundred and seventy-four (974); the value of Octet count field is five (5) and Value field is shown in Table 323.

Table 323 – Write route request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for this entry
1	Unsigned16	Peer_nickname	Nickname of the peer with which the route is established
3	Unsigned16	Graph_ID	Graph_ID for this route

8.2.6.15.2 Response primitive

The Value field of response is shown in Table 324. The Response code values are shown in Table 325.

Table 324 – Write route response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for this entry
1	Unsigned16	Peer_nickname	Nickname of the peer with which the route is established
3	Unsigned16	Graph_ID	Graph_ID for this route
5	Unsigned8	Route count	Total number of Route entries remaining

Table 325 – Write route command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	No Route entries available
66	Error	Invalid Peer_nickname
67	Error	Invalid Graph_ID

8.2.6.15.3 Procedure at responding device

This command sets or modifies a route for a particular service in a network device.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.16 Delete route FAL PDU

8.2.6.16.1 Request primitive

The value of Command is nine hundred and seventy-five (975); the value of Octet count field is one (1) and Value field is shown in Table 326.

Table 326 – Delete route request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for this entry

8.2.6.16.2 Response primitive

The Value field of response is shown in Table 327. The Response code values are shown in Table 328.

Table 327 – Delete route response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for this entry
1	Unsigned8	Route count	Total number of Route entries remaining

Table 328 – Delete route command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
65	Error	Invalid Route identifier

8.2.6.16.3 Procedure at responding device

This command deletes a Route entry in a network device.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.17 Write source route FAL PDU

8.2.6.17.1 Request primitive

The value of Command is nine hundred and seventy-six (976); the value of Octet count field is variable and Value field is shown in Table 329.

Table 329 – Write source route request value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for which source route has to be written
1	Unsigned8	Hop count	Number of hops in this route
2	Unsigned16	Hop	Nickname for hop entry 0
4..			

^a The fields from octet number 2 to 3 are repeated as many times as the value of Hop count.

8.2.6.17.2 Response primitive

The Value field of response is shown in Table 330. The Response code values are shown in Table 331.

Table 330 – Write source route response value field

Octet offset ^a	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for which source route has to be written
1	Unsigned8	Hop count	Number of hops in this route
2	Unsigned16	Hop	Nickname for hop entry 0
4..n			
n+1	Unsigned8	Source route count	Number of Source route entries remaining

^a The fields from octet number 2 to 3 are repeated as many times as the value of Hop count.

Table 331 – Write source route command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
8	Warning	Broadcast addresses deleted in response
16	Error	Access restricted
65	Error	No more entries available
66	Error	Invalid Route identifier
67	Error	Invalid Nickname
68	Error	Invalid Hop count

8.2.6.17.3 Procedure at responding device

This command writes a Source route in a network device. Broadcast addresses are not legal address values in this command and shall not be included in Source routes or the requests or responses for this command.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.6.18 Delete source route FAL PDU

8.2.6.18.1 Request primitive

The value of Command is nine hundred and seventy-seven (977); the value of Octet count field is one (1) and Value field is shown in Table 332.

Table 332 – Delete source route request value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for this entry

8.2.6.18.2 Response primitive

The Value field of response is shown in Table 333. The Response code values are shown in Table 334.

Table 333 – Delete source route response value field

Octet offset	Data type	Parameter name	Description
0	Unsigned8	Route_ID	Route identifier for this entry

Table 334 – Delete source route command specific response codes

Value	Class	Description
0	Success	No error
5	Error	Too few data octets received
16	Error	Access restricted
66	Error	Invalid Route identifier

8.2.6.18.3 Procedure at responding device

This command deletes a source route entry in a network device.

If this command is received from any source other than the network manager, then the device shall reject this request with a response code of “Access Restricted”.

8.2.7 Data coding rules

8.2.7.1 General

If the user data is always a simple (primitive) component, it is encoded as shown in Figure 48. The semantics of the user data are known implicitly from the position in the PDU, the length is fixed and implicitly known.

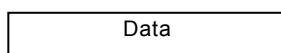


Figure 48 – Coding without identification

8.2.7.2 Coding for integer types

Integer values are signed quantities as shown in Figure 49 and Table 335. The MSB of this data type is transferred first followed by the next octets and the LSB is transferred last.

Notation:	Integer8, Integer16, Integer24, Integer32		
Range of Values:	Data Type	range of values	length
	Integer8	$-128 \leq i \leq 127$	1 octet
	Integer16	$-32768 \leq i \leq 32767$	2 octets
	Integer24	$-2^{23} \leq i \leq 2^{23} - 1$	3 octets
	Integer32	$-2^{31} \leq i \leq 2^{31} - 1$	4 octets
Coding:	In two's complement representation the MSB (Most Significant Bit) is the bit after the sign bit (SN) in the first octet. SN = 0: positive numbers and z SN = 1: negative numbers		

Figure 49 – Coding of Integer type data

Table 335 – Coding of Integer16 type data

Octets	Bits							
	8	7	6	5	4	3	2	1
1	SN	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8
2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

8.2.7.3 Coding for unsigned Types

Unsigned Values are encoded as shown in Figure 50 and Table 336. The MSB of this data type is transferred first followed by the next octets and the LSB is transferred last.

Notation:	Unsigned8, Unsigned16, Unsigned24, Unsigned32, Unsigned40		
Range of Values:	Data Type	range of values	length
	Unsigned8	$0 \leq i \leq 255$	1 octet
	Unsigned16	$0 \leq i \leq 65535$	2 octets
	Unsigned24	$0 \leq i \leq 2^{24} - 1$	3 octets
	Unsigned32	$0 \leq i \leq 2^{32} - 1$	4 octets
	Unsigned40	$0 \leq i \leq 2^{40} - 1$	5 octets

Figure 50 – Coding of Unsigned type data

Table 336 – Coding of Unsigned16 type data

Octets	Bits							
	8	7	6	5	4	3	2	1
1	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8
2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

8.2.7.4 Coding for floating point Type

Floating Point values are encoded as shown in Figure 51 and Figure 52. The sign and MSB of the exponent is transferred first followed by the balance of the exponent and the MSB to LSB of the fraction. If the value of an object of floating point data type is not known, then the value transferred is 0x7F, 0xA0, followed by all zeros (0x00); this value represents a 'Not-a-number'.

Notation:		Floating-Point (4 octet)							
Range of Values:		see IEC 60559, Short Real Number (32 bits)							
Coding:		see IEC 60559, Short Real Number (32 bits)							
bits	8	7	6	5	4	3	2	1	LSB
octets	Exponent (E)								
1	SN	2^7	2^6	2^5	2^4	2^3	2^2	2^1	
2	(E)	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
3	Fraction (F)								
	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}	
4	Fraction (F)								
	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}	2^{-23}	
SN: sign 0 = positive, 1 = negative									

Figure 51 – Coding of single precision floating point type data

Notation:		Floating-Point (8 octet)							
Range of Values:		see IEC 60559, Short Real Number (64 bits)							
Coding:		see IEC 60559, Short Real Number (64 bits)							
bits	8	7	6	5	4	3	2	1	LSB
octets	Exponent (E)								
1	SN	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	
2	Exponent (E)				Fraction (F)				
	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	
3	Fraction (F)								
	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	
4	Fraction (F)								
	2^{-13}	2^{-14}	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	
4	Fraction (F)								
	2^{-21}	2^{-22}	2^{-23}	2^{-24}	2^{-25}	2^{-26}	2^{-27}	2^{-28}	
4	Fraction (F)								
	2^{-29}	2^{-30}	2^{-31}	2^{-32}	2^{-33}	2^{-34}	2^{-35}	2^{-36}	
4	Fraction (F)								
	2^{-37}	2^{-38}	2^{-39}	2^{-40}	2^{-41}	2^{-42}	2^{-43}	2^{-44}	
4	Fraction (F)								
	2^{-45}	2^{-46}	2^{-47}	2^{-48}	2^{-49}	2^{-50}	2^{-51}	2^{-52}	
SN: sign 0 = positive, 1 = negative									

Figure 52 – Coding of double precision floating point type data

8.2.7.5 Coding for date type

The data type Date consists of a calendar date as shown in Table 337 and Figure 53. The first octet representing day of month is transferred first, followed by the month octet and the year octet is transferred last.

Notation: Date
Range of Values: 1 January 1900 to 31 December 2155
Coding: in 3 octets

Table 337 – Coding for Date type

Parameter	Range of values	Meaning of the parameters
d. of m.	1...31	day of month
months	1...12	months
years	0...255	years – 1900

Bits	8	7	6	5	4	3	2	1	
octets	RSV	RSV	RSV	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰	1...31 d. of m.
1									
2	RSV	RSV	RSV	RSV	2 ³	2 ²	2 ¹	2 ⁰	1...12 months
3	RSV	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰	0...255 years

Figure 53 – Coding of Date type data

8.2.7.6 Coding for enumeration

The data type Enumeration is used to associate specific user definitions to numeric codes. For every object of Enumeration data type, there is a definition table that is shown in Annex F. This data type is transferred as either Unsigned8 for 1-octet long data, or Unsigned16 for 2-octet long data.

8.2.7.7 Coding for bit field

The Bit Field type of data is used to transfer objects encoded as single-bit data. Table 338 and Table 339 show the numbering scheme of the bits of the data type Bit Field. Only multiples of eight are legal values of the length (in bits) of the Bit Field. For a 2-octet Bit Field the most significant octet is communicated first in a PDU. For every object of Bit Field data type, there is a definition table that is shown in Annex F.

Table 338 – Coding of 1-octet Bit Field type data

Octet	Bits							
	8	7	6	5	4	3	2	1
1	8	7	6	5	4	3	2	1

Table 339 – Coding of 2-octet Bit Field type data

Octet	Bits							
	8	7	6	5	4	3	2	1
1	16	15	14	13	12	11	10	9
2	8	7	6	5	4	3	2	1

8.2.7.8 Coding for time type

The data type Time is 32-bit unsigned integer and represents time in the increments of 1/32 of a millisecond. If this data type is used to represent time of day, then it indicates number of 1/32 of milliseconds since midnight. The MSB of this data type is transferred first followed by the next octets and the LSB is transferred last.

8.2.7.9 Coding for security key type

The data type security key is a string of sixteen octets. The MSB of this data type is transferred first followed by the next octets and the LSB is transferred last.

8.2.7.10 Coding for Unique ID type

The data type Unique ID is a string of five octets. The MSB of this data type is transferred first followed by the next octets and the LSB is transferred last.

8.2.7.11 Coding for engineering unit type

The data type Engineering unit is one octet long. The coding of this data type is same as that of Unsigned8.

8.2.7.12 Coding for packed ASCII string type

Packed ASCII is a 6-bit character code representing a subset of the ASCII character code set as shown in Table 340. It is produced by compressing four Packed ASCII characters into three octets. This type of string is always a multiple of 4 characters (3 octets) and is padded out to the end of the data item with SP characters. The length of the string is not transferred along with the string. The length is predetermined for each type of object, which is identified by object index. If the number of characters in the value of a string object is less than its predetermined length, then unused characters at the end of the string are set to SP (0x20). For example 4 SP characters at the end of a string would appear as the 3 octets: 0x82, 0x08, and 0x20. The first character of this string is transferred first, followed by the next character and the last character is transferred last.

Table 340 – Packed ASCII character set

Bits 4 and 5	Bits 0 to 3															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
2	SP	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?

NOTE Most significant hexadecimal digit is top to bottom; least significant is left to right. SP indicates a space character.

8.2.7.13 Coding for restricted ISO Latin-1 string type

ISO Latin-1 (ISO/IEC 8859-1) string consists of one character per octet. A restricted version that omits rows 0, 1, 8 and 9 in the Latin-1 table is used, as shown in Table 341.

The length of the string is not transferred along with the string. The length is predetermined for each type of object, which is identified by object index. If the number of characters in the value of a string object is less than its predetermined length, then unused characters at the end of the string are set to zeros (0x00). One zero (0x00) at the end of a string ISO Latin-1 data item may not be sufficient to meet this requirement. A valid restricted ISO Latin-1 character, other than zero, is permitted in the last octet of a restricted ISO Latin-1 data object. The first character of this string is transferred first, followed by the next character and the last character is transferred last.

Table 341 – Subset of ISO Latin-1 characters

Bits 4 to 7	Bits 0 to 3															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2	SP	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	
8																
9																
A	NBSP	ı	ç	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
B	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

NOTE Most significant hexadecimal digit is top to bottom; least significant is left to right. SP indicates a space character. NBSP indicates a non-breaking space character. SHY indicates a soft hyphen. Grayed out cells means that no character is assigned to this code.

8.3 Common procedures

8.3.1 Delayed response

8.3.1.1 General operation

The delayed response mechanism (DRM) enables a slave to indicate to the requesting master that it received the request but is not able to immediately formulate the reply. This mechanism provides an informative and flexible convention for slave devices needing additional time to perform operations such as self diagnostics, calibration, configuration or bandwidth requests. Unlike the response code “Busy”, the master knows that the slave has understood the command and is still communicating. The response APDU specification in 8.2.3, 8.2.4, 8.2.5 and 8.2.6 show which AL services allow the DRM responses. All network devices shall support DRM and comply with the following rules.

- If a request APDU is received that would normally initiate a DR and no buffers are available then the slave shall respond with a Busy.
- The device shall always respond to an Identity Command even if a DR is in progress. The field device may successfully respond to “Read” commands even if a DR is in progress.
- If a DR is in progress that will prevent a slave from responding to a request APDU, then the slave shall respond with a “DR_CONFLICT” code. For example DR_CONFLICT response is returned if a master tries to initiate a transducer trim while one is already in process.
- After DR processing is complete, the slave shall respond to READ commands even if the master has not sent the result of the DR.
- Bridging devices should support one DR buffer for each communication channel supported according to this standard.

Slave devices shall use the response codes listed in Table 342 to synchronize DR operation with master devices.

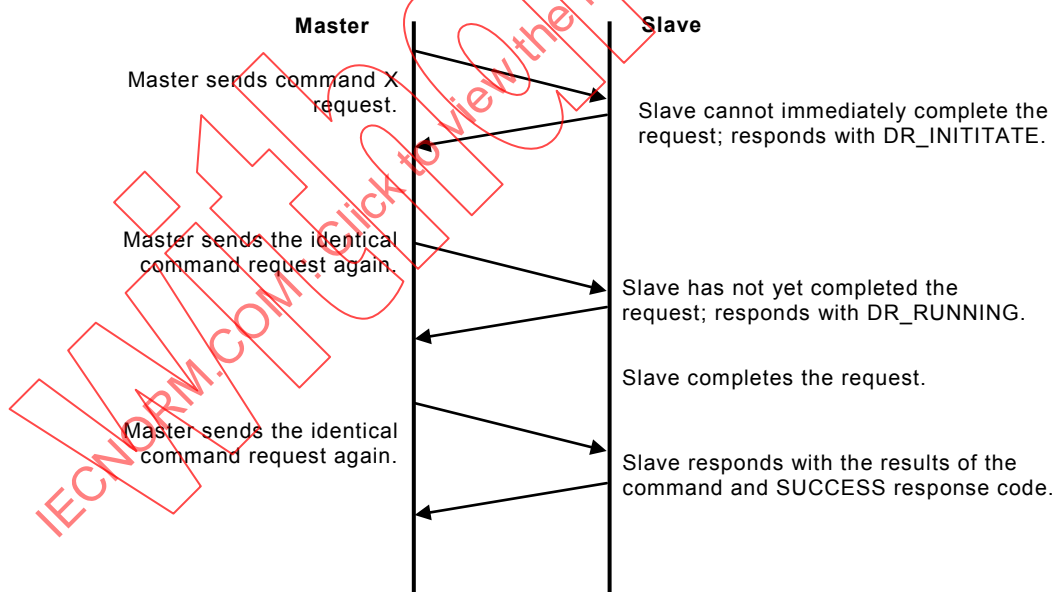
Table 342 – DRM related response codes

Mnemonic	Value	Description
Busy	32	Indicates the device is busy performing an indivisible operation. For commands supporting DRM, Busy indicates that no more DR buffers are available.
DR_INITIATE	33	Indicates the start of a DR. The slave device needs additional time to process the command
DR	34	Indicates that the slave device is still processing the command.
DR_DEAD	35	Used by multiplexers supporting Type 20 protocol, to indicate that no response at all was received from the slave device.
DR_CONFLICT	36	Indicates that the command cannot be processed because it is in conflict with DR currently being processed.

All master class devices shall support DRM. When a master receives a DR_INITIATE response, the exact command (including the data) has to be reissued to retrieve the results. The master can issue this command after a delay.

8.3.1.2 Normal DR operation

A simple DR transaction is shown in Figure 54. The master sends a command that the slave cannot immediately complete. The slave initiates the DR and begins processing the command for later retrieval by the master. Sometime later the master issues the same exact command. The slave is not finished and issues a DR_RUNNING. Later the command completes and the master retrieves the result.

**Figure 54 – Normal DR operation**

8.3.1.3 Use of DR_CONFLICT response code

The example of DR_CONFLICT is shown in Figure 55. If a DR_CONFLICT is received, the master should delay its retry to allow the responding slave to complete the pending DR. Masters should follow this procedure to minimize repetitive polling that, in turn, would delay the field device's completion of the DR.

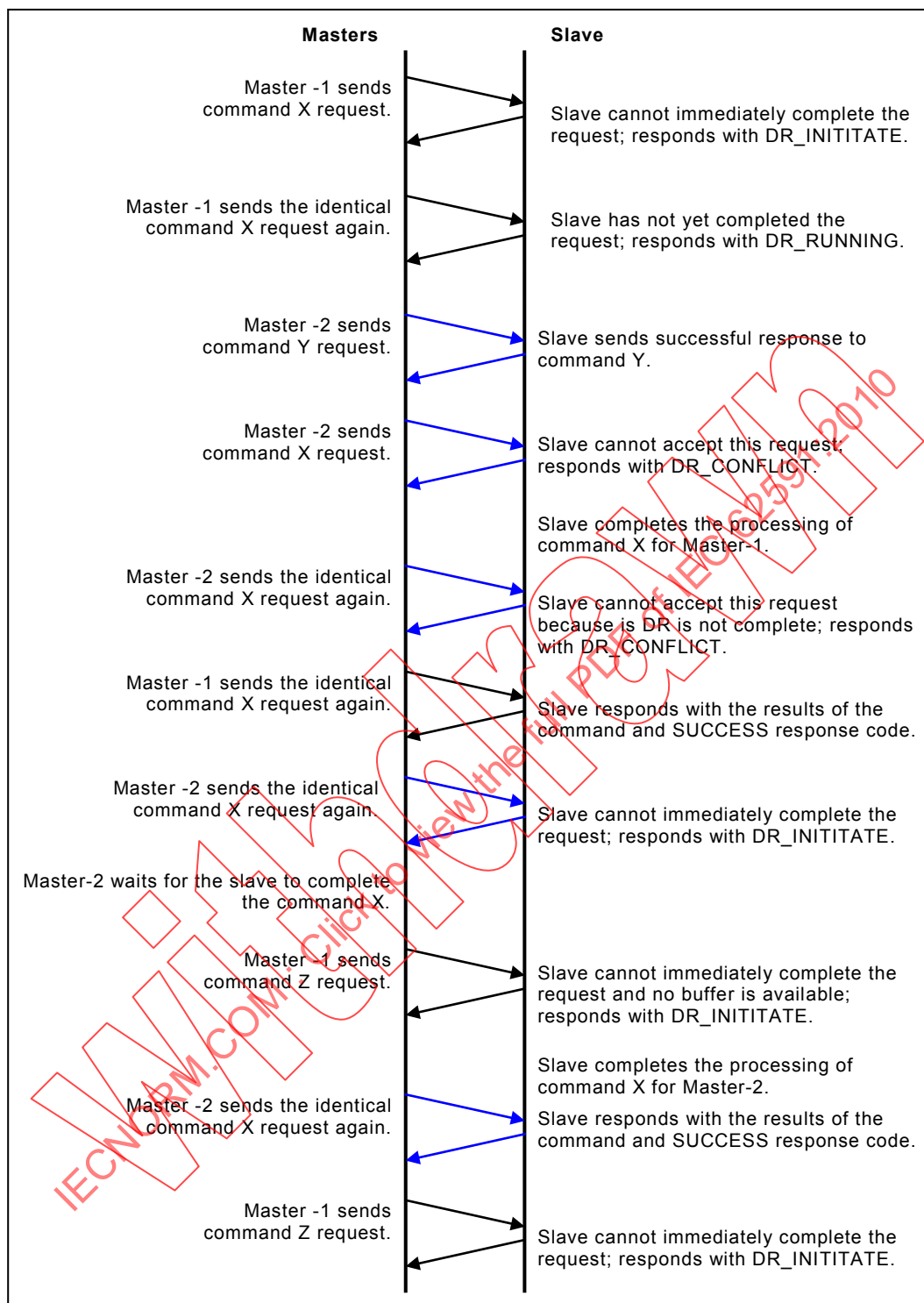


Figure 55 – Command responses during DR processing

8.3.1.4 Multiple DR buffers

A slave device can have two DR buffers, one for each master. Some devices (for example multiplexers and I/O systems) may have many more buffers. Multiple DR buffers allow a DR from each master to be queued. However, the slave shall be careful to protect against inconsistencies. If implemented properly, out-of-order-completion of pending delayed responses is possible as shown in Figure 56.

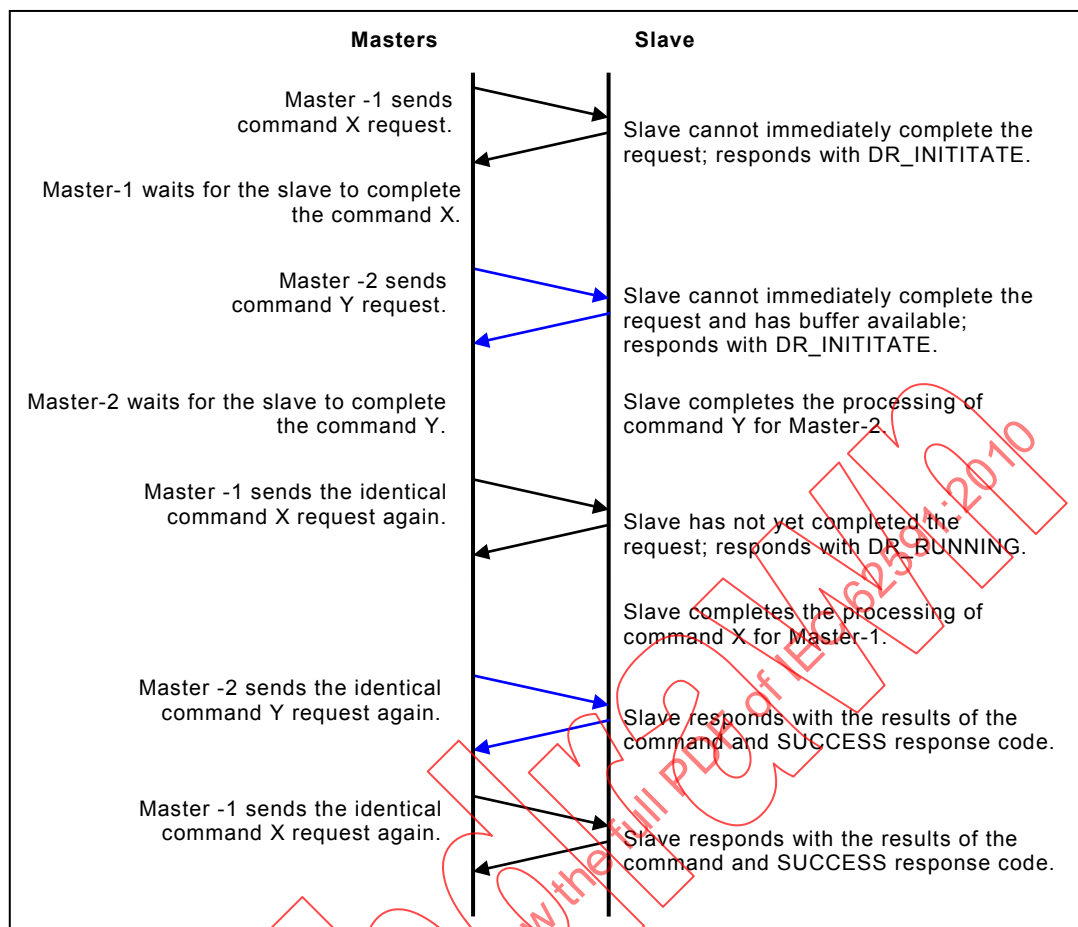


Figure 56 – Slaves with multiple DR buffers

8.3.1.5 Bridge device use of DRM

Bridge devices (for example multiplexers and I/O systems) function as slaves for the master that issues a request and as master for slaves that are addressed. Such devices are identified by setting the Protocol_Bridge_Device bit in the Flags byte of Identity Commands.

Bridge devices may introduce delays in relaying a command to and from the field device or sub-device. As a result, bridge devices are allowed to use the DRM for all command.

An additional response code (DR_DEAD) is used if the slave connected to the bridge device fails to reply at all. A master receiving DR_DEAD shall assume that a serious error has occurred. Since an intelligent bridge should automatically retry the command if the slave does not answer, the master should immediately notify the user that the slave device is not responding.

8.3.2 Response time

To minimize the network latency, the responding device shall meet the timing specified below.

- Prepare the response to a network management configuration specific (see 8.2.6) request, such that the response can be sent in the second slot following the slot in which the PDU was received (i.e., one intervening slot between the request and the response is allowed for command processing).
- Prepare the response to all other requests, such that the response can be sent in the sixth slot following the slot in which the PDU was received.

- If the response code has to be 'DR initiated', then this response shall not be generated until after 75 % of the TL max_reply_time has elapsed.

8.4 FAL service protocol machine (FSPM)

8.4.1 General

FAL Service Protocol Machine is common for AL User services. It has one state called "ACTIVE" as shown in Figure 57.

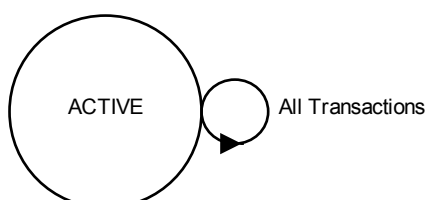


Figure 57 – State transition diagram of FSPM

8.4.2 FSPM state tables

Table 343 and Table 344 specify the FSPM protocol machine. All read and write services have identical state transition and identical actions. Therefore, this table does not list every service request and response separately.

Table 343 – FSPM state Table – client transactions

#	Current state	Event or condition => action	Next state
S1	ACTIVE	Identify.req => Transaction.req { remote_address := AREP_ID, transport_type := TransportType (Identify.req), priority := Priority (Identify.req), req_user_data := FormReqApdu (Identify.req) }	ACTIVE
S2	ACTIVE	Read(any).req => Transaction.req { remote_address := AREP_ID, transport_type := TransportType (Variable index), priority := Priority (Variable index), req_user_data := FormReqApdu (Read.req) }	ACTIVE
S3	ACTIVE	Write(any).req => Transaction.req { remote_address := AREP_ID, transport_type := type (Variable index), priority := priority (Variable index), req_user_data := FormReqApdu (Write.req) }	ACTIVE
S4	ACTIVE	Transaction.cnf && status == "Error" && RetrieveCommand (rsp_user_data) == 0 11 21 => Identify.cnf(-) { Status := "Error" }	ACTIVE

#	Current state	Event or condition => action	Next state
S5	ACTIVE	Transaction.cnf && CommandErr (rsp_user_data) == "True" && RetrieveCommand (rsp_user_data) == 0 11 21 => Identify.cnf(-) { Resp_code := ResponseCode (rsp_user_data), Dev_stat := DeviceStat (rsp_user_data) } }	ACTIVE
S6	ACTIVE	Transaction.cnf && status == "Error" && RetrieveCommand (rsp_user_data) != 0 11 21 => Read.cnf(-) Write.cnf(-) { Status := "Error" } }	ACTIVE
S7	ACTIVE	Transaction.cnf && CommandErr (rsp_user_data) == "True" && RetrieveCommand (rsp_user_data) != 0 11 21 => Identify.cnf(-) Write.cnf(-) { Resp_code := ResponseCode (rsp_user_data), Dev_stat := DeviceStat (rsp_user_data) } }	ACTIVE
S8	ACTIVE	Transaction.cnf && CommandErr (rsp_user_data) == "False" && RetrieveCommand (rsp_user_data) == 0 11 21 => Identify.cnf(+) { Resp_code := ResponseCode (rsp_user_data), Dev_stat := DeviceStat (rsp_user_data), data := Value (rsp_user_data) } }	ACTIVE
S9	ACTIVE	Transaction.cnf && CommandErr (rsp_user_data) == "False" && RetrieveCommand (rsp_user_data) != 0 11 21 => Read (any).cnf(+) Write (any).cnf(+) { Resp_code := ResponseCode (rsp_user_data), Dev_stat := DeviceStat (rsp_user_data), data := Value (rsp_user_data) } }	ACTIVE
S10	ACTIVE	Receive.ind && CommandErr (user_data) == "False" => Information report.ind { AREP_ID := remote_address, Variable index := RetrieveCommand (rsp_user_data), Resp_code := ResponseCode (rsp_user_data), Dev_stat := DeviceStat (rsp_user_data), data := Value (rsp_user_data) } }	
NOTE Read (any) includes all AL Read services; Write (any) includes all AL Write services.			

Table 344 – FSPM state Table – server transactions

#	Current state	Event or condition => action	Next state
R1	ACTIVE	Transaction.ind => al_service.ind { AREP_ID := remote_address, Variable index := RetrieveCommand (req_user_data), data := Value (req_user_data, transport_type) }	ACTIVE
R2	ACTIVE	al_service.rsp => Transaction.rsp { remote_address := AREP_ID, rsp_user_data := FormRspApdu (al_service.rsp) }	ACTIVE
R3	ACTIVE	Information report(any).req => Send.req { remote_address := AREP_ID, transport_type := type(Variable index), priority := priority(Variable index), rsp_user_data := FormApdu (Information report.req) }	ACTIVE
NOTE al_service includes all AL services.			

8.4.3 Functions used by FSPM

Table 345 to Table 353 describe the functions used by FSPM.

Table 345 – Function TransportType ()

Name	TransportType	Used in	FSPM
Input		Output	
Variable index		Transport type	
Function			
TransportType depends upon the command number (variable index). It has one of the following values: Request-unicast, Search-broadcast, Request-broadcast, Response-unicast, Response-broadcast, Block-transfer request, Block-transfer response, Publish-broadcast and Publish-unicast.			

Table 346 – Function PriorityType ()

Name	Priority	Used in	FSPM
Input		Output	
Variable index		Priority	
Function			
Transport type depends upon the command number (variable index). It has one of the following values: Command, Process-data, Normal and Alarm.			

Table 347 – Function FormReqApdu ()

Name	FormReqApdu	Used in	FSPM
Input		Output	
User request		APDU	
Function			
Uses the parameters to form the request APDU as specified in 8.2.2.			

Table 348 – Function FormRspAdu ()

Name	FormRspAdu	Used in	FSPM
Input		Output	
User response		APDU	
Function	Uses the parameters to form the response APDU as specified in 8.2.2.		

Table 349 – Function RetrieveCommand ()

Name	RetrieveCommand	Used in	FSPM
Input		Output	
APDU		Variable index	
Function	Retrieve the variable index (command number) from the APDU.		

Table 350 – Function CommandErr ()

Name	CommandErr	Used in	FSPM
Input		Output	
APDU		True False	
Function	Looks at the Response code field in the APDU. If its value is not zero (0) and if the Value field does not exist, then this function returns True; else it returns False.		

Table 351 – Function ResponseCode ()

Name	ResponseCode	Used in	FSPM
Input		Output	
APDU		Response code	
Function	This function returns Response code field in the APDU. It can be either communication error or command response code.		

Table 352 – Function DeviceStat ()

Name	DeviceStat	Used in	FSPM
Input		Output	
APDU		Application process status	
Function	This function returns “Application process status” field in the APDU.		

Table 353 – Function Value ()

Name	Value	Used in	FSPM
Input		Output	
APDU, transport_type		Data	
Function	This function returns request or response data octets of the APDU. If transport_type is response, then there response code in addition to Value field in the APDU.		

8.4.4 Parameters of FSPM/ARPM primitives

The parameters used with the primitives exchanged between the FSPM and the ARPM are described in Table 354.

Table 354 – Parameters used with primitives exchanged between FSPM and ARPM

Parameter name	Description
remote_address	This parameter is the network address of the destination device in Transaction.req, Transaction.rsp and Send.req; source device in Transaction.ind and Receive.ind.
transport_type	It indicates the type of transport pipe to use.
priority	It indicates the priority of the request.
req_user_data	This parameter conveys request APDU.
rsp_user_data	This parameter conveys response APDU.
status	This parameter conveys the status of response confirmation.

8.5 Application relationship protocol machines (ARPMs)

8.5.1 AREP mapping to transport layer

8.5.1.1 General

This section describes the mapping of the AL to the Fieldbus transport layer. It does not redefine the TLSAP attributes or TLME attributes that are defined in the transport layer specification; rather, it defines how they are used by each of the AR classes. A means to configure and monitor the values of these attributes is provided by network management.

The following class definitions describe the TSAP attributes and the TLME attributes required to support each of the AREP classes.

NOTE Undefined attributes use the same definitions as those previously defined.

8.5.1.2 TL mapping of client AREP class

8.5.1.2.1 Client AREP class formal model

The TL Mapping attributes and their permitted values and the TL services used with the Client AREP class are defined in this subclause.

```

CLASS: Client
PARENT CLASS: Top
ATTRIBUTES:
1. (m) Key Attribute: Remote_address
DLL SERVICES:
1. (m) OpService: TL-DATA-EXCHANGE.req
2. (m) OpService: TL-DATA-EXCHANGE.cnf
3. (m) OpService: TL-DATA-TRANSFER.ind

```

8.5.1.2.2 Attributes

8.5.1.2.2.1 Remote_address

This attribute specifies the remote address to which request APDUs are sent, or from which confirm APDUs are received.

8.5.1.2.3 TL services

8.5.1.2.3.1 TL-DATA-EXCHANGE.req

This service is used in Master device to transfer an APDU to the remote Slave TL user, which shall send a response.

8.5.1.2.3.2 TL-DATA-EXCHANGE.cnf

The TL uses this service to transfer the response APDU from a remote Slave TL user in response to a prior request.

8.5.1.2.3.3 TL-DATA-TRANSFER.ind

This service is used in Master device to receive an APDU from the remote Slave TL user without a prior request.

8.5.1.3 TL mapping of server AREP class

8.5.1.3.1 Server AREP class formal model

The TL Mapping attributes and their permitted values and the TL services used with the Server AREP class are defined in this subclause.

CLASS: Server
PARENT CLASS: Top
ATTRIBUTES:

1. (m) Key Attribute: Address

DLL SERVICES:

1. (m) OpsService: TL-DATA-EXCHANGE.ind
2. (m) OpsService: TL-DATA-EXCHANGE.rsp
3. (m) OpsService: TL-DATA-TRANSFER.req

8.5.1.3.2 Attributes

8.5.1.3.2.1 Address

This attribute specifies the address that the Slave device uses to receive and send TPDU.

8.5.1.3.3 TL services

8.5.1.3.3.1 TL-DATA-EXCHANGE.ind

The TL uses this service to transfer the indication APDU from a remote Master TL user, which shall send a response.

8.5.1.3.3.2 TL-DATA-EXCHANGE.rsp

This service is used in Slave device to transfer an APDU to the remote Master TL user, which is a response to prior indication.

8.5.1.3.3.3 TL-DATA-TRANSFER.req

This service is used in Master device to transfer an APDU to the remote Master TL user, which is not a response to an indication.

8.5.2 Application relationship protocol machines (ARPMs)

8.5.2.1 AR protocol machine (ARPM) for client AREP

8.5.2.1.1 Client ARPM states

The defined states and their descriptions of the Client ARPM are shown in Table 355 and Figure 58.

Table 355 – Client ARPM states

State name	Description
IDLE	The AREP is not active.
WAIT for CONFIRM	The AREP has sent a request PDU to a Server and is waiting for a confirmation from the Server.

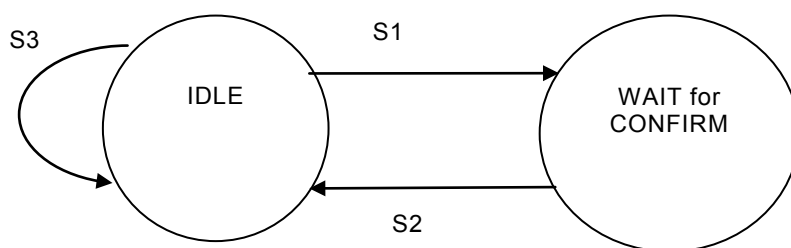


Figure 58 – State transition diagram of the client ARPM

8.5.2.1.2 Client ARPM state table

Table 356 specifies the Client ARPM state machine.

Table 356 – Client ARPM state table

#	Current state	Event or condition => action	Next state
S1	IDLE	Transaction.req => DTC_req { called_address := remote_address, transport_type := transport_type, priority := priority, req_user_data := req_user_data }	WAIT for CONFIRM
S2	WAIT for CONFIRM	DTC_cnf => Transaction.cnf { rsp_user_data := rsp_user_data, status := status }	IDLE
S3	IDLE	DTU_ind && CommandErr (rsp_user_data) == "False" => Receive.ind { remote_address := calling_address, transport_type := transport_type, rsp_user_data := rsp_user_data }	IDLE

NOTE The parameters transport_type, priority and status are present only for wireless Type 20 fieldbus.

8.5.2.2 AR protocol machine (ARPM) for server AREP

8.5.2.2.1 Server ARPM states

The defined states and their descriptions of the Server ARPM are shown in Table 357 and Figure 59.

Table 357 – Server ARPM states

State name	Description
IDLE	The AREP is not active.
WAIT for RESPONSE	The AREP has received a PDU from DLL, has sent the corresponding indication to the AL user and is waiting for a response from the AL user

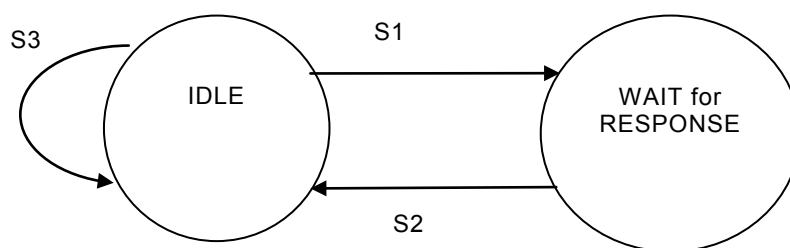


Figure 59 – State transition diagram of the server ARPM

8.5.2.2.2 Server ARPM state table

Table 358 specifies the Server ARPM state machine.

Table 358 – Server ARPM state table

#	Current state	Event or condition => action	Next state
S1	IDLE	DTC_ind => Transaction.ind { remote_address := calling_address, transport_type := transport_type, req_user_data := req_user_data }	WAIT for RESPONSE
S2	WAIT for RESPONSE	Transaction.rsp => DTC_rsp { called_address := remote_address, rsp_user_data := rsp_user_data }	IDLE
S3	IDLE	Send.req => DTU_req { called_address := remote_address, transport_type := transport_type, priority := priority, rsp_user_data := rsp_user_data }	IDLE

NOTE The parameter transport_type is present only for wireless Type 20 fieldbus.

8.5.3 AREP state machine primitive definitions

The primitives exchanged between the TMPM and the ARPM are specified in Table 359 and Table 360.

Table 359 – Primitives issued from ARPM to TMPM

Primitive name	Source	Associated parameters	Functions
DTC_req	ARPM	called_address, transport_type, priority, req_user_data	This primitive is used to request the TMPM to transfer a confirmed request APDU to a Slave device
DTC_rsp	ARPM	called_address, rsp_user_data	This primitive is used to request the TMPM to transfer a response APDU to a Master device
DTU_req	ARPM	called_address, transport_type, priority, req_user_data	This primitive is used to request the TMPM to transfer a unconfirmed request APDU to a Master device

Table 360 – Primitives issued by TPM to ARPM

Primitive name	Source	Associated parameters	Functions
DTC_ind	TPM	calling_address, transport_type, priority, req_user_data	This primitive is used to pass an APDU received as a Transport layer service data unit to the Server ARPM
DTC_cnf	TPM	rsp_user_data, status	This primitive is used to convey a response APDU received from the Slave device
DTU_ind	TPM	calling_address, transport_type, priority, rsp_user_data	This primitive is used to pass an APDU received as a Transport layer service data unit to the Client ARPM

8.5.4 AREP state machine functions

ARPM does not use any functions.

8.6 TL mapping protocol machine (TPM)

8.6.1 TPM states

The defined states and their descriptions of the TPM are shown in Table 361 and Figure 60.

Table 361 – TPM state descriptions

State name	Description
ACTIVE	The TPM in the ACTIVE state is ready to transmit or receive primitives to or from the transport layer and the ARPM

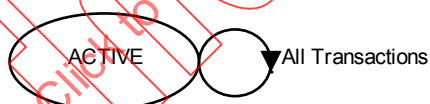


Figure 60 – State transition diagram of TPM

8.6.2 TPM state machines

Table 362 and Table 363 specify the TPM state machine.

Table 362 – TPM state Table – Client transactions

#	Current state	Event or condition => action	Next state
S1	ACTIVE	DTC_req => TL-DATA-EXCHANGE.req { destination_address := called_address, transport_type := transport_type, priority := priority, tls_user_data := req_user_data }	ACTIVE
S2	ACTIVE	TL-DATA-EXCHANGE.cnf => DTC_cnf { rsp_user_data := tls_user_data, status := status }	ACTIVE

#	Current state	Event or condition => action	Next state
S2	ACTIVE	TL-DATA-TRANSFER.ind => DTU_ind calling_address := source_address, transport_type := transport_type, priority := priority, rsp_user_data:= tls_user_data }	ACTIVE

Table 363 – TPM state Table – Server transactions

#	Current state	Event or condition => action	Next state
S1	ACTIVE	DTC_rsp => TL-DATA-EXCHANGE.rsp { tls_user_data := rsp_user_data }	ACTIVE
S2	ACTIVE	TL-DATA-EXCHANGE.ind => DTC_ind { calling_address := source_address, transport_type := transport_type, priority := priority, req_user_data:= tls_user_data }	ACTIVE
S3	ACTIVE	DTU_req => TL-DATA-TRANSFER.req { destination_address := called_address, transport_type := transport_type, priority := priority, tls_user_data := rsp_user_data }	ACTIVE

8.6.3 Primitives exchanged between transport layer and TPM

The primitives exchanged between the data-link layer and the TPM are specified in Table 364.

Table 364 – Primitives exchanged between network layer and TPM

Primitive name	Source	Associated parameters
TL-DATA-EXCHANGE.req	TPM	destination_address, transport_type, priority, tls_user_data
TL-DATA-EXCHANGE.ind	Transport layer	source_address, transport_type, priority, tls_user_data
TL-DATA-EXCHANGE.rsp	TPM	tls_user_data
TL-Data-exchange.cnf	Transport layer	tls_user_data, status
TL-DATA-TRANSFER.req	TPM	destination_address, transport_type, priority, tls_user_data
TL-DATA-TRANSFER.ind	Transport layer	source_address, transport_type, priority, tls_user_data

8.6.4 Functions used by TPM

TPM does not use any functions.

9 Communication Profile Family 9 (HART™)

9.1 General Overview

The principles and conventions for a Communication Profile Family (CPF) and a Communication Profile (CP) are specified in IEC 61784-1 and apply also to this standard except the conformance statement.

NOTE The concept of Communication Profiles and the association to Types is given in IEC/TR 61158-1.

Communication Profile Family 9 defines a profile based on Type 20 IEC 61158 series, which corresponds to parts of a communication system commonly known as the HART™. There are two profiles:

- CP 9/1 also known as HART™ Universal Command set, and
- CP 9/2 also known as WirelessHART™.

All of the services and protocol elements of IEC 61158-5-20 and IEC 61158-6-20 application layer are referenced by IEC 61784-1 CP 9/1. The CP 9/2 is based on this standard and references to the services and protocol elements of IEC 61158-5-20 and IEC 61158-6-20 application layer.

Table 365 gives an overview of the specified profile sets.

Table 365 – CPF 9: overview of profile sets

Layer	Profile 9/1	Profile 9/2
Application	IEC 61158-5-20 IEC 61158-6-20	Clause 7 and 8
Data-link	—	Clause 5 and 6
Physical	—	Clause 4 with references to IEEE 802.15.4-2006

For CP 9/2 there are additional behaviors specified:

- Security in Annex A,
- Network Management in Annex C, and
- Gateway in Annex D.

9.2 Profile 9/1

CP 9/1 is specified in IEC 61784-1.

9.3 Profile 9/2

9.3.1 Conformance statement

IEC 61784-2, Clause 4 applies, but the project number shall be of this standard instead of IEC 61784-2 and the use of <Type> is mandatory.

A statement of compliance with a communication profile (CP) of this standard shall be stated as either

- Compliance to IEC 62591:2010 CP 9/2 <organization name><Identifier>, or
- Compliance to IEC 62591 (Ed.1.0) CP 9/2 <organization name><Identifier>.

where the <Identifier> is the name of the device types specified in 9.3.2 and a unique <organization name> specifies the corresponding values:

- Manufacturer ID (see IEC 61158-6-20:2010 **Error! Bookmark not defined.**, Table 7),
- Expanded Device Type (see IEC 61158-6-20:2010 **Error! Bookmark not defined.**, Table 7),
- Command Rev (see IEC 61158-6-20:2010 **Error! Bookmark not defined.**, Table 7), and
- OUI (see 6.2.1.2.4).

These parameters shall be unique for the CP 9/2 devices that are within one network and in range of other networks.

NOTE 1 A device may support more than one device type of 9.3.2.

NOTE 2 Organization name could be for example a vendor or group or consortium name.

EXAMPLE The HART™ Communication Foundation would mark their compliant devices with:

IEC 62591:2009 CP 9/2 WirelessHART™ Field Device

or another organization would name their compliant devices with:

IEC 62591:2009 CP 9/2 MyOrganization_name Field Device

If WirelessHART™ is intended to be used as <organization name> then

- the parameters Manufacturer ID and Expanded Device Type shall be assigned by HCF;

NOTE 3 See document HCF_SPEC-183 for already assigned codes.

- the Engineering unit codes shall be assigned by HCF;

NOTE 4 See document HCF_SPEC-183 for already assigned codes.

- the parameter Command Rev shall be set to a valid code of HCF specification revision numbers (for example 7.2);
- a OUI, which has been obtained by HCF, shall be set to 0x001B1E; and
- a certificate from HCF shall be obtained based on a successful compliance test.

9.3.2 Device types

9.3.2.1 General

Figure 61 shows various network components specified for CPF 9. The plant automation network can be based on a standard protocol or it can be a proprietary network. Its specification is out of the scope of this standard. The gateway is connected to the wireless network through its network access point(s). More than one network access points can be used to increase the throughput and improve the overall reliability of the wireless network.

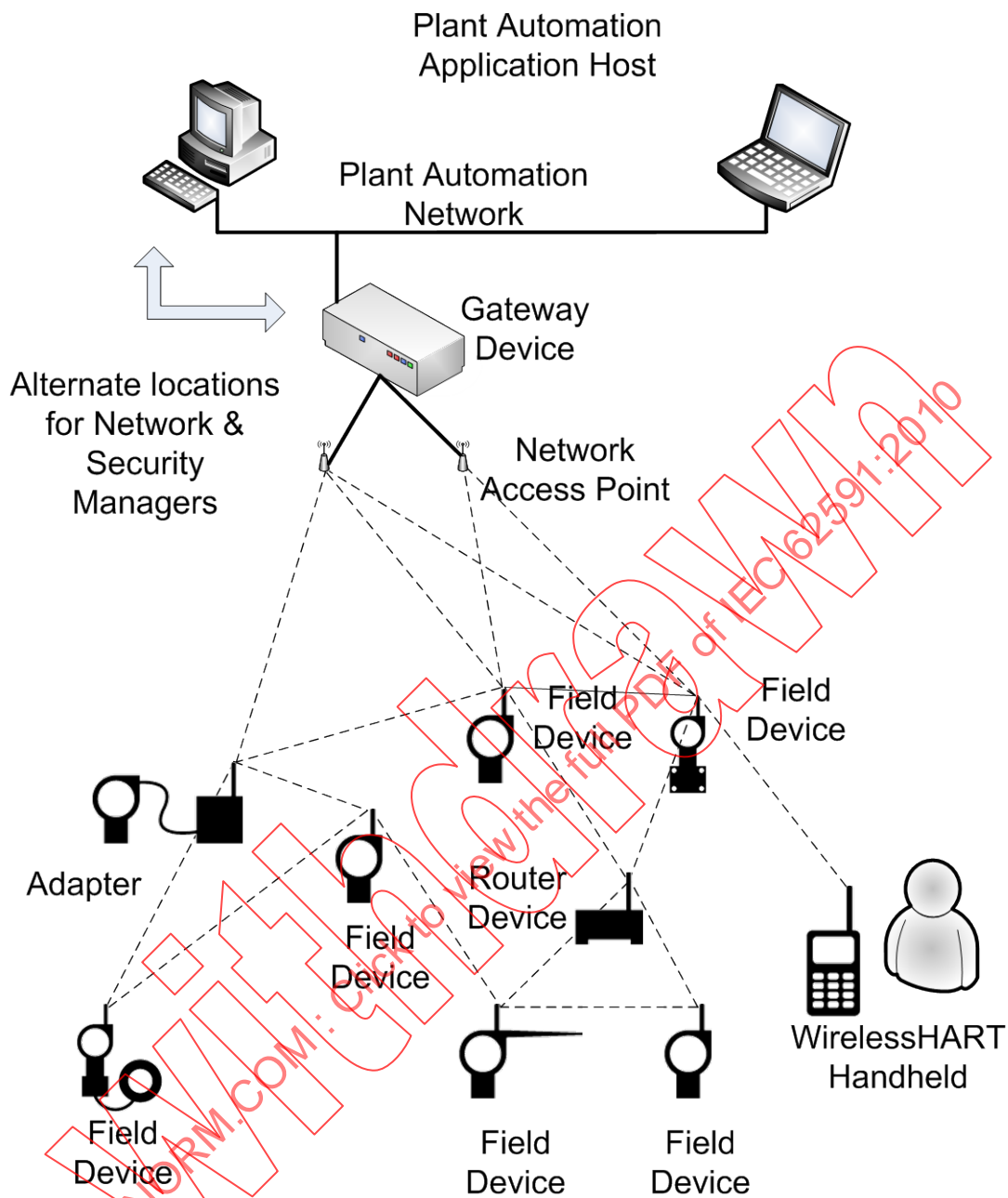


Figure 61 – Wireless network components

Any device directly connected to the wireless network is one of the following types of network device:

- Field device,
- Adapter,
- Router,
- Gateway,
- Handheld device.

All wireless network devices transmit and receive PDUs and perform the basic functions necessary to support network formation and maintenance. All network devices are able to source and sink packets and capable of routing packets received from other devices in the network. Each network device also has attributes holding information on update rates,

sessions and device resources covering items such as the size of the superframe, etc. Each network device contains a list of Neighbor devices that it has identified during its listening operations.

All network devices have a 5 octet Unique ID assigned at the time of manufacturing the device. The EUI-64 address is a concatenation of the 24-bit OUI value assigned by the IEEE Registration Authority and a 40-bit extension identifier, which is the 5 octet Unique ID.

9.3.2.2 Field device

A Field device is directly connected to the field Process to measure it or to control it or it is connected to plant equipment. It is a producer and consumer of packets and is capable of routing packets received from other network devices compliant to this standard. The Field Device may be line, loop, battery powered or powered in some other fashion. Field device may also support traditional 4 mA to 20 mA current loop signaling. The Field device shall have a maintenance port for example according to IEC 61784-1 CP 9/1 and may only have a wire connection to the process automation system.

9.3.2.3 Adapter

An Adapter device provides physical and logical connectivity to non-native communicating device(s) to the wireless network. It uses internal routing tables to coordinate traffic flow between the wireless network and its non-native communicating sub-device(s). An Adapter is not directly connected to the field process. It supports all commands required of the Field device (see 9.3.2.2) including the publishing of process data on behalf of the non-native connected field device. In response to Identify command, it sets Device Flag to 0x04 (Protocol bridge device).

Non-native communicating devices can be HART™ devices according to IEC 61784-1 CP 9/1 or 4 mA to 20mA devices.

NOTE The gateway connects the Field devices to the host system or plant automation network. The Adapter makes the non-native device look like a Field device.

9.3.2.4 Router

A Router device forwards packets from one network device to another. It uses its graphs and connections to find the next hop neighbor to which the received packet has to be forwarded. In general standalone routers are not required since all network devices support routing. However it may be beneficial (for example to extend the network, or to save the power of a Field devices in the network) to add additional devices to improve routing in the network. A router is not connected to the process and does not operate as a gateway. When a network access point does not source the network clock it shall synchronize to the network devices acting as its clock parents.

9.3.2.5 Gateway

Gateway device connects the wireless network according to this standard to a plant automation network, allowing data to flow between the two networks. It provides host applications access to the network devices. A gateway device can be used to convert from one protocol to another, as go-between two or more networks that use the same protocol, or to convert commands and data from one format to another. The wireless network also uses the gateway as the source for the synchronized clock used by the timeslots and superframes.

A gateway requires a Network Access Point, which is logically part of the gateway, to connect to the wireless network. The gateway can use more than one network access point to improve the effective throughput and reliability of the network. Network access points propagate the clock to the network. There is always at least one network access point providing the clock to the network.

The Annex D specifies the functions performed by the gateway.

9.3.2.6 Handheld device

A handheld device is used in the installation, control, monitoring, and maintenance of network devices. It is portable equipment and it is operated by the plant personnel. There are two approaches to connect handheld device to a network device.

- Wireless connected handheld device that communicates directly to the wireless network according to this standard. This handheld device joins the network as a wireless field device. When operating with a target network device that is connected to a wireless network, the handheld device operates in a special mode that allows it to communicate with one device at a time. Handheld is not required to support routing.
- Plant automation network connected handheld device that connects to the plant automation network through some other networking technology. This device communicates with the network devices through the gateway device in the same way as the external plant automation host does. It appears as a host application to the network device.

9.3.2.7 Network manager

The network manager does not directly connect to the wireless network, but it is logically part of the gateway. It has its own Unique ID and thus it communicates with other network devices using the application layer protocol of this standard. The network manager is responsible for configuration of the network; scheduling communication between network devices; management of the routing tables and monitoring and reporting the health of the wireless network. It contains a complete list of network devices and ensures each has a network-unique short 16-bit Nickname.

As part of its system functions, the network manager collects performance and diagnostic information. This information is accessible during run-time, making it possible to view and analyze the behavior of the overall network. If problems are detected, the reconfiguration of the network is performed while the network is operating. This network grooming is performed continuously as the overall network operation and performance varies due to changes in network load and environmental conditions.

The Annex C specifies the functions performed by the network manager.

9.3.2.8 Security manager

Join, network and session keys shall be provided to the network manager and join keys shall be provided to network devices. These keys are used for device authentication and encryption of data in the network. The security manager is responsible for the generation, storage, and management of keys. There is one Security manager associated with each wireless network. One security manager may service multiple wireless networks. The security manager function is required to manage these keys. The security manager may be a centralized function in some plant automation networks, servicing more than just one wireless network and in some cases other networks and applications.

The security manager communicates directly with the network manager for the distribution of the keys. The functional requirements and the operation of the security manager and its communication with the network manager are outside the scope of this standard.

9.3.3 Physical layer

9.3.3.1 General overview

Wireless devices (see 9.3.2) use Radio Frequencies (RF) to communicate between devices. The distance between two devices can be 100 meters or more depending upon the transmitter power, antenna type and lack of obstruction between two devices. Wireless devices can be

powered either by mains power, battery, energy harvesting devices (for example solar, vibration, etc), or a combination there of.

This specification defines the Physical Layer (PhL). This specification is used for devices implementing wireless mesh network communication as defined by this standard. The physical layer is responsible for the signaling method, signal strength, device sensitivity, and environment for sending bits across the network media. This wireless PhL is based on the 2 450 MHz DSSS physical layer according to IEEE STD 802.15.4-2006 and employing O-QPSK modulation. This profile specification includes the following:

- the PhL requirements many of which are adopted directly from IEEE STD 802.15.4-2006 including specific requirements on channels, transmit power, and receive sensitivity, and
- the PhL requirements to ensure interoperability among wireless devices.

The PhL of this standard deals with radio characteristics making a network connection. It associates with any part of the network structure that doesn't process information in any way. The physical layer is responsible for sending the bits across the network media. It does not define what a bit is or how it is used merely how it's sent. The PhL is responsible for transmitting and receiving the data.

Items defined by the PhL include RF transmitters, receivers, and transceivers. Any item that does not process information but is required for the sending and receiving of data is defined by this layer.

9.3.3.2 Requirements based on IEEE STD 802.15.4-2006

Table 366 specifies the IEEE STD 802.15.4-2006 PhL selection for a communicating device and it's MAU.

Table 366 – CP 9/2: PhL protocol selection

Clause of IEEE 802.15.4-2006	Header	Presence	Constraints
6.1	General requirements and definitions	—	—
6.1.1	Operating frequency range	Partial	Only 2 450 MHz band is included – see Table 367
6.1.2	Channel assignments	Partial	Only Channels 11-25 of channel page 0 are included – see Table 368
6.1.3	Minimum long interframe spacing (LIFS) and short interframe spacing (SIFS) periods	Partial	Only 2 450 MHz band is included
6.1.4	RF power measurement	YES	—
6.1.5	Transmit power	YES	—
6.1.6	Out-of-band spurious emission	YES	—
6.1.7	Receiver sensitivity definitions	YES	—
6.2	PHY service specifications	NO	See 4.1.2 for replacement of this subclause
6.3	PPDU format	YES	—
6.4	PHY constants and PIB attributes	—	—
6.4.1	PHY constants	YES	—
6.4.2	PHY PIB attributes	Partial	See 9.3.3.3.8
6.5	2 450 MHz PHY specifications	YES	See 9.3.3.3.7 for further explanation
6.6	868/915 MHz band binary phase-shift keying (BPSK) PHY specifications	NO	Only 2 450 MHz band is included
6.7	868/915 MHz band (optional) amplitude shift keying (ASK) PHY specifications	NO	Only 2 450 MHz band is included
6.8	868/915 MHz band (optional) O-QPSK PHY specifications	NO	Only 2 450 MHz band is included
6.9	General radio specifications	—	—
6.9.1	TX-to-RX turnaround time	YES	—
6.9.2	RX-to-TX turnaround time	YES	—
6.9.3	Error-vector magnitude (EVM) definition	YES	—
6.9.4	Transmit center frequency tolerance	YES	—
6.9.5	Transmit power	NO	See 9.3.3.3.5 for replacement of this subclause
6.9.6	Receiver maximum input level of desired signal	YES	—
6.9.7	Receiver ED	YES	—
6.9.8	Link quality indicator (LQI)	YES	—
6.9.9	Clear channel assessment (CCA)	Partial	Only CCA mode 2 (carrier sense) is included

9.3.3.3 Other physical layer requirements

9.3.3.3.1 Frequency Allocations

The transceiver shall employ DSSS and operate in the license-free ISM band as shown in Table 367.

Table 367 – Transceiver Specifications

Frequency	Communication rate (kchip/s)	Modulation	Bit rate (kb/s)	Symbol rate (ksymbol/s)	Symbols
2 400-2 483,5 MHz	2 000	O-QPSK	250	62,5	16-ary Orthogonal

9.3.3.3.2 Channel numbers and frequency assignments

The 2 450 MHz band supports channel assignments 11-26 as specified by IEEE STD 802.15.4-2006. Since channel 26 is not compliant with the regulation of some world regions, it is not included in this profile.

The frequency assignment for the channels 11 – 25 is shown in Table 368.

Table 368 – Frequency assignments

Channel number	Frequency (MHz)	Channel number	Frequency (MHz)	Channel number	Frequency (MHz)
11	2 405	16	2 430	21	2 455
12	2 410	17	2 435	22	2 460
13	2 415	18	2 440	23	2 465
14	2 420	19	2 445	24	2 470
15	2 425	20	2 450	25	2 475

9.3.3.3.3 Radio transceivers

A communicating device using this profile can be implemented with readily available 'off the shelf' or custom radio transceivers that are compliant to IEEE STD 802.15.4-2006 and meets the criteria listed below.

This standard is intended to conform to worldwide regulations in Europe, Canada, Japan, China, the United States and other locations.

9.3.3.3.4 Unspecified or improved required radio performance

This profile includes following that is not specified in IEEE STD 802.15.4-2006.

- Time to switch between channels shall be 12-symbol periods max (0,192 ms).
- Recommended cold start radio turn-on time is 4 ms maximum. It includes frequency lock and settling time.

9.3.3.3.5 Transmit power

Transmit power shall be the Equivalent Isotropic Radiated Power (EIRP) of the device. All communicating device using this profile shall provide a nominal EIRP of +10 dBm (10 mW) ± 3 dB.

In addition, manufacturers shall ensure that their device complies with localized transmit power regulations applicable to the device's installation.

In both cases, the rated transmit power shall be representative of the manufacturer's production device with power source (batteries, solar, or otherwise) at 95% to 100% of rated capacity. All devices should be provided with an Omni-direction antenna. The rated transmit power data including maximum output power and radiation patterns for all antennae supported by the product shall be available in the product manual or upon request.

9.3.3.3.6 Output power control

The device power level shall be programmable at discrete, monotonic levels from -10 dBm to +10 dBm EIRP (within ± 4 dBm of error in actual power level). The application layer command is available to control the output power.

9.3.3.3.7 Receiver sensitivity

Device sensitivity is defined as the combined sensitivity of the receiver and the antenna as a system. All devices shall achieve a system (radio, circuit card, cabling, and connectors) device sensitivity of -85 dB or better.

9.3.3.3.8 PHY PIB attributes

This profile includes the attributes specified in the IEEE STD 802.15.4-2006, 6.4.2 and additional attributes that apply only to this profile. Some of these attributes are assigned a constant value for this profile as shown in Table 369.

Table 369 – PHY PIB attributes

Attribute	Comment
phyCurrentChannel	Set by PH-ENABLE request
phyChannelsSupported	Only channels 11 – 25 are supported
phyTransmitPower	The values from ‘-10 dBm to +10 dBm’ are supported
phyCCAMode	Only value 2 (carrier sense) is included
phyCurrentPage	Only value 0 is included
phyMaxFrameDuration	Only value ‘266’ is included
phySHRDuration	Only value ‘10’ is included
phySymbolsPerOctet	Only value ‘2’ is included
SleepState	It takes one of the two values – sleep, awake
RcvOvewrflowEnable	If true, then it enables the receive overflow error indication

9.3.4 Data-link layer

9.3.4.1 DLL service selection

Table 370 specifies the DLL service selection within Clause 5 for this profile. It includes the services provided by data-link, network and transport layers.

Table 370 – CP 9/2: DLL service selection

Clause	Header	Presence	Constraints
5.1	Data-link layer services	—	—
5.1.1	Facilities of the data-link layer services	YES	—
5.1.2	QoS for message service	YES	—
5.1.3	Sequence of primitives	YES	—
5.1.4	DL-TRANSMIT Service	YES	—
5.1.5	DL-RECEIVE service	—	This service is optional. It is normally used for network troubleshooting in "promiscuous operating mode".
5.1.6	DL-DISCONNECT service	YES	—
5.1.7	DL-PATH-FAILURE service	YES	—
5.1.8	DL-ADVERTISE service	YES	—
5.1.9	DL-NEIGHBOR service	YES	—
5.1.10	DLM-SET	YES	—
5.1.11	DLM-GET	YES	—
5.1.12	DLM-ACTION	YES	—
5.1.13	DLM-EVENT	YES	—
5.2	Network layer services	YES	—
5.3	Transport layer services	YES	—

9.3.4.2 DLL protocol selection

Table 371 specifies the DLL service selection within Clause 6 for this profile. It includes the protocol specifications of data-link, network and transport layers.

Table 371 – CP 9/2: DLL protocol selection

Clause	Header	Presence	Constraints
6.1	Overview	YES	—
6.2	Logical link control	YES	—
6.3	Medium access control	YES	—
6.4	DL-management-information	YES	—
6.5	Network layer	YES	—
6.6	Transport layer	YES	—

9.3.5 Application layer

9.3.5.1 AL service selection

9.3.5.1.1 AL service selection common to all device types

IEC 61158-5-20 applies. Table 372 shows the AL service selection within Clause 7 for this profile.

Table 372 – CP 9/2: AL service selection

Clause	Header	Presence	Constraints
7.1	Common concepts	YES	—
7.2	Data Type ASE	YES	—
7.3	Communication model specification	—	—
7.3.1	Common parameters	YES	—
7.3.2	ASEs	—	—
7.3.2.1	Virtual field device ASE	YES	—
7.3.2.2	Variable ASE	YES	—
7.3.2.3	Action ASE	YES	—
7.3.2.4	Device application services	Partial	See Table 373
7.3.2.5	Layer management services	Partial	See Table 374
7.3.2.6	Gateway and network manager services	Partial	See Table 375
7.3.2.7	Network management configuration services	Partial	See Table 376
7.3.3	ARs	YES	—

9.3.5.1.2 Variable ASE service selection

This clause specifies the services that are required for various device types. The following convention is used:

- X: the service is required;
- O: the service optional
- Blank cell: the service is not required.

Table 373, Table 374, Table 375 and Table 376 specify the Variable ASE service selection within Clause 7 for this profile.

Table 373 – CP 9/2: Device application service selection

Clause	Header	Var ID ^a	Field device	Adapter	Gateway	Hand held	Network manager
7.3.2.4.1	Read primary variable	1	X	X			
7.3.2.4.2	Read loop current and percent of range	2	X	X			
7.3.2.4.3	Read dynamic variables and loop current	3	X	X			
7.3.2.4.4	Write loop configuration	6	X	X			
7.3.2.4.5	Read loop configuration	7	X	X			
7.3.2.4.6	Read dynamic variable families classifications	8	X	X			
7.3.2.4.7	Read device variables with status	9	X	X			
7.3.2.4.8	Read message	12	X	X			
7.3.2.4.9	Read tag, descriptor and date	13	X	X			
7.3.2.4.10	Read primary variable transducer information	14	X	X			
7.3.2.4.11	Read device information	15	X	X			
7.3.2.4.12	Read final assembly number	16	X	X			
7.3.2.4.13	Write message	17	X	X			
7.3.2.4.14	Write tag, descriptor and date	18	X	X			
7.3.2.4.15	Write final assembly number	19	X	X			
7.3.2.4.16	Read Long_tag	20	X	X			
7.3.2.4.17	Write Long_tag	22	X	X			
7.3.2.4.18	Reset configuration changed flag	38	X	X	X		
7.3.2.4.19	Perform self test	41	X	X	X		
7.3.2.4.20	Perform device reset	42	X	X	X		
7.3.2.4.21	Read additional device status	48	X	X	X		
7.3.2.4.22	Reset more status available						
7.3.2.4.23	Read device variable information	54	X	X			
7.3.2.4.24	Write device variable	79	X	X			
7.3.2.4.25	Read real-time clock	90	X	X			
7.3.2.4.26	Write publish data period	103	X	X	X		
7.3.2.4.27	Write publish data trigger	104	X	X	X		
7.3.2.4.28	Read publish data mode configuration	105	X	X	X		
7.3.2.4.29	Flush delayed responses	106	X	X	X		
7.3.2.4.30	Write publish data device variables	107	X	X	X		
7.3.2.4.31	Write publish data mode command number	108	X	X	X		
7.3.2.4.32	Write publish data mode control	109	X	X	X		
7.3.2.4.33	Read event notification summary	115	X	X	X		
7.3.2.4.34	Write event notification bit mask	116	X	X	X		
7.3.2.4.35	Write event notification timing	117	X	X	X		
7.3.2.4.36	Write event notification control	118	X	X	X		
7.3.2.4.37	Write event notification acknowledgement	119	X	X	X		

^a VAR ID is Numeric Identifier in the service definition. It is also referred to as HART™ command number.

Table 374 – CP 9/2: Layer management service selection

Clause	Header	Var ID ^a	Field device	Adapter	Gate way	Hand held	Network manager
7.3.2.5.1	Write Join_key	768	X	X		X	
7.3.2.5.2	Read join status	769	X	X		X	
7.3.2.5.3	Write active_search_shed_time	770	X	X		X	X
7.3.2.5.4	Write join mode configuration	771	X	X		X	
7.3.2.5.5	Read join mode configuration	772	X	X		X	X
7.3.2.5.6	Write network_ID	773	X	X	X	X	X
7.3.2.5.7	Read network_ID	774	X	X	X	X	X
7.3.2.5.8	Write Network_tag	775	X	X	X	X	
7.3.2.5.9	Read Network_tag	776	X	X	X	X	
7.3.2.5.10	Read wireless device capabilities	777	X	X			O
7.3.2.5.11	Read battery life	778	X	X	X		O
7.3.2.5.12	Report device health	779	X	X			X
7.3.2.5.13	Read neighbor health list	780	X	X			X
7.3.2.5.14	Read device nickname	781	X	X	X		
7.3.2.5.15	Read session list	782	X	X	X		
7.3.2.5.16	Read superframe list	783	X	X			
7.3.2.5.17	Read link list	784	X	X			
7.3.2.5.18	Read graph list	785	X	X			
7.3.2.5.19	Read neighbor attribute flag	786	X	X			
7.3.2.5.20	Read neighbor signal level	787	X	X			X
7.3.2.5.21	Report path down alarm	788	X	X			X
7.3.2.5.22	Report source route failed alarm	789	X	X			X
7.3.2.5.23	Report graph route failed alarm	790	X	X			X
7.3.2.5.24	Report transport layer failed alarm	791	X	X			X
7.3.2.5.25	Write UTC time mapping	793	X	X	O		
7.3.2.5.26	Read UTC time mapping	794	X	X	X		X
7.3.2.5.27	Write timer interval	795	X	X			X
7.3.2.5.28	Read timer interval	796	X	X			X
7.3.2.5.29	Write radio power output	797	X	X			X
7.3.2.5.30	Read radio power output	798	X	X	X		X
7.3.2.5.31	Request timetable connection	799	X	X			X
7.3.2.5.32	Read timetable list	800	X	X			
7.3.2.5.33	Delete timetable	801	X	X			X
7.3.2.5.34	Read route list	802	X	X			
7.3.2.5.35	Read source route	803	X	X			
7.3.2.5.36	Read CCA mode	804	X	X	X		
7.3.2.5.37	Write CCA mod	805	X	X			X
7.3.2.5.38	Read handheld superframe	806	X	X	X		X
7.3.2.5.39	Write handheld superframe	807	X	X		X	
7.3.2.5.40	Read packet hop-to-live	808	X	X	X		X
7.3.2.5.41	Write packet hop-to-live	809	X	X			O
7.3.2.5.42	Read Join_priority	810	X	X	X		
7.3.2.5.43	Write Join_priority	811	X	X			X
7.3.2.5.44	Read receive Priority_threshold	812	X	X	X		
7.3.2.5.45	Write receive Priority_threshold	813	X	X			X
7.3.2.5.46	Read device list	814	O	O	X		
7.3.2.5.47	Add device list entry	815	O	O	X		
7.3.2.5.48	Delete device list entry	816	O	O	X		
7.3.2.5.49	Read channel blacklist	817	X	X	X		
7.3.2.5.50	Write channel blacklist	818			X		
7.3.2.5.51	Read back-off exponent	819	X	X			
7.3.2.5.52	Write back-off exponent	820	X	X			X
7.3.2.5.53	Write network access mode	821			X		
7.3.2.5.54	Read network access mode	822			X		
7.3.2.5.55	Request session	823				X	X

^a VAR ID is Numeric Identifier in the service definition. It is also referred to as HART™ command number.

Table 375 – CP 9/2: Gateway and network manager service selection

Clause	Header	Var ID ^a	FD	Adap ter	Gate way	Hand held	Network manager
7.3.2.6.1	Read network device identity using Unique ID	832			X		
7.3.2.6.2	Read device's neighbor health	833			X		
7.3.2.6.3	Read network topology information	834			X		
7.3.2.6.4	Read publish data message list	835			X		
7.3.2.6.5	Flush cached responses for a device	836			X		
7.3.2.6.6	Write update notification bit mask for a device	837			X		
7.3.2.6.7	Read update notification bit mask for a device	838			X		
7.3.2.6.8	Report change notification	839			X		
7.3.2.6.9	Read network device statistics	840			X		
7.3.2.6.10	Read network device identity using nickname	841			X		
7.3.2.6.11	Write network device's scheduling flags	842			X		
7.3.2.6.12	Read network device's scheduling flags	843			X		
7.3.2.6.13	Read network constraints	844			X		
7.3.2.6.14	Write network constraints	845			X		

^a VAR ID is Numeric Identifier in the service definition. It is also referred to as HART™ command number.

Table 376 – CP 9/2: Network management configuration service selection

Clause	Header	Var ID ^a	FD	Adap ter	Gate way	Hand held	Network manager
7.3.2.7.1	Disconnect device	960	X	X			X
7.3.2.7.2	Write network key	961	X	X			X
7.3.2.7.3	Write device nickname	962	X	X			X
7.3.2.7.4	Write session	963	X	X			X
7.3.2.7.5	Delete session	964	X	X			X
7.3.2.7.6	Write superframe	965	X	X			X
7.3.2.7.7	Delete superframe	966	X	X			X
7.3.2.7.8	Write link	967	X	X			X
7.3.2.7.9	Delete link	968	X	X			X
7.3.2.7.10	Write graph connection	969	X	X			X
7.3.2.7.11	Delete graph connection	970	X	X			X
7.3.2.7.12	Write neighbor attribute flag	971	X	X			X
7.3.2.7.13	Write network suspend	972	X	X			X
7.3.2.7.14	Write timetable	973	X	X			X
7.3.2.7.15	Write route	974	X	X			X
7.3.2.7.16	Delete route	975	X	X			X
7.3.2.7.17	Write source route	976	X	X			X
7.3.2.7.18	Delete source route	977	X	X			X

^a VAR ID is Numeric Identifier in the service definition. It is also referred to as HART™ command number.

9.3.5.2 AL protocol selection

9.3.5.2.1 AL protocol selection common to all device types

IEC 61158-6-20 applies. Table 377 specifies the AL service selection within Clause 8 for this profile.

Table 377 – CP 9/2: AL protocol selection

Clause	Header	Presence	Constraints
8.1	Abstract syntax	YES	—
8.2	Transfer syntax	—	—
8.2.1	Common APDU fields	—	—
8.2.1.1	Command number field	YES	—
8.2.1.2	Octet count field	YES	—
8.2.1.3	Response code field	YES	—
8.2.1.4	Application process status	YES	See Table 379
8.2.1.5	Extended status	YES	—
8.2.1.6	Value field	YES	—
8.2.1.7	Formats of APDU structures	YES	—
8.2.2	Common APDU structure	YES	—
8.2.3	Device application service-specific APDU structures	Partial	See Table 379
8.2.4	Layer management service specific APDU structures	Partial	See Table 380
8.2.5	Gateway and network manager specific APDU structures	Partial	See Table 381
8.2.6	Network management configuration specific APDU structures	Partial	See Table 382
8.2.7	Data coding rules	YES	—
8.3	Common procedures	YES	—
8.4	FAL Service Protocol Machine (FSPM)	YES	—
8.5	Application Relationship Protocol Machines (ARPMs)	YES	—
8.6	TL mapping protocol machine (TMPM)	YES	—

Table 378 – Application process status values

Bit mask	Definition
0x80	Device Malfunction – the device detected a serious error or failure that compromises device operation
0x40	Configuration changed – an operation was performed that changed the device's configuration
0x20	Cold Start – a power failure or Device Reset has occurred
0x10	More status available – more status information is available via another Read Command
0x08	Loop Current Fixed – the Loop Current is being held at a fixed value and is not responding to process variations
0x04	Loop Current Saturated – the Loop Current has reached its upper (or lower) endpoint limit and cannot increase (or decrease) any further
0x02	Non-Primary Variable Out of Limits – a Device Variable not mapped to the Primary Variable is beyond its operating limits
0x01	Primary Variable Out of Limits – the Primary Variable is beyond its operating limit

9.3.5.2.2 Variable ASE protocol selection

The service-specific APDU selection tables use following convention:

- X: the APDU is required;
- O: the APDU optional
- blank: the APDU is not required.

Table 379 – CP 9/2: Device application service-specific APDU selection

Clause	Header	CMD ID ^a	Field device	Adapter	Gate way	Hand held	Network manager
8.2.3.1	Identify	0, 11, 21	X	X			
8.2.3.2	Read primary variable	1	X	X			
8.2.3.3	Read loop current and percent of range	2	X	X			
8.2.3.4	Read dynamic variables and loop current	3	X	X			
8.2.3.5	Write loop configuration	6	X	X			
8.2.3.6	Read loop configuration	7	X	X			
8.2.3.7	Read dynamic variable families classifications	8	X	X			
8.2.3.8	Read device variables with status	9	X	X			
8.2.3.9	Read message	12	X	X			
8.2.3.10	Read tag, descriptor and date	13	X	X			
8.2.3.11	Read primary variable transducer information	14	X	X			
8.2.3.12	Read device information	15	X	X			
8.2.3.13	Read final assembly number	16	X	X			
8.2.3.14	Write message	17	X	X			
8.2.3.15	Write tag, descriptor and date	18	X	X			
8.2.3.16	Write final assembly number	19	X	X			
8.2.3.17	Read Long_tag	20	X	X			
8.2.3.18	Write Long_tag	22	X	X			
8.2.3.19	Reset configuration changed flag	38	X	X	X		
8.2.3.20	Perform self test	41	X	X	X		
8.2.3.21	Perform device reset	42	X	X	X		
8.2.3.22	Read additional device status	48	X	X	X		
8.2.3.23	Reset more status available						
8.2.3.24	Read device variable information	54	X	X			
8.2.3.25	Write device variable	79	X	X			
8.2.3.26	Read real-time clock	90	X	X			
8.2.3.27	Write publish data period	103	X	X	X		
8.2.3.28	Write publish data trigger	104	X	X	X		
8.2.3.29	Read publish data mode configuration	105	X	X	X		
8.2.3.30	Flush delayed responses	106	X	X	X		
8.2.3.31	Write publish data device variables	107	X	X	X		
8.2.3.32	Write publish data mode command number	108	X	X	X		
8.2.3.33	Write publish data mode control	109	X	X	X		
8.2.3.34	Read event notification summary	115	X	X	X		
8.2.3.35	Write event notification bit mask	116	X	X	X		
8.2.3.36	Write event notification timing	117	X	X	X		
8.2.3.37	Write event notification control	118	X	X	X		
8.2.3.38	Write event notification acknowledgement	119	X	X	X		

^a CMD ID is the value of Command field in the APDU. It is also referred to as HART™ command number.

Table 380 – CP 9/2: Layer management service-specific APDU selection

Clause	Header	Var ID ¹	Field device	Adapter	Gate way	Hand held	Network manager
8.2.4.1	Write Join_key	768	X	X		X	
8.2.4.2	Read join status	769	X	X		X	
8.2.4.3	Write active_search_shed_time	770	X	X		X	X
8.2.4.4	Write join mode configuration	771	X	X		X	
8.2.4.5	Read join mode configuration	772	X	X		X	X
8.2.4.6	Write network_ID	773	X	X	X	X	X
8.2.4.7	Read network_ID	774	X	X	X	X	X
8.2.4.8	Write Network_tag	775	X	X	X	X	
8.2.4.9	Read Network_tag	776	X	X	X	X	
8.2.4.10	Read wireless device capabilities	777	X	X			O
8.2.4.11	Read battery life	778	X	X	X		O
8.2.4.12	Report device health	779	X	X			X
8.2.4.13	Read neighbor health list	780	X	X			X
8.2.4.14	Read device nickname	781	X	X	X		
8.2.4.15	Read session list	782	X	X	X		
8.2.4.16	Read superframe list	783	X	X			
8.2.4.17	Read link list	784	X	X			
8.2.4.18	Read graph list	785	X	X			
8.2.4.19	Read neighbor attribute flag	786	X	X			
8.2.4.20	Read neighbor signal level	787	X	X			X
8.2.4.21	Report path down alarm	788	X	X			X
8.2.4.22	Report source route failed alarm	789	X	X			X
8.2.4.23	Report graph route failed alarm	790	X	X			X
8.2.4.24	Report transport layer failed alarm	791	X	X			X
8.2.4.25	Write UTC time mapping	793	X	X	O		
8.2.4.26	Read UTC time mapping	794	X	X	X		X
8.2.4.27	Write timer interval	795	X	X			X
8.2.4.28	Read timer interval	796	X	X			X
8.2.4.29	Write radio power output	797	X	X			X
8.2.4.30	Read radio power output	798	X	X	X		X
8.2.4.31	Request timetable connection	799	X	X			X
8.2.4.32	Read timetable list	800	X	X			
8.2.4.33	Delete timetable	801	X	X			X
8.2.4.34	Read route list	802	X	X			
8.2.4.35	Read source route	803	X	X			
8.2.4.36	Read CCA mode	804	X	X	X		
8.2.4.37	Write CCA mod	805	X	X			X
8.2.4.38	Read handheld superframe	806	X	X	X		X
8.2.4.39	Write handheld superframe	807	X	X		X	
8.2.4.40	Read packet hop-to-live	808	X	X	X		X
8.2.4.41	Write packet hop-to-live	809	X	X			O
8.2.4.42	Read Join_priority	810	X	X	X		
8.2.4.43	Write Join_priority	811	X	X			X
8.2.4.44	Read receive Priority_threshold	812	X	X	X		
8.2.4.45	Write receive Priority_threshold	813	X	X			X
8.2.4.46	Read device list	814	O	O	X		
8.2.4.47	Add device list entry	815	O	O	X		
8.2.4.48	Delete device list entry	816	O	O	X		
8.2.4.49	Read channel blacklist	817	X	X	X		
8.2.4.50	Write channel blacklist	818			X		
8.2.4.51	Read back-off exponent	819	X	X			
8.2.4.52	Write back-off exponent	820	X	X			X
8.2.4.53	Write network access mode	821			X		
8.2.4.54	Read network access mode	822			X		
8.2.4.55	Request session	823				X	X

NOTE CMD ID is the value of Command field in the APDU. It is also referred to as HART™ command number.

Table 381 – CP 9/2: Gateway and network manager service-specific APDU selection

Clause	Header	Var ID ^a	FD	Adap ter	Gate way	Hand held	Network manager
8.2.5.1	Read network device identity using Unique ID	832			X		
8.2.5.2	Read device's neighbor health	833			X		
8.2.5.3	Read network topology information	834			X		
8.2.5.4	Read publish data message list	835			X		
8.2.5.5	Flush cached responses for a device	836			X		
8.2.5.6	Write update notification bit mask for a device	837			X		
8.2.5.7	Read update notification bit mask for a device	838			X		
8.2.5.8	Report change notification	839			X		
8.2.5.9	Read network device statistics	840			X		
8.2.5.10	Read network device identity using nickname	841			X		
8.2.5.11	Write network device's scheduling flags	842			X		
8.2.5.12	Read network device's scheduling flags	843			X		
8.2.5.13	Read network constraints	844			X		
8.2.5.14	Write network constraints	845			X		

^a CMD ID is the value of Command field in the APDU. It is also referred to as HART™ command number.

Table 382 – CP 9/2: Network management configuration service-specific APDU selection

Clause	Header	Var ID ^a	FD	Adap ter	Gate way	Hand held	Network manager
8.2.6.1	Disconnect device	960	X	X			X
8.2.6.2	Write network key	961	X	X			X
8.2.6.3	Write device nickname	962	X	X			X
8.2.6.4	Write session	963	X	X			X
8.2.6.5	Delete session	964	X	X			X
8.2.6.6	Write superframe	965	X	X			X
8.2.6.7	Delete superframe	966	X	X			X
8.2.6.8	Write link	967	X	X			X
8.2.6.9	Delete link	968	X	X			X
8.2.6.10	Write graph connection	969	X	X			X
8.2.6.11	Delete graph connection	970	X	X			X
8.2.6.12	Write neighbor attribute flag	971	X	X			X
8.2.6.13	Write network suspend	972	X	X			X
8.2.6.14	Write timetable	973	X	X			X
8.2.6.15	Write route	974	X	X			X
8.2.6.16	Delete route	975	X	X			X
8.2.6.17	Write source route	976	X	X			X
8.2.6.18	Delete source route	977	X	X			X

^a CMD ID is the value of Command field in the APDU. It is also referred to as HART™ command number.

Annex A **(normative)**

Security

A.1 General

Security function for a network according to this standard specifies the communications security functionality and related security aspects. Other security aspects such as physical device security are not in the scope of this standard. The specific messages that are protected by the security functionality are:

- single-hop MAC sublayer protocol data units,
- end-to-end transport layer protocol data units, and
- management of keys within AL protocol data units.

Security should be policy driven and multi-layered, not subject to the choice of the device installer or software author. It shall be structured so that casual acts of a single individual that put system security at risk either are not permitted, or are strictly controlled. Security related services should be selected by policy coupled with individual application choice.

The policy should be distributed with a shared-secret symmetric key, permitting focused policy application that can be changed by a network security manager sharing a shared-secret key with the target device. Where policy permits more than one selection for a security attribute, the application should determine the selection within the permitted range via its initial or current service request.

The hop-by-hop transmission security is provided by the LLC sublayer. It defends against attackers who are outside the system and do not share system secrets.

The end to end security is provided by the security sublayer at the top of the network layer. It defends against attackers who may be on the network path between the source and the destination.

A new device is connected to the network only after it has received a Join_key via its maintenance port.

Devices with credentials participate in secured communications with other such devices through the use of a shared-secret symmetric key (see 6.5.3.2) that authenticates their messages to each other.

When only two devices share a secret key, receipt of an authenticated message by one of the devices shall imply either that it was sent by the other device or that the receiving device's own software has been compromised. Thus, for a pair of devices, message authentication implies the additional condition of source authentication i.e. that the claimed message source was the actual message sender.

NOTE Protection against compromise of the cryptographic boundaries inside the hardware of devices is outside the scope of this standard. Other publications, including ISO/IEC 15408 and the US NIST FIPS 140 series, address those vulnerabilities; compliance decisions are left to those that evaluate what implementers have built.

The primary security elements of the provided services include authorization of communications relationships between or among communicating entities, (see 6.5.5.4 - Quarantined state and request session command 823).

In both cases, a symmetric data key that is shared among intended communicants is used. Attackers that do not share the key cannot modify the message without being detected and cannot recover the encrypted higher-layer payload information (other than its size).

In general, a security policy associated with each key determines the type and amount of keyed protection to be provided when that key is used. Operation is based on a shared sense of time based on clock synchronization. The LLC sublayer and transport layer security services authenticate to the receiver the time of PDU transmission and the approximate time of TPDU creation. The protocol elements are specified in the appropriate layers.

See 6.2.4 Error detection and security for MAC sublayer security protocol.

See 6.5.3 Security function for end to end security protocol.

See A.2 for joining protocol.

See A.3 for changing key operation.

See A.4 for security management.

See A.5 for maintenance port.

See C.2 for the key distribution function performed by the network manager.

The following APDU are used for managing the security objects:

Commands 768 (see 8.2.4.1), 782 (see 8.2.4.15), 823 (see 8.2.4.55), 961 (see 8.2.6.2), 963 (see 8.2.6.4) and 964 (see 8.2.6.5).

A.2 Joining

A.2.1 The join sequence

A.2.1.1 Overview

"Joining" refers to the process used by a device to obtain access to the network and to become integrated into it. An overview of the sequence is shown in Figure A.1. The steps required to join a new device into the operational network are listed below.

- The device shall be provisioned by a maintenance tool. It consists of writing the network_ID and the Join_key in the new device.

NOTE The Join_key can be different for every device.

- The new device monitors the network and receives the Advertise DLPDUs transmitted by other devices. The Advertise DLPDU includes sufficient information for the new device to synchronize to the network clock and communicate on the correct link i.e. the correct slot number and channel. The new device locates the network with the matching network_ID and synchronizes to it.
- It then establishes a secure channel between it and the network manager using the Join_key to encipher the initial communications between the two. It sends its credentials to the network manager to demonstrate it is trustworthy.
- The network manager verifies the trustworthiness of the new device.
 - The device's Identity according to CMD ID #0 and Long_tag are presented to the network manager.
 - The Join_key works like a password.

- The network manager considers all of these three items when making a decision to allow the device into the network.
- After the device is deemed trustworthy, the network manager provides the network manager session key and the network key to the joining device.
- After security requirements for new devices have been met, the network manager proceeds to integrate the device into the network. This is accomplished by provisioning the device with normal superframes and links.
- The network manager may choose to leave the device quarantined. In this case the device can participate in the network but does not have a gateway session.

IECNORM.COM: Click to view the full PDF of IEC 62591:2010

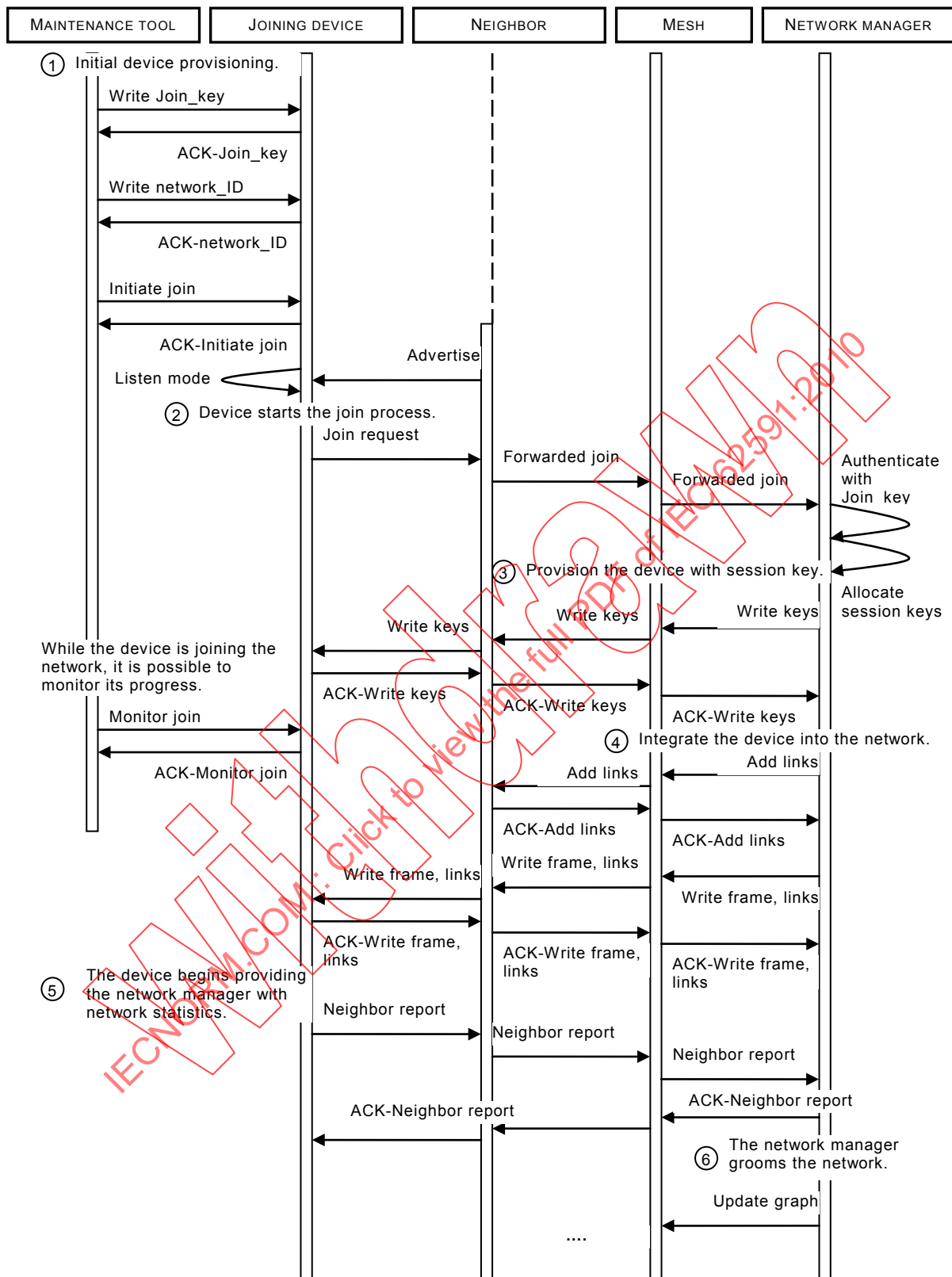


Figure A.1 – Join sequence

- After the quarantined device obtains a session with the gateway, it becomes operational. It then begins acquiring the bandwidth and communication resources required to publish process data and events as dictated by its configuration.

A.2.1.2 Initial Device Provisioning

Prior to attempting to join the network the device requires two pieces of data – the Join_key and the network_ID. The network_ID identifies the network the device is to join and the Join_key is the network password that will allow the device to join the network. These two items should be written to the device via the device's maintenance port (see Annex A.5).

After initial provisioning is complete, the device can be immediately configured to join the network. Alternatively, the device can be mounted in the process and then the end user can use a device maintenance tool to place the device into join mode.

The maintenance tool may also be used to monitor the join process allowing the operator to intervene, if necessary.

A.2.1.3 Listening for Network Traffic

After the join has been initiated, the device's DLE is placed into search mode to synchronize to the network and receive Advertise DLPDU. In this mode, the device shall identify neighbors (advertising or not) and gather neighbor statistics for example average signal level. After identifying one or more advertising neighbors, the joining device shall select one of them as parent for its joining process. The neighbor with the lowest joining priority and adequate RSL shall be selected as parent neighbor.

A.2.1.4 Presenting Credentials

After the device has synchronized to the network, it transmits a join request (see ② in Figure A.1) to the network manager. The join request and its encipherment contain the device's credentials for the network manager to inspect prior to allowing the device to join the network. The device's objective is to obtain a session key a network key, so that it can continue to communicate with the network manager. The network key is used for data-link layer device-to-device, one hop security and authentication. There are three key credentials:

- The device's identity (i.e. Command 0 response),
- Its Long_tag (Command 20 response); and
- The device's Join_key.

The first two are contained in the join request's payload. The Join_key is used as the Session_key for all communications until the device receives session and network keys from the network manager. Therefore, successful authentication by the network manager confirms the device has the correct Join_key.

Since the joining device does not know the network key, it shall use the well-known key for communicating the DLPDUs with the advertising neighbor, which is used as prospective parent. This prospective parent receives the DLPDU from the joining device and forwards it through the mesh using the network key.

After sending the join request, the joining device shall start a response timer to monitor the response. If this timer expires, then the device shall send another join request to the next available advertising neighbor. Join requests are sent until max_join_retries is exceeded. See Table 124.

A.2.1.5 Getting the First Keys

When the network manager receives a join request it shall verify that the request was from a trusted device. A trusted device:

- has a trusted identity (i.e., the right device name),
- is using an appropriate Join_key, and

- properly combines the device name and Join_key with each other.

The join request has sufficient information to satisfy all of these criteria. Identity is established using the binary information in the command 0 or using the Long_tag in command 20.

NOTE There are a wide range of methodologies and security techniques to establish whether a joining device is to be trusted. Selection of the strategy employed by the network manager to verify that the join device is a trusted device is beyond the scope of this specification.

If the device's credentials meet the verification criteria, then the join request is considered as authentic. The network manager shall allocate session key, the device's nickname and write these two and the network key into the joining device (see ③ in Figure A.1). The join response PDU is enciphered using the Join_key. The NPDU containing the keys shall be routed to the joining device via its prospective parent.

The device shall acknowledge the reception of join response. The acknowledgement PDU shall use the just received session and network keys.

A.2.1.6 Device Integration into the Network

Since, the device has session key and the network key, it is able to communicate with the network manager. At this point in the Join sequence, its communication requires the network manager to use proxy routing to reach the joining device. The network manager shall integrate the joining device more tightly into the network. This includes:

- providing the device with the identity of at least two time-source parents,
- updating the communication tables in the device's parents, and
- transferring the device's communication from join links to normal links.

Examples of the associated communication activity are shown starting at ④ in Figure A.1. This communication activity will continue for some time and result in the network manager writing a series of frames and links to the joining device. After this, the device is integrated into the network and shall stop using the join links for communication.

The Network Manager shall maintain all join links related to a device while that device is in the process of joining.

A.2.1.7 Quarantine

If a device has joined the network, but it does not have any session with the gateway, the device shall be in "Quarantined" state. In this state, the device can only communicate with the network manager and shall not publish any process data. The device shall otherwise operate normally, conveying DLPDUs received from neighbors, generating neighbor reports and network statistics. The network manager can modify the network communication configuration as needed to groom the network.

Upon entering "Quarantined" state, the HealthReport timer shall be initialized to Health_report_time (see Table 58) and started. The device shall begin generating health reports (see ⑤ in Figure A.1). It generates health reports whenever the HealthReport timer expires. Health reports shall be transmitted at the "Process-data" priority level. Following every health report, the HealthReport timer shall be reset to Health_report_time.

After joining the network, the first health report shall be generated after identifying at least three neighbors or when the HealthReport timer expires, whichever occurs first. If multiple Advertise packets were captured, the first report shall be generated immediately after entering the Quarantine state.

Health reports consist of the response APDUs for Command 779 (see 8.2.4.12), 780 (see 8.2.4.13) and 787 (see 8.2.4.21). Using Command 780, statistics on all linked neighbors are

returned. Command 787 reports on all detected neighbors that do not have links to the device. Depending on the network and number of devices, multiple packets are normally generated. Health reports are aggregated into as few TPDUs as possible.

Depending on the network manager's security strategy, it may immediately write the gateway session to the quarantined device or leave the device quarantined for a time. In any case, the device shall remain quarantined until it receives a session allowing communication with the gateway.

A.2.1.8 Becoming operational

When a new gateway session is written to the device it becomes operational. Once this session is operational, a gateway will normally begin filling the gateway's data cache for the device. This will cause a surge in request/response traffic to the device and it may be several hundred transactions. This communication uses the maintenance service. The maintenance service is owned and controlled by the gateway.

NOTE 1 Upon receiving a session to a new device, the gateway should request significant bandwidth (for example one request/response per second) to the device while it fills its cache then lower it to a normal rate (for example one request/response per 10 s).

NOTE 2 In addition, external host applications (for example instrument management packages, process automation controllers, etc.) may begin communicating with the device through the gateway. The gateway should increase and decrease the maintenance service bandwidth guaranteeing responsive communications with the field device. This communications rate should be similar to that found in wired HART™ communication network.

After the device receives its gateway session, it shall obtain enough bandwidth and communication resources to meet its responsibilities to the process automation system as a whole. After receiving begins requested bandwidth for the process data (the publish service) and alarms (the event service), the device shall start sending publishing reports.

All maintenance traffic is request to the field device and response from the field device. The field device shall assume the gateway has allocated sufficient communication bandwidth. Therefore, responses to command requests generated by the gateway and the network manager shall be immediate and not associated with any Timetable in the device.

A.2.2 The network layer join process

A.2.2.1 General

The NLE state machine as shown in Figure A.2 enforces the high-level join procedure (including retries for security purposes) while the DLE state machine is focused on synchronizing the device to the communication slot times. The primary objective of the join process at the network layer is to receive admission to the network and obtain a frame with links allowing the device to communicate reliably with the network manager.

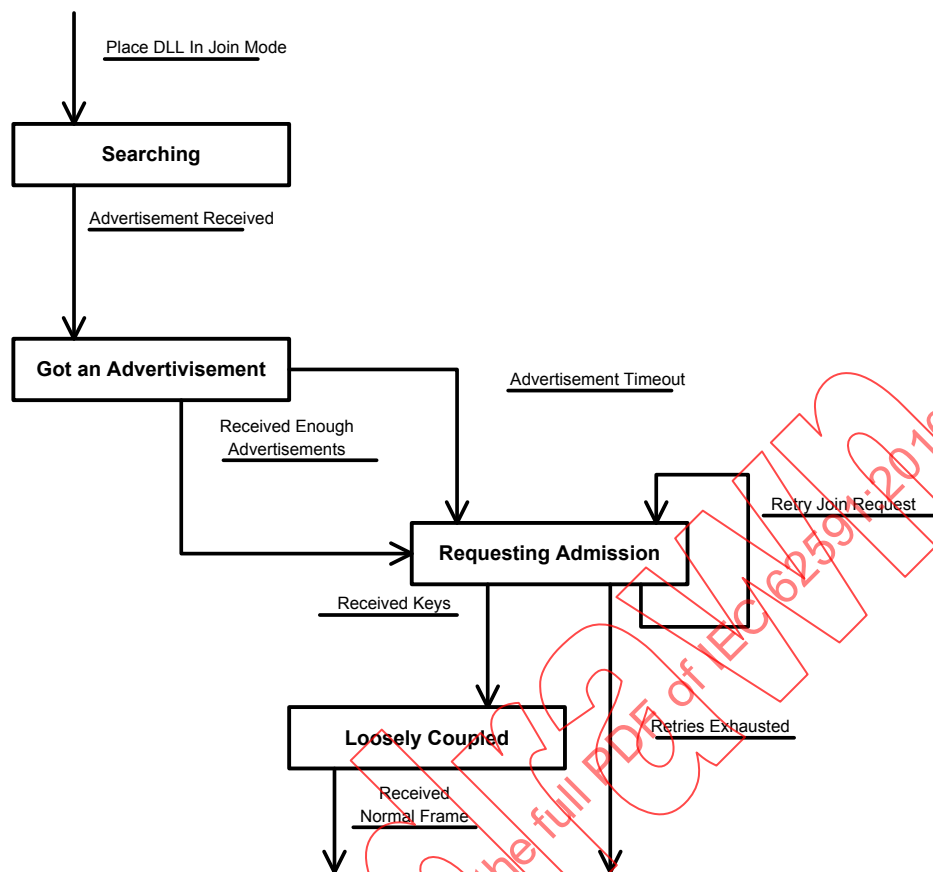


Figure A.2 – Network layer join procedure

A.2.2.2 Searching

While in the searching mode the NLE waits for reception of an Advertise packet. While in this mode the DLE is searching for the network, synchronizing to it and receiving DLPDUs.

After the Advertise indication primitive is received from the DLE, the NLE sequences to the "Got an Advertising Neighbor" state.

A.2.2.3 Got an advertising neighbor

Upon entering the "Got an Advertising Neighbor" the AdWaitTimer shall be initialized to Ad_wait_timeout (see Table 58) and started. The device shall continue to receive additional Advertise packets. If at least three different Advertise packets have been received or the AdWaitTimer expires, the device shall change to the "Requesting Admission" state.

A.2.2.4 Requesting Admission

Upon entry into the "Requesting Admission" state, the device shall send a join request to the network manager, initialize the JoinRspTimer to Join_rsp_time (see Table 58), and start the JoinRspTimer. If JoinRspTimer expires, the device shall:

- a join request to the network manager,
- start the JoinRspTimer, and
- decrement the JoinRetry counter.

If the network manager responds by writing the network key, the network manager session and the device's nickname, the device shall change to the "Loosely Coupled" state.

Else, if the JoinRetry counter becomes zero, the state machine shall exit with an error.

A.2.2.5 Loosely Coupled

After the device has a nickname and the network key, it is able to communicate with the network manager. However, the device's connection to the network is tenuous at best (it only can talk via the shared join links). The device shall stay in the "Loosely Coupled" state until it receives a normal frame and links. If the device gets such a frame and links, then it shall exit the network layer join process successfully. Reliable communications between the network manager and the device is now ensured.

A.2.3 The data-link layer join process

A.2.3.1 General

The DLE is triggered to enter the network search mode by the NLE. The objective of this mode is to synchronize the device's slot timing to that of the network. The first step in synchronizing is to begin actively searching for the network. Figure A.3 shows the state diagram of the DL network search procedure.

The join process is a device-level process and requires close coordination between the network and data-link layers.

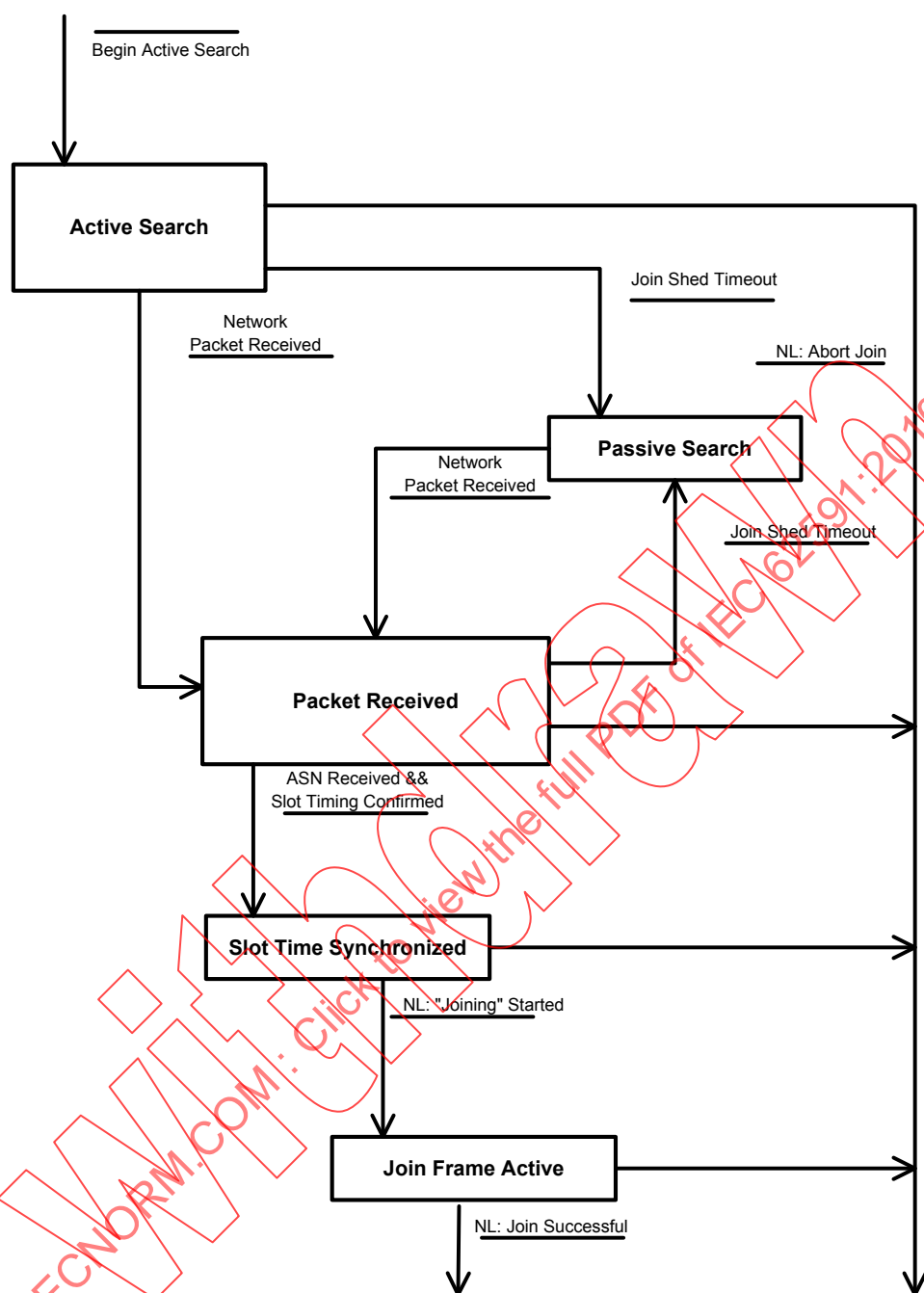


Figure A.3 – Data-link layer join procedure

A.2.3.2 Active search

While in the "Active search" state, the device shall leave its transceiver in receive mode while continuously listening for packets. When the DLE enters this state, it shall set a timer to Active_search_shed_time to limit the network active search time. If this timer expires without identifying the network, the device shall transition to the "Passive search" state.

The device's transceiver shall remain on a channel for the interval indicated by the ChannelSearchTime value. When that time period lapses, the device shall switch to the next channel and resume listening. Channels are scanned sequentially until Active_search_shed_time lapses or a packet is received.

NOTE The default value of ChannelSearchTime is 400. An implementation can select a different value depending upon its need to conserve power.

When a packet is received and its Network_ID matches the network_ID set in the device, the device shall transition to the "Packet received" state.

A.2.3.3 Passive search

The "Passive search" state is a reduced cycle, power saving mode of operation entered when the network cannot be located. In this mode, the device continues to look for network packet, but at a slower rate. The device shall wake up and listen for the interval indicated by the PassiveWakeTime. If no packet is received, the device shall turn off its transceiver and returns to low-power mode. This cycle shall repeat as specified by the PassiveCycleTime.

NOTE The default value of PassiveWakeTime is 6,5 s; the default value of PassiveCycleTime is 600 s. An implementation can select a different value depending upon its need to conserve power.

If a packet is received, the device shall transition to the "Packet received" state. The device can also be forced back into the "Active search" state by the reception of the "Force join mode" command.

A.2.3.4 Packet received

The objective of this state is to synchronize the device's slot timer to the network. The device shall receive DLPDUs. If the DLPDU is not an ACK, then the start time of that packet shall be recorded and the device's slot time is established. Subsequently, the start time of all additional non-ACK packets shall be compared to the device's slot timing and statistics shall be compiled measuring the device's synchronization to the network slot time. The device is considered synchronized when the slot timing statistics converge.

While in this state the device shall continue to receive DLPDUs and update its neighbor table. In addition, channels shall be changed as indicated by the ChannelSearchTime value. If an Advertise DLPDU is received, the channels searched shall be limited to those indicated in the Advertise packet's channel map.

This device shall transition to the "Slot time synchronized" state when

- the slot time is synchronized, and
- the Absolute Slot Time has been aligned to the network.

A.2.3.5 Slot time synchronized

In this state the DLE can reduce its listening to slot times. As DLPDUs are received, the device's timers shall be updated keeping it in sync with the network. The DLE shall wait for to receive another Advertise DLPDU. When an Advertise DLPDU is received, the DLE shall provide Advertise indication primitive to the NLE. The device shall configure the Join frame(s) and links as indicated in the Advertise DLPDU it receives. These frames remain disabled until join requests are generated. In addition, the graph indicated in the Advertise packet is initialized and the connection to the advertising device created.

In this state the DLE shall wait for the NLE to signal that it has started the join process by sending join request. When a join request is received from the NLE, the device shall enable the join frame(s) and transition to the "Join frame active" state.

A.2.3.6 Join frame active

The first join attempt shall be made via the neighbor demonstrating adequate RSL and indicating the lowest Join_priority. After the join request is transmitted, the device shall begin transmitting Keep-Alive DLPDUs, as needed, to maintain synchronization with Neighbors connected via join links. The device shall stay in this state until the NLE signals that the join was successful or the join is aborted.

Before the join request is generated, the join frames received via the Advertise DLPDUs are enabled. Joined frames contain shared slots. Therefore, collisions with other joining devices are probable. To minimize collisions resulting from many devices trying to join simultaneously, the Backoff_exp shall be initialized to four prior to transmitting the first join request and the Back-Off Counter (Backoff_cntr) calculated. This results in the randomized slot being used for transmitting the join request.

A.3 Network key change operation

The Key change operation is shown in Figure A.4. The sequence begins with the Network manager (the master in this example) generating an acknowledged Broadcast-request (see 8.2.6.2). The Transport entity sends it to all devices (see ①).

Each device, in turn, sends the response to the command (see ②). The device response is enciphered using the unicast device-Network Manager session. This ensures intervening devices cannot spoof the response, which contains the Key. However, to allow the Network manager to correlate the response the transport type is set to Response-broadcast.

The Network manager's Transport entity receives acknowledges from all devices (retrying as needed). At the stipulated Absolute Slot Number the Network Key in all devices is changed to the new value (see ③).

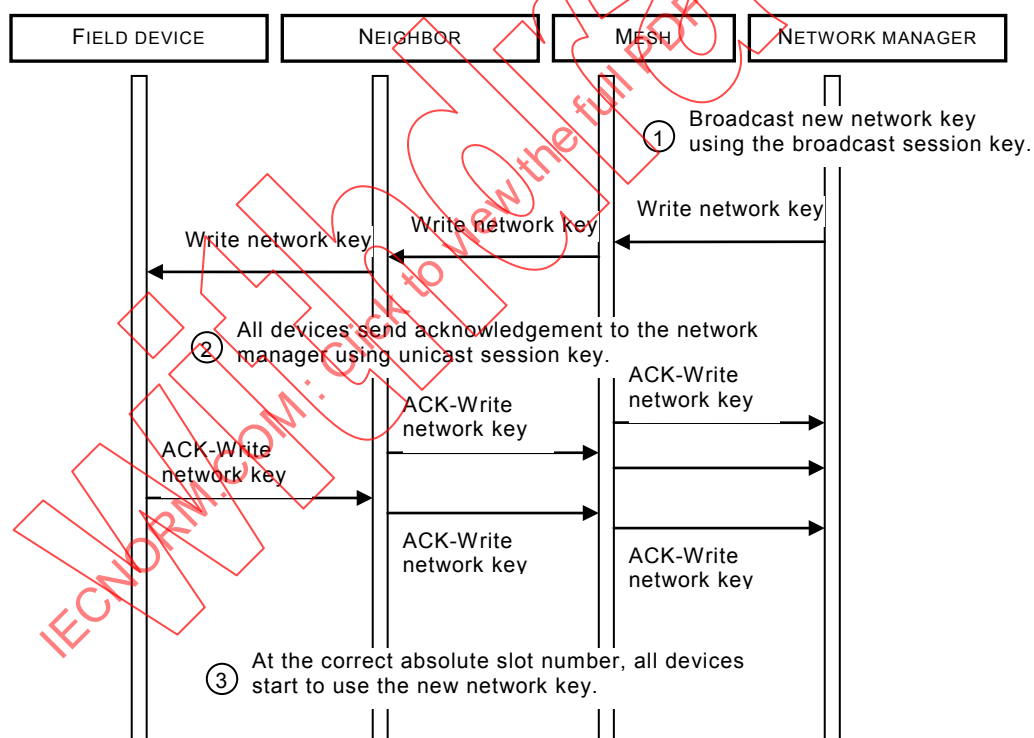


Figure A.4 – Network key change operation

A.4 Security management

The security manager generates and manages the cryptographic material used by the network. It is responsible for the generation, storage, and management of keys. The security management is a function that is separate from the network management function. The security manager may be a centralized function in some plant automation networks, servicing more than one wireless network and in some cases other networks and applications. There is one security manager associated with each wireless network. One security manager may service multiple wireless networks.

The security manager works with the network manager to secure the wireless network from adversarial threats to its operation. A secure connection between the network manager and the security manager is required. This secure connection is beyond the scope of this specification.

A.5 Maintenance port

All field devices shall provide a maintenance port that complies with the requirements of this standard. The maintenance port interface is used for provisioning (for example to load the Join_key and network_ID or to monitor the join process). The device shall allow reading of the device parameters and writing of parameters required for provisioning via the maintenance port. Any write request that can only be sent by the network manager shall not be available via the maintenance port; the device shall respond to such request with "Access Restricted". Any tools connected to the maintenance port shall not have access to the wireless network.

The maintenance port can be either a standard wired HART™ interface designed for connection to the process automation system according to IEC 61784-1 CP 9/1 or a dedicated maintenance port. If it is a dedicated maintenance port, the port shall comply with following requirements.

- Through the maintenance port, the field device shall appear to be a wired HART™ multi-drop slave device. The field device shall be configured for polling address 1.
- The input impedance shall be 500 ohms simplifying the direct connection of legacy maintenance tools and applications supporting the wired HART™ communication protocol.
- The maintenance port shall not support publish data communication.
- The maintenance port shall not be permanently wired to the process automation system.
- The maintenance port shall be clearly marked as the maintenance port.

Annex B (normative)

Wireless procedures

B.1 Initializing a wireless network

Prior to forming the network, the network manager shall be provisioned with the network_ID. Using this ID and a supply of security keys network formation can be initiated. This begins with the network manager creating a secure and private connection with the gateway. As part of its initialization sequence, the network manager shall download following to each access point of the gateway:

- the network management superframe supporting base bandwidth required to monitor and service the network,
- the network graph supporting upstream traffic to the network manager,
- the Join superframe and join links allowing new devices to join the network, and
- dedicated and shared links (both transmit and receive) supporting management of devices, the transport of health reports, and communication of alarms (for example path down).

The access points shall be configured to be active in every slot. This will maximize the network capacity available for advertising, network management and join packets. The network manager shall enable the first superframe, which starts the ASN zero (i.e. the network is born). The access point(s) shall begin transmitting Advertise DLPDU, so that other devices can join the network and the network begins to form.

There are three components of network formation: advertising, joining, and parameter negotiation. As part of advertising, network devices that are already part of the network send DLPDU announcing their presence to the network. Advertise DLPDU includes the network_ID, ASN, join frames and join links. Devices that are trying to join the network listen for these DLPDUs. The new device attempts to join the network as specified in A.2.

When a device joins the network, this device and the gateway both request allocation of network capacity from the network manager. The device requests the network capacity to publish process data and the gateway requests network capacity to support request and response traffic to this new device. The network manager uses these service requests to estimate and manage the available network capacity. If there is sufficient network capacity, the newly joined device shall be allocated superframes and links according to the requests. If network capacity becomes constrained, then the network manager may reduce the network capacity requested or refuse to allocate any network capacity to the requesting device.

B.2 Neighbor discovery

The neighbor discovery process is used by a device to learn the potential connectivity to new devices. The device maintains a list of discovered devices in its Neighbor table following the neighbors with links to the device. Network devices shall periodically report neighbor information in their health reports using Commands 780 (8.2.4.13) and 787 (8.2.4.20). The network manager shall use the information in the health report to adjust the overall network graph and in some cases, adjust the network schedule. This overall sequence is described in Figure B.1.

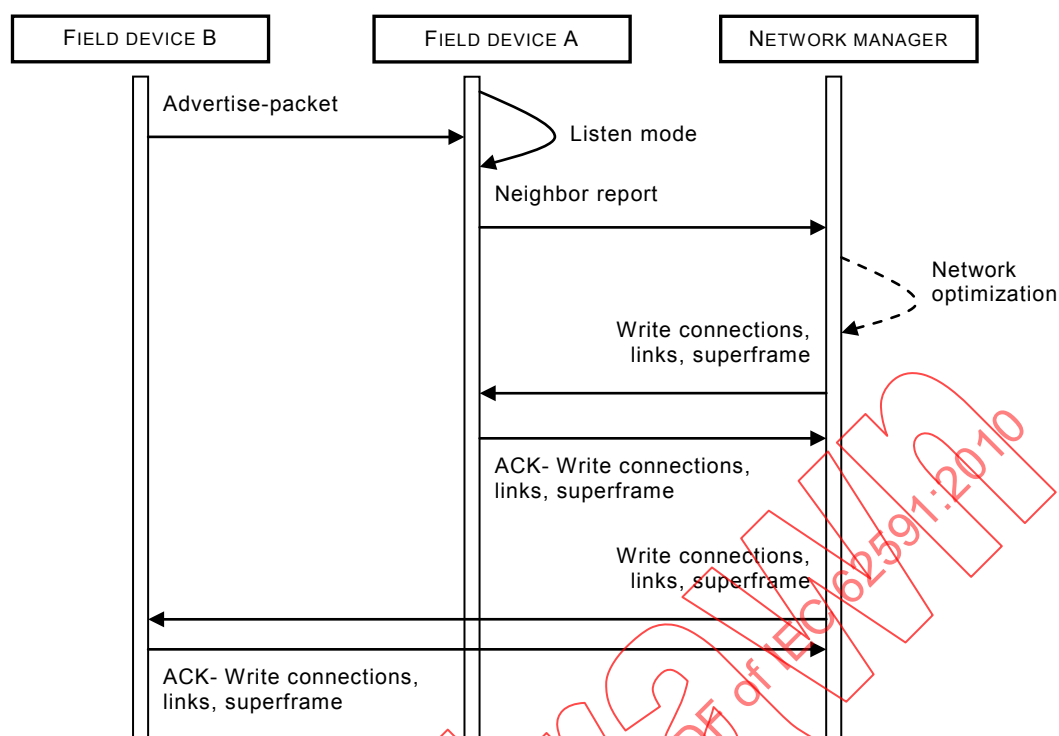


Figure B.1 – Neighbor discovery

B.3 Path failure

When a network device loses connectivity to a neighbor, it shall report the path failure to the network manager. Figure B.2 depicts three devices and shows that device A and device B have been successfully communicating (for example device A published via device B). However, interference or some blockage disrupts communication between device A and device B. The mesh allows device A to continue responding via an alternate route (for example via device C). Since communication was lost, device A and device B shall begin transmitting Keep-Alive DLPDUs to probe the connection. Communications continues to be broken and after the Path_fail_time interval expires, a path-down alarm (command 788 – see 8.2.4.21) shall be generated by both non-communicating devices.

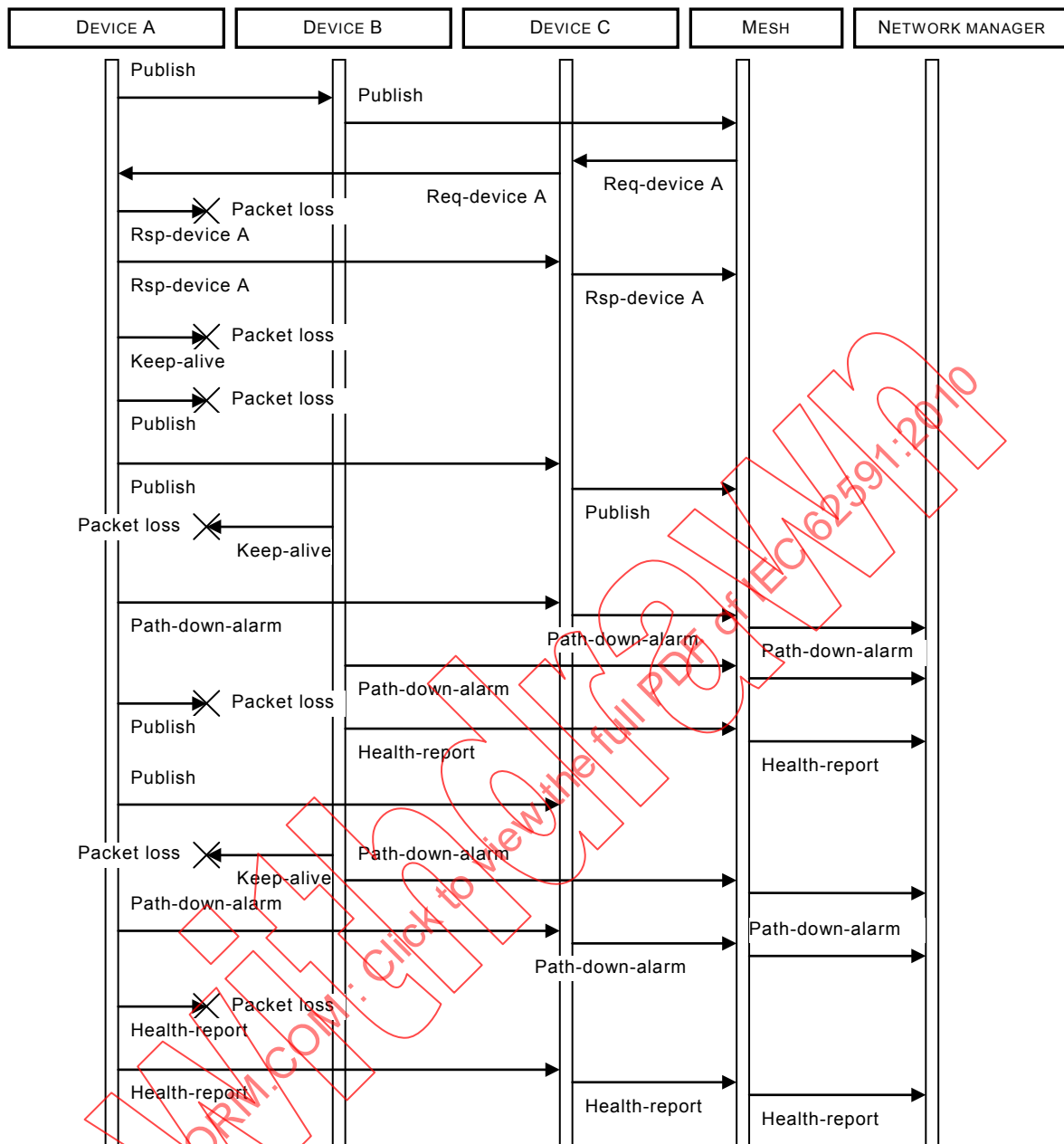


Figure B.2 – Path failure

B.4 Device leaving the network

In the network operation there will be times when a device is either suspended or disconnected from the network. When this occurs, the device shall go through a complete re-join sequence. Prior to departing the network the device shall not accept any additional DLPDU, discard the DLPDUs waiting for transmission and shall send a disconnect DLPDU to all of its neighbors as shown in Figure B.3.

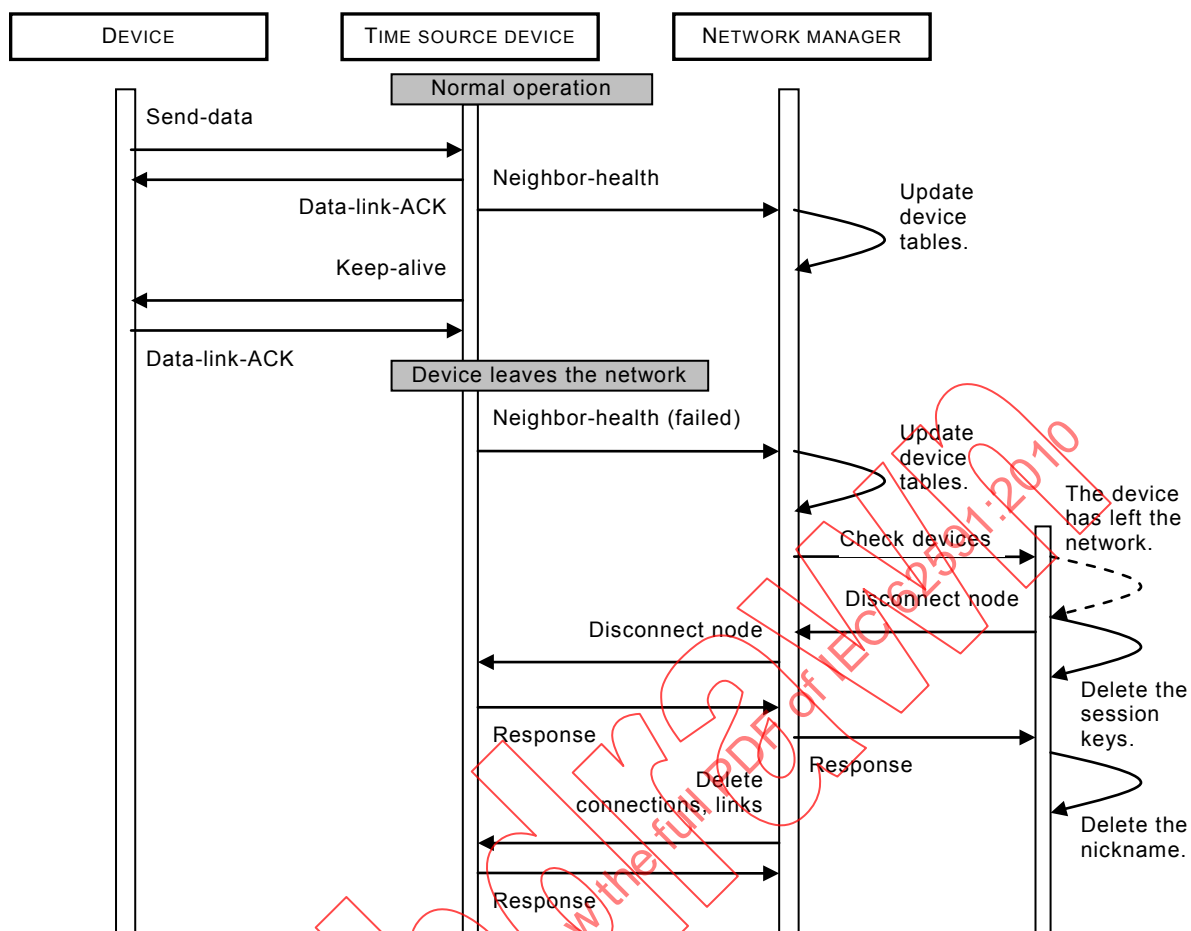


Figure B.3 – Device leaving the network

Annex C (normative)

Network management

C.1 Overview

The network manager is responsible for the overall management, scheduling, and optimization of the wireless network. As part of its tasks the network manager initializes and maintains network communication parameter values. The network manager provides mechanisms for devices to join and leave the network. It is also responsible for managing dedicated and shared network resources.

The network manager and the gateway shall be a single logical entity. The relationship of the network manager to the rest of the wireless network is illustrated in Figure C.1. The network manager may be co-located with the gateway in the same physical box or located in a completely separate physical box.

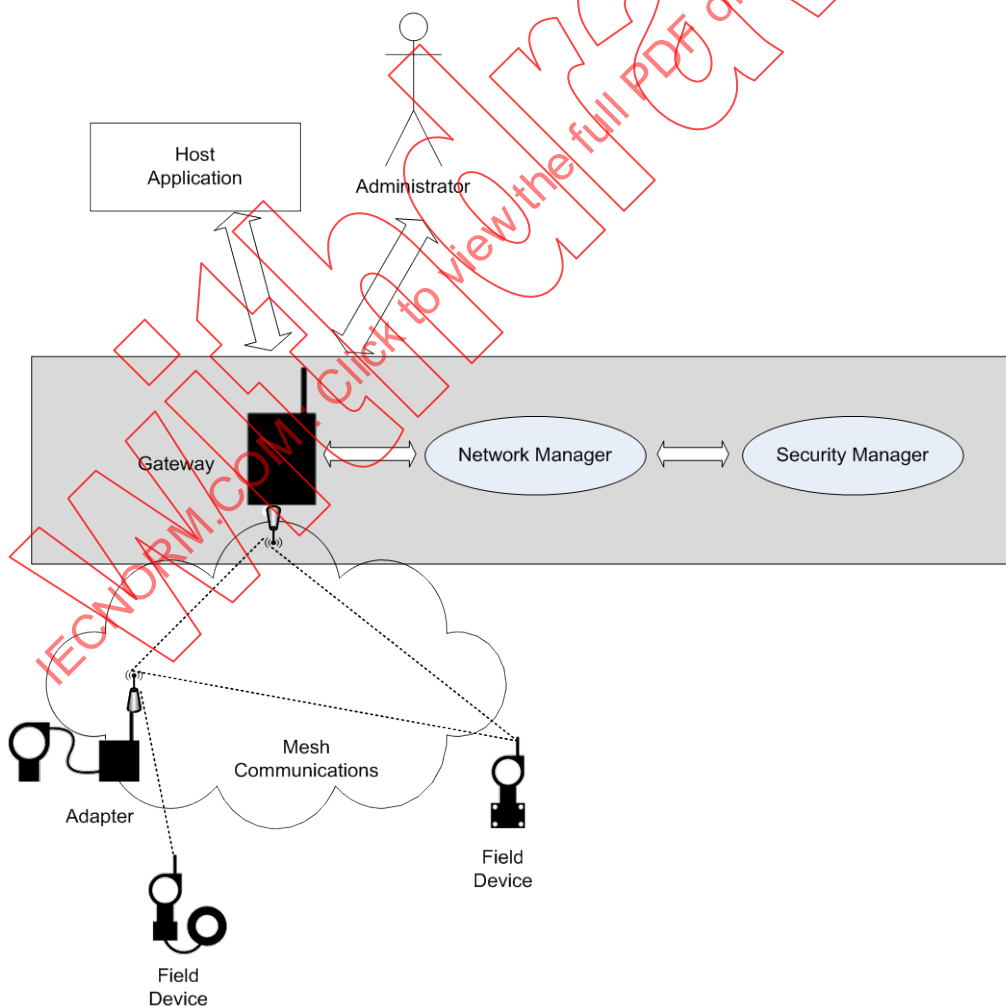


Figure C.1 – Network manager in wireless network

Although the network manager need not have a direct physical layer connection, it can communicate through the gateway that has the access point(s) to the network. The network

manager shall have a fixed well known address (unique ID = 0xF980000001; nickname = 0xF980), which is used as destination for the APDUs from field device to the network manager. It is possible to have redundant network managers connected to one network, but there shall be one and only one active network manager per instance of a wireless network.

NOTE One network manager can be used for more than one network. But, the devices in one network cannot communicate with the devices in the other network connected to the same network manger.

The Network Manager communicates with devices on the wireless network through the FAL. It also uses application layer PDU (see 8.2.6) to write and read the layer management objects in each device. The network manager is also responsible for collecting and maintaining diagnostics about the overall health of the network. These diagnostics are available to be reported to host-based applications. The diagnostics are also used to adapt the overall network to changing conditions.

For the network manager to perform its complete set of functions, it needs information about the devices, information about how the network is to be used, and feedback from the network on how well the network is performing. Configuration and setup information about devices is read from the devices. Communication resources are requested by devices, applications, and plant users. Feedback on how well the network is performing is provided by the devices through health reports and diagnostics.

The interface between a gateway and the network manager is outside the scope of this standard. The network manager and the gateway are responsible for establishing a secure connection with each other, and maintaining this connection to carry control and data traffic. Thus, it is not necessary for the gateway to go through the normal network device join process. After the gateway connects to the network manager, the network manager may configure the gateway to begin advertising to other devices.

C.2 Network manager functions

The network manager shall perform the functions summarized in Table C.1.

Table C.1 – Network manager requirements

Network function	Requirement
Network formation and configuration	Provide logic for initializing itself and starting up the network
	Manage topology; adapt the network to changes as diagnostic information is reported from devices
	Manage distribution of the network key that is provided by the security manager to all network devices. Separate network manager and gateway keys shall be used for unicast and broadcast traffic that originates from each of them.
	Manage join process. The network manager shall validate the device that tries to join the network. After authenticating a network device, the network manager shall give the joining network device the network key and four session keys: 1- network manager unicast session keys, 2 - network manager broadcast session key, 3 - gateway unicast session key, and 4 - gateway broadcast session key. Device shall be configured with network_ID before it attempts to join the network, so that it can find the right network to join.
	Assign 16-bit nickname. The network manager shall assign and manage network unique 16-bit nickname (network addresses) to each network device. The network manager is responsible for ensuring that the Neighbor table inside each device is up-to-date.
	Establish a connection with the gateway. Whenever the gateway receives a message destined for the network manager gateway forwards it to the network manager. The network manager shall be connected to the network via the gateway.
	Configure at least one of the network access points connected to the gateway to provide the network clock.
	Manage network configuration. Maintain a full map of the network configuration, including any information about the network that has been distributed to network devices.
	Respond to requests for network information. For example when a host application makes a network management information request for a device in the network, the network manager is responsible for providing the response.
Routing	Create and manage network route. The network route is a complete map of the network.
	Manage neighbor tables. The network manager shall collect network statistics and neighbor table information from each device through periodic health reports. This information shall be used to adapt the network to changes.
	Build route tables for graph routing. Graph routing is ideal for both scheduled upstream and downstream communications. Upstream communications include process measurements and alarms. Downstream communications include set point changes to actuators.
	Build source route lists for source routing.
	Allocate communication resources to the network devices, gateway and to itself so that it has the network capacity allocated to manage the network and the devices have sufficient network capacity to communicate.
Network Schedule	Create superframes. Multiple superframes are used to support communications at specific scan rates. Additional superframes shall be allocated to support device management and diagnostic applications which require large amounts of traffic for short periods of time.
	Assign links in Superframes.
	Create link tables. Each link includes exactly one slot associated with a superframe, its type (normal, advertising, discovery), its options (transmit, receive, shared), neighbor information, channel offset, and the device connected to this link.
	Activate and deactivate superframes in response to application demands.

Network function	Requirement
Channel Management	Keep track of blacklisted channels. NOTE The assigning of a channel to the blacklist is a manual operation.
	Provide channel offset. The channel offset is used to calculate the channel number when channel hopping. The channel offset takes on a value of 0 to (number of channels less number of blacklisted channels).
Network Diagnostics and Adapting	Maintain overall wireless network diagnostic information. If a network device has not received a packet from one of its neighbors within the Keep_alive_time, the device sends a path-down notification to the network manager indicating that the path is no longer available.
	Maintain record of health information about each network device.
	Allocate communication resources as requested by network devices. Devices request network capacity so that it can support publish data operation, event notification, and block mode traffic. Gateways request network capacity to support its client demands. Network traffic is biased towards a particular path by increasing or decreasing the number of links through a particular device.
	Optimize routes and schedules in order to improve operation of the network while conserving power within devices.

Annex D (normative)

Gateway

D.1 Overview

Gateway device connects the wireless network to a plant automation network, allowing data to flow between the two networks. It provides host applications access to the network devices. A gateway device can be used to convert from one protocol to another, as go-between two or more networks that use the same protocol, or to convert commands and data from one format to another. The wireless network also uses the gateway as the source for the synchronized clock used by the timeslots and superframes.

The gateway provides a sink or source point for wireless network traffic. The gateway also provides a connection to the network manager. The gateway shall have fixed, well known address (unique ID = 0xF981000002; nickname = 0xF981). There shall be one and only one gateway per network.

The gateway uses application layer services (see 8.2.5) to communicate with network devices. The gateway also works as a server responsible for collecting and maintaining cached data and command responses from all devices in the network. These cached responses correspond to publish data messages, event notifications and command responses. These cached responses are returned immediately to host application requests. This reduces network communication load improving power utilization and host application responsiveness.

D.2 Access point

The network access point provides the physical connection into the wireless network. The gateway can use more than one network access point to improve the effective throughput and reliability of the network.

NOTE 1 The physical location of the access point is selected based on the network topology and the signal strength. This is generally different than the location of the gateway, which is based on the location of the automation network. If there is more than one access point, then these are located at some distance to avoid RF interference.

The gateway resides at the root of all graphs allowing PDUs to be routed to the most convenient network access point. The access point shall work as a network layer router. It shall have its own unique ID and network address, which is different than the unique ID and network address of the gateway. The nickname for the network access point is assigned by the network manager. The network manager is also responsible for configuring the routes in the access point. When a gateway uses multiple network access points, the network manager provides redundant routing for each network device using at least of two of the gateway's network access points. The gateway shall use the routing table (configured by the network manager) to select one of several access points as the next hop for sending the NPDU. If a network access point fails, the network traffic will be impaired but communication will continue.

NOTE 2 The communication between the gateway and the access point is outside the scope of this standard.

Network access points shall propagate the clock to the network. There is always at least one network access point providing the clock to the network. If there are several network access points providing the clock, the gateway shall ensure that they stay synchronized with each other. This leads to one of three possible configurations:

- a single network access point (see Figure D.1),
- multiple network access points, each providing the network clock (Figure D.2), or
- multiple network access points with at least one not providing the network clock (Figure D.3).

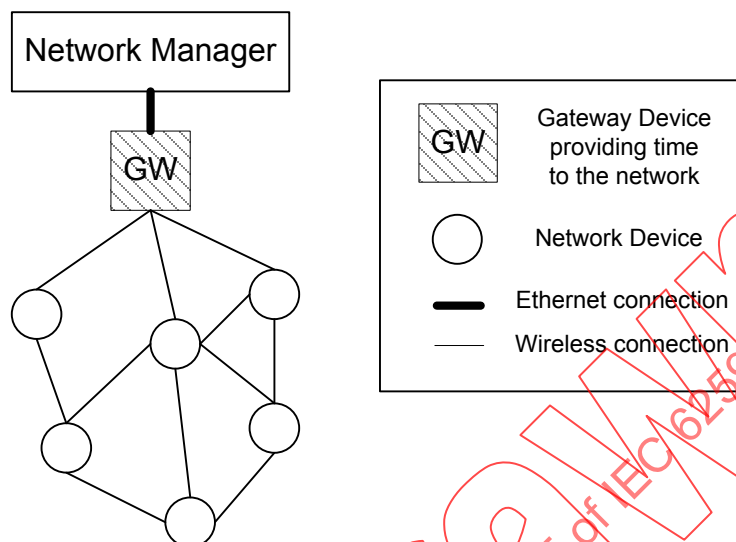


Figure D.1 – Single network access point with clock

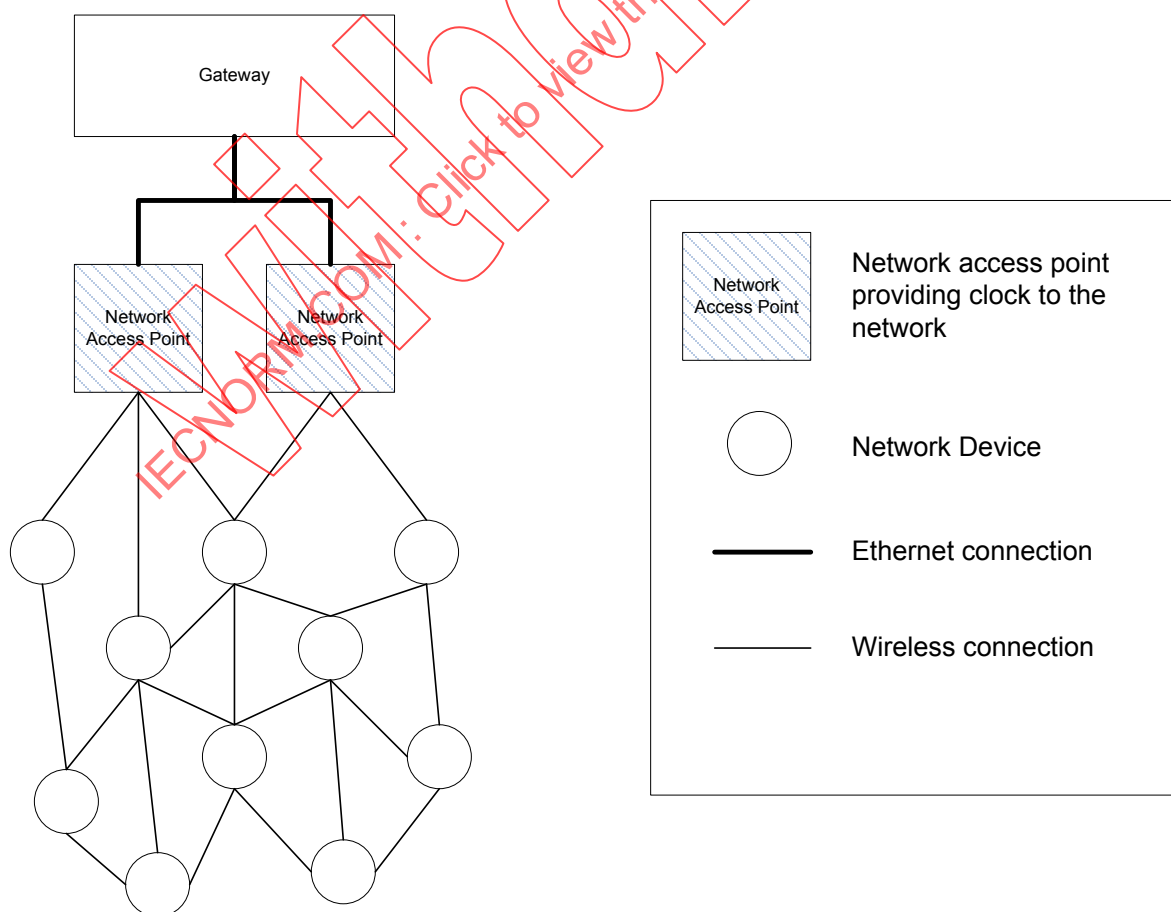


Figure D.2 – Multiple network access points with clocks

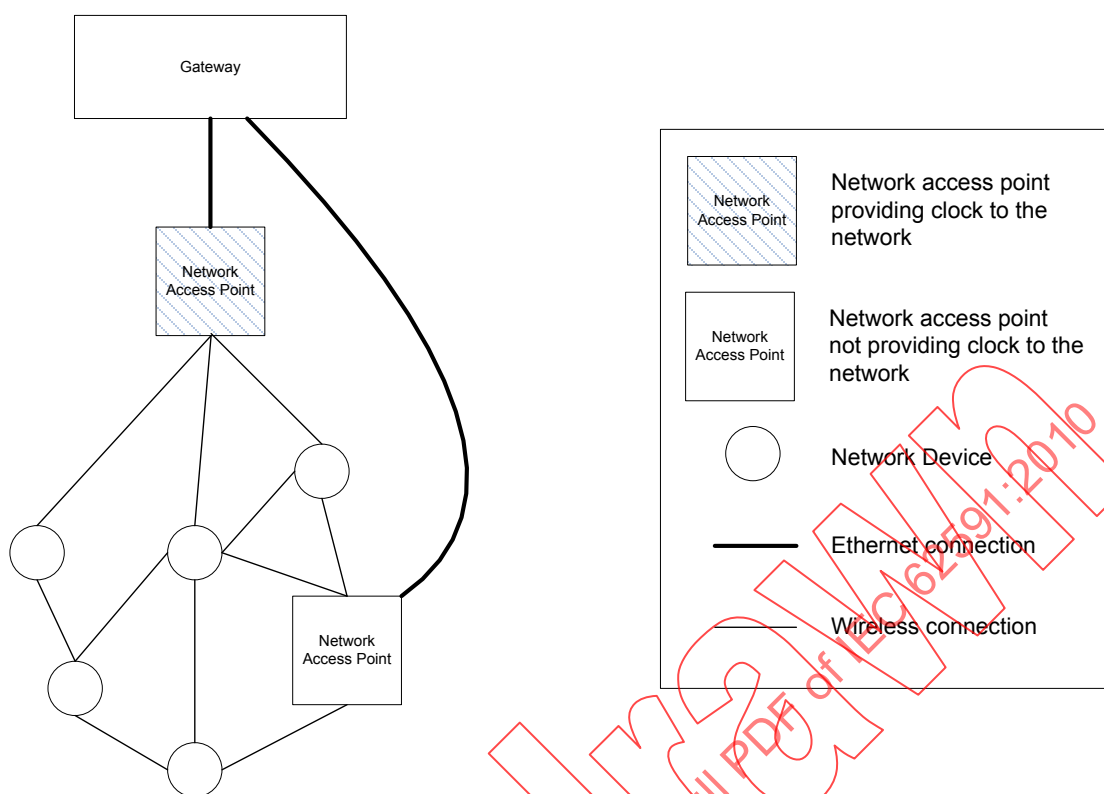


Figure D.3 – Network access point not providing clock

D.3 Functions

D.3.1 General requirements

The gateway shall provide a single entry point into the wireless network. It has a unique ID and nickname. Therefore a host application can communicate with the gateway by using APDU of this standard.

- The gateway shall participate in the wireless network as a network device.

NOTE 1 It communicates to all other network devices connected to the same wireless network through the network access point(s).

- It shall communicate directly with the network manager.
- It shall source time synchronization messages.
- It shall support one or more service access points for connecting to the automation network and plant backbone.

NOTE 2 This standard does not specify the protocol between the host and the gateway. The network used on the host side may consist of a variety of technologies. Most PLC, DCS or SCADA vendors utilize a proprietary network. Asset management and device management functions tend to use open protocols, such as TCP/IP and one of several standard MAC/PHY layers such as IEEE 802.11 and IEEE 802.3. This interface may also be HSE specified by Fieldbus Foundation.

- The gateway shall check the destination address in a request message from the host. If the request is for one of the AL services defined in this standard then it shall service that request as specified below.
 - If the destination is the gateway itself, then it shall send the response message as required by the AL protocol specified in this standard.
 - Else, if the destination address in the request from the host application is for another network device, and if the response is cached and if it is current, the gateway shall

return the cached response. For example if a host issues identify request to a device in the network, the gateway shall check to see if it has a cached response to this request.

- Else, if the gateway does not have a cached response for that device, or if it is not current, it shall forward the command to the device, using APDU specified in this standard, receive the response APDU from the device and return the resulting response to the client.
- The gateway shall implement a data cache to optimize the overall performance of the wireless network and improve responsiveness to host applications. This cache shall be used to store the response to the requests that are used to read following:
 - publish data messages,
 - event notification,
 - cached command responses,
 - diagnostics, and
 - large data transfers.

NOTE 3 The examples of large data transfers are a valve uploading its signature information and the results from a vibration analysis, data and file transfers down to a device.

- The gateway shall publish device variable data that it is caching, to other devices in the wireless network.
- The gateway shall maintain a complete list of all outstanding requests APDUs that have been sent to the network devices for which a response has not yet been received. If the response is not received within twenty-four hours, then that request shall be purged.
- The gateway shall send acknowledgement to the event notification APDU received from a network device.

D.3.2 Adding new network devices

Whenever the gateway adds a new network device in its route table, the gateway shall send the device an Identify (see 8.2.3.1 command 0) and read Long_tag (see 8.2.3.17 command 20) APDU. The responses shall be cached for that device in the gateway.

D.3.3 Device configuration change status notifications

Whenever the gateway detects that the configuration of a network device has changed, it shall send the Identify (see 8.2.3.1 command 0) APDU to that device and read its read Long_tag (see 8.2.3.17 command 20). The responses shall be cached for that device in the gateway. The configuration change counter shall be stored as part of the Identify response. If the device is a wireless adapter then the gateway shall repeat the "Join sequence for adapter sub-devices" connected to the wireless adapter.

D.3.4 Gateway change notification function

The gateway shall send change notification response (see APDU in 8.2.5.8 for the contents of the response) to the host application, when change in a network device is detected by the gateway. These notifications provide an indication that a value or status has changed, but do not include the actual changed values. When the host application receives a change notification, it can send a request to the gateway to read the associated information. For example when a host application receives a change notification from the gateway for a publish data update from a device; the host application may issue a request to the gateway to return the cached response message.

D.3.5 Cached response messages

The gateway shall be able to cache the responses to the commands listed in Table D.1.

Table D.1 – Cached Response Messages

	CMD ID^a	Command Descriptor
Cached Response in gateway upon Read confirm	0	Identify
	11	Identify
	13	Read tag, descriptor and date
	20	Read Long_tag
	48	Read additional device status
	1	Read primary variable
	2	Read loop current and percent of range
	3	Read dynamic variables and loop current
	9	Read device variables with status
Event Notification	119	Write event notification acknowledgement response
Responses Cached in gateway upon Write confirm	18	Write tag, descriptor and date
	22	Write Long_tag

^a CMD ID is the value of Command field in the APDU. It is also referred to as HART™ command number.

D.3.6 Service access point

Service access point provides a connection to the automation network and plant backbone. It is an interface between the host systems or applications and the gateway. The required functions of this interface are specified below.

- It shall provide support for accessing all network devices that are connected to the network including the devices that are connected via an adapter.
- It shall provide access to cached response messages:
 - publish data responses,
 - event notification responses,
 - cached command responses.
- It shall provide access to diagnostics.
- It shall provide access to network manager data.
- It shall support block mode data transfers.
- It shall support functions to transfer the request received from the host (connected to the gateway), that have the network device (connected to the gateway) as the destination and response from the network device to the host.

D.3.7 Communication between gateway and network manager

NOTE This standard does not specify the architecture of the network manager, gateway and security manager.

The network manager and the gateway may be physically combined or may be separate. The network manager and the gateway shall establish and maintain a secure communication channel with each other. All communications between the network manager and the network devices pass through the gateway. Therefore, the gateway shall route PDUs to the specified destination – network device, host application or network manager.

The network manager creates an initial superframe, assigns links in it for the gateway's network access points, and configures the gateway. The ASN 0 is established when this initial frame is activated.

Annex E (normative)

Command response codes

E.1 Overview

E.1.1 General

Response code is a field in the response APDU from a slave device to the master device (see 8.2.1.3). This field indicates either the successful completion of the requested command, or completion of the requested command with a warning or an error in performing the requested command. When there is an error or warning, response code supplies as much detail as possible to simplify correction of any malfunction or misinterpretation of data. The purpose of the specification in this annex is to uniformly define all responses available to manufacturers for inclusion in their devices conforming to this standard. The response code field is encoded as a 7-bit enumeration and the range of its numeric value is 0 to 127.

E.1.2 Response code classifications

Response Codes are classified two ways. The first way is the severity of the exception encountered in the device's execution of the command. It can be either the successful completion of the requested command, or completion of the requested command with a warning or an error in performing the requested command, as shown in Table 78. If the severity is "Warning" then it indicates that the command was executed with some deviation. If the severity is "Error" then it indicates that the command was not executed.

The second way of classification is the number of definitions for one value of the response code – single-definition or multiple-definitions. Single-definition values have the same meaning independent of the command that uses them. Multiple-definition values have different definitions for different commands. These values have several meanings, which vary from one command to another command. However, all values have a single meaning for a given command at all times. All devices shall use only those values for response code values that are defined in this standard for all commands specified in application layer protocol and shall comply with the definitions specified for each command. All 128 possible response code values are assigned and classified. The classifications of all values are shown in Table 78.

Table E.1 – Response code values

Value	Severity class	Number of definitions
0	Success	Single
1-7, 16-23, 32-64	Error	Single
9-13, 15, 28, 29, 65-95	Error	Multiple
24-27, 96-111	Warning	Single
8, 14, 30, 31, 112-127	Warning	Multiple

E.1.3 Reserved values

Some of the single definition values are reserved for future use by this standard. A device shall not return reserved value as response code. The reserved values are:

1, 21-27, 29, 31, 37-59, 72-95, 96-111 and 112-127.

E.1.4 Unimplemented command

All devices shall send response to all commands addressed to it, even if the device does not implement the received command. If the device does not implement a command, then It shall send the response with the code value '64', which is defined as "command not implemented".

E.1.5 Busy

A device may return "Busy" response (response code value equal to '32') in response to a master request. It indicates to the requesting application that the field device cannot begin its execution of the command due to other functions being performed. After receiving a "Busy" response from the slave, the host applications should continuously retry the request until the field device can execute the command. The device shall use "Busy" code only if it is listed in the command specification.

NOTE Response to a READ command does not return 'Busy' code.

E.2 Response code definitions

E.2.1 General

This subclause provides the actual definition associated with each value of response code. For single definition codes all devices shall use the definition described in this subclause.

For multi-definition codes a variety of example definitions are included. The actual definition for a multi-definition response code is described in the application layer protocol for each command.

E.2.2 Value '0'

Success – no errors or warnings were found in verifying the parameters for this command and it was executed successfully.

E.2.3 Value '1' – single-definition error

This value of the response code is reserved.

E.2.4 Value '2' – single-definition error

Invalid selection – the code or index was not allowed in this command or for this field device.

E.2.5 Value '3' – single-definition error

Received parameter value larger than allowed – the value of a parameter was too large and the command could not be executed in the field device.

E.2.6 Value '4' – single-definition error

Received parameter value smaller than allowed – the value of a parameter was too small and the command could not be executed in the field device.

E.2.7 Value '5' – single-definition error

Too few data octets received – the number of octets contained in the received PDU was less than required to execute the command.

E.2.8 Value '6' – single-definition error

Device-specific command error – an error has occurred for which a command-specific response code value has not been defined. This code is indicative of a device problem. If the device returns this code then it shall set the "Device malfunction" and "More status available" bits in the "Application process status" octet. The device shall provide the identification of the specific error source as a response to Command 48 "Read additional device status".

E.2.9 Value '7' – single-definition error

In write protect mode – the device is write protected and cannot accept this write command.

E.2.10 Value '8' – multi-definition warning

The exact meaning of this response code depends on the command. The command was executed, but a deviation from the request was necessary to complete the command successfully. The values actually used are returned in the response data octets. A few examples are shown below.

Update failure – the real-time data returned from the field device has not changed since the last time it was read.

Update in progress – the command cannot be executed, because the device is pending the completion of a command that requires a relatively long time to. The Client should retry the request after a delay.

Update times adjusted – the update period in the command was invalid. The device has corrected the period as returned in the response.

Update period increased – there is insufficient bandwidth to fulfill the publish rate requirement. The "Publish message" is enabled at a reduced rate.

Set to the nearest possible value – the data sent to the device has been rounded or truncated due to limitations within the device. The command has been accepted.

All but running delayed responses flushed – a delayed response is being performed by the device that cannot be interrupted or cancelled by Command 106 "Flush delayed responses".

E.2.11 Value '9' – multi-definition error

The exact meaning of this response code depends on the command. A few examples are shown below.

Lower range value too high – the "Lower range value" in the request was above the "Upper transducer limit" or some other physical device limitation is exceeded.

Applied process too high – the process applied to the field device was too high.

Invalid publish message – the "Publish message number" is not supported by the device.

Configuration change counter mismatch – the "Configuration change counter" value did not match the current value in the device. The "Configuration changed" bit remains set.

Insufficient bandwidth – there is not enough bandwidth available to enable the designated publish message.

E.2.12 Value '10' – multi-definition error

The exact meaning of this response code depends on the command. A few examples are shown below.

Lower range value too low – the “Lower range value” in the request was below the “Lower transducer limit” or some other physical device limitation is exceeded.

Applied process too low – the process applied to the field device was too low.

Invalid local panel lock – the code received in the request is not supported or the code is invalid.

E.2.13 Value '11' – multi-definition error

The exact meaning of this response code depends on the command. A few examples are shown below.

Upper range value too high – the “Upper range value” in the request was above the “Upper transducer limit” or some other physical device limitation is exceeded.

Invalid device variable code – the “Device variable” specified in the request is valid, but it is not supported for the received command or the operation specified by the command.

Invalid device variable classification – the “Device variable classification code” included in the received write command is not valid for the “Device variable” specified in the received command.

Cannot lock panel – the device is in a mode that does not allow the local panel to be locked for example the operator is currently using the local panel.

E.2.14 Value '12' – multi-definition error

The exact meaning of this response code depends on the command. A few examples are shown below.

Upper range value too low – the “Upper range value” in the request was below the “Lower transducer limit” or some other physical device limitation is exceeded.

Invalid units code – the “units code” specified in the request is not supported for the received command or the “Device variable” or “Dynamic variable” specified in the received command.

Invalid slot number – the requested slot code to capture the “Dynamic variable” or “Device variable” is not valid for the received command.

E.2.15 Value '13' – multi-definition error

The exact meaning of this response code depends on the command. A few examples are shown below.

Invalid transfer function code – the requested transfer function is not supported for this “Dynamic variable”.

Upper and lower range values out of limits – the “Upper and Lower range values” in the received request are outside the transducer limits or some other physical device limitation has been exceeded.

Computation error – an arithmetic error was encountered while the device was attempting to apply the new values in the received request.

Command number not supported – the requested command number to be captured is not supported in the device.

E.2.16 Value '14' – multi-definition warning

The exact meaning of this response code depends on the command. The command was executed, but a deviation from the request was necessary to complete the command successfully. The values actually used are returned in the response data octets. A few examples are shown below.

Span too small – the span as determined from the “Upper and Lower range values”, was below the minimum span supported by the device.

Dynamic variables returned for device variables – the device does not expose any “Device variables”. Therefore, “Dynamic variable” values were returned for codes 0-3 in Command 9.

Update rate uncertain – the device has not yet joined the network and therefore, it does not know if there will be sufficient capacity to meet the required notification parameters.

New lower range value pushed upper range value over transducer limit – the device used the “Lower range value” received in the requested and modified the “Upper range value” accordingly. The “Upper range value” is saturated and the span is not maintained.

E.2.17 Value '15' – multi-definition error

The exact meaning of this response code depends on the command. An example is shown below.

Invalid analog channel code number – the analog channel does not exist in the device.

E.2.18 Value '16' – single-definition error

Access restricted – the command is not allowed for the device in its current operating mode. For example the device is locked for Command 71.

E.2.19 Value '17' – single-definition error

Invalid device variable index – the “Device variable” in the received request does not exist in the device.

E.2.20 Value '18' – single-definition error

Invalid units code – the “Units code” in the received request is not supported for the received command or the “Device variable” or “Dynamic variable” specified in the received command.

E.2.21 Value '19' – single-definition error

Device variable index not allowed – the “Device variable” in the received request is not supported for the received command.

E.2.22 Value '20' – single-definition error

Invalid extended command number – the extended command number of the received command is less than 512.

E.2.23 Values '21'to '23' – single-definition error

These values of the response code are reserved.

E.2.24 Values '24'to '27' – single-definition warning

These values of the response code are reserved.

E.2.25 Value '28' – multi-definition error

The exact meaning of this response code depends on the command. An example is shown below.

Invalid 'range units code' – the “Units code” in the received request is not supported by the device for this “Dynamic variable” specified in the received command.

E.2.26 Value '29' – multi-definition error

This value of the response code is reserved.

E.2.27 Value '30' – multi-definition warning

The exact meaning of this response code depends on the command. The command was executed, but a deviation from the request was necessary to complete the command successfully. The values actually used are returned in the response data octets. An example is shown below.

Command response truncated – the device was not able to return as many “Device variable” values as requested. This is usually caused by limited command buffer length.

E.2.28 Value '31' – multi-definition warning

This value of the response code is reserved.

E.2.29 Value '32' – single-definition error

Busy – this code can be caused by one of two conditions:

- a) the device is busy performing a function that cannot be interrupted by this command; or
- b) the execution of the requested command requires the delayed response mechanism and all delayed response buffers are used.

In either case the requesting application should retry this request several times until the field device can execute the command.

NOTE The recommended value of retries of the command that causes “Busy” response is 30 or more.

E.2.30 Value '33' – single-definition error

Delayed response initiated – the command could not be serviced within the response time specified by the data link layer protocol and therefore the delayed response was initiated. The requesting application should retry the exact same request after a time delay (for example after delaying several seconds). When communicating via a multiplexer this delay could be as little as half a second.

E.2.31 Value '34' – single-definition error

Delayed response running – the execution of the delayed response is not yet finished. The command execution process is still running in the responding device.

E.2.32 Value '35' – single-definition error

Delayed response dead – this response indicates that the responding device did not reply to the request. This response is valid for intelligent bridging devices only.

E.2.33 Value '36' – single-definition error

Delayed response conflict – the requested command would cause a conflict with a delayed response currently executing in the responding device.

E.2.34 Values '37' to '59' – single-definition error

These values of the response code are reserved.

E.2.35 Value '60' – single-definition error

This code is used in the ACK DLPDU. Use of this code indicates that the DLPDU was received correctly by the peer device but was not accepted by the peer device. The received DLPDU is not stored or forwarded by the device generating the ACK DLPDU.

Payload too long – the command payload is too long to fit in the DLPDU.

E.2.36 Value '61' – single-definition error

This code is used in the ACK DLPDU. Use of this code indicates that the DLPDU was received correctly by the peer device but was not accepted by the peer device. The received DLPDU is not stored or forwarded by the device generating the ACK DLPDU.

No buffers available – the responding device had no buffers available to accept the DLPDU.

E.2.37 Value '62' – single-definition error

This code is used in the ACK DLPDU. Use of this code indicates that the DLPDU was received correctly by the peer device but was not accepted by the peer device. The received DLPDU is not stored or forwarded by the device generating the ACK DLPDU.

No alarm/event buffers available – the responding device did not have any alarm or event buffer available to receive the DLPDU.

E.2.38 Value '63' – single-definition error

This code is used in the ACK DLPDU. Use of this code indicates that the DLPDU was received correctly by the peer device but was not accepted by the peer device. The received DLPDU is not stored or forwarded by the device generating the ACK DLPDU.

Priority too low – the priority of the received DLPDU was too low for the responding device to accept it.

E.2.39 Value '64' – single-definition error

Command not implemented – the requested command is not implemented. The device shall not return this value for a command, if the device is required to implement the received command based on the profile class of the device.

E.2.40 Value '65' – multi-definition error

The exact meaning of this response code depends on the command. The descriptions are shown below.

Declined,
Operator overridden,
Force join declined,
Invalid network_ID,
Invalid Timer_type,
Invalid hop-to-live,
Invalid Join_priority,
Unknown nickname,
Invalid nickname,
Service request denied,
Link not found,
No more entries available,
Entry not found,
Illegal frequency channel bits,
Unknown unique ID,
Key change failed,
Session with given peer device does not exist,
Invalid suspend time.

E.2.41 Value '66' – multi-definition error

The exact meaning of this response code depends on the command. The descriptions are shown below.

Invalid number of slots,
Session_type invalid,
Invalid execution time,
Unknown notification flag,
Device list conflict,
Invalid neighbor attribute,
Invalid service ID,
Invalid Graph_ID,
Invalid resume time,
Link already exists,
Invalid reason code,
Unknown service flag,
Invalid timer interval.

E.2.42 Value '67' – multi-definition error

The exact meaning of this response code depends on the command. The descriptions are shown below.

Invalid Graph_ID,
Unknown Superframe_ID,
Invalid superframe mode,

Reason code rejected, service not deleted,
Unknown application domain.

E.2.43 Value '68' – multi-definition error

The exact meaning of this response code depends on the command. The descriptions are shown below.

Unknown nickname,
Invalid execution time,
Invalid slot number.

E.2.44 Value '69' – multi-definition error

The exact meaning of this response code depends on the command. One description is shown below:

Unknown link options.

E.2.45 Value '70' – multi-definition error

The exact meaning of this response code depends on the command. One description is shown below:

Invalid Channel_Offset.

E.2.46 Value '71' – multi-definition error

The exact meaning of this response code depends on the command. One description is shown below:

Invalid Link_type.

E.2.47 Values '72' to '95' – multi-definition error

These values of the response code are reserved.

E.2.48 Values '96' to '111' – single-definition warning

These values of the response code are reserved.

E.2.49 Values '112' to '127' – multi-definition warning

These values of the response code are reserved.

Annex F (normative)

Common tables

F.1 Overview

F.1.1 General

The protocol of this standard uses lookup tables to associate specific definitions to numeric codes or to define the interpretation of an individual bit that is set to '1'. If the protocol references a table, the numeric values have specific definitions that allow all hosts to consistently utilize that data. This Annex contains lookup tables (for example engineering units, Device variable classification code etc.) used for Enumeration or Bit Field data types.

If one of the tables in this Annex is referenced by the protocol specification, then the table or a subset of that table shall be used exactly as specified in this Annex and undefined (i.e., unlisted) values shall not be used.

F.1.2 Enumeration

Data items that take on a single meaning from a list or table are encoded as enumerations and communicated as an unsigned integer of 1-octet or 2-octet length. Not all codes are allocated sequentially. Any intervening undefined codes are reserved for future use by this standard. Reserved values shall not be returned or transferred by any device. The code values 250 to 255 for all enumeration data types have the identical definition as shown in Table F.1.

Table F.1 – Reserved values

Code	Meaning	Description
250	Not used	implies that the responding device does not use this table
251	None	implies that the responding device does not use this table or does not use any other code defined in the table
252	Unknown	implies that the responding device uses this table, but has not yet been configured
253	Special	implies that the responding device uses this table, but does not use standard definition
254	Expansion	implies an expansion of a definition
255	Reserved	implies reserved by this standard for future use
NOTE The code 253 Special may be used in some instances for example when the flow units are being set by the user to something unusual like acre-feet/hour.		

F.1.3 Bit Field

Communication of information encoded as single-bit data (such as status and diagnostic information) is communicated using Bit Field as shown in 8.2.7.7. Individual bits are packed together, and reserved bits are added as necessary to transfer one octet. The lookup tables in this annex define the meaning of each bit in the Bit Field. Each bit shall be either true (set to a 1) or false (set to a 0). Any undefined bits are reserved for future use by this standard and shall be encoded as '0'.

The Bit Field data type is one of the following sizes:

- 4-bit,
- 1-octet, or

2-octets.

The 4-bit Bit Field uses the following format for the bits:

0x01 for bit 1 (least significant bit),
 0x02 for bit 2,
 0x04 for bit 3, and
 0x08 for bit 4 (most significant bit).

The 1-octet and the first eight bits of 2-octet Bit Fields use the following format:

0x01 for bit 1 (least significant bit),
 0x02 for bit 2,
 0x04 for bit 3,
 0x08 for bit 4,
 0x10 for bit 5,
 0x20 for bit 6,
 0x40 for bit 7, and
 0x80 for bit 8 (most significant bit of the first octet).

The 2-octet Bit Field uses the following format for the next eight bits:

0x0100 for bit 9 (least significant bit of the most significant octet),
 0x0200 for bit 10,
 0x0400 for bit 11,
 0x0800 for bit 12,
 0x1000 for bit 13,
 0x2000 for bit 14,
 0x4000 for bit 15, and
 0x8000 for bit 16 (most significant bit of the most significant octet).

F.2 Table definitions

F.2.1 Publish mode control codes

The 1-octet enumeration values for publish mode control are shown in Table F.2.

Table F.2 – Publish mode control codes

Code	Publish mode
0	Off
1	Enable Publish on token-passing data-link layer only
2	Enable Publish on TDMA data-link layer only
3	Enable Publish on TDMA and token-passing data-link layers

F.2.2 Write device variable codes

These codes as shown in Table F.3 indicate whether the “Device variable” value is a fixed value or if its value is derived from the normal operation.

Table F.3 – Write device variable codes

Code	Description
0	Normal
1	Fixed value

F.2.3 Device variable family codes

These codes indicate which family the “Device variable” belongs to as shown in Table F.4. If a device does not support the “Device family” commands for a “Device variable”, then the device shall respond with 250 (“Not used”).

Table F.4 – Device variable family codes

Code	Device variable family
0-3	Reserved
4	Temperature
5	Pressure
6	Valve or Actuator
7	Simple PID control
8	pH
9	Conductivity
10	Totalizer
11	Level
12	Vortex Flow
13	Mag Flow
14	Coriolis Flow
250	Not used

F.2.4 Device variable classification codes

These 1-octet enumeration codes indicate the function performed by the “Device variable” as shown in Table F.5. This identifies the type of process connection supported by the “Device variable” and the “Engineering unit code” table to be used.

Table F.5 – Device variable classification codes

Code	Device variable classification
0	Device variable not classified
1-63	Reserved
64	Temperature
65	Pressure
66	Volumetric flow
67	Velocity
68	Volume
69	Length
70	Time
71	Mass
72	Mass flow
73	Mass per volume
74	Viscosity
75	Angular velocity
76	Area
77	Energy (work)
78	Force
79	Power
80	Frequency
81	Analytical
82	Capacitance
83	Emf
84	Current
85	Resistance
86	Angle
87	Conductance
88	Volume per volume
89	Volume per mass
90	Concentration
91	Valve actuator
92	Level
93	Vortex flow
94	Mag flow
95	Coriolis flow

F.2.5 Analog channel saturated codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.6.

Table F.6 – Analog channel saturated codes

Code	Channel number
0x01	Analog channel 1
0x02	Analog channel 2
0x04	Analog channel 3
0x08	Analog channel 4

F.2.6 Analog channel fixed codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.7.

Table F.7 – Analog channel fixed codes

Code	Analog channel number
0x01	Analog channel 1
0x02	Analog channel 2
0x04	Analog channel 3
0x08	Analog channel 4

F.2.7 Standardized status 0 codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.8.

Table F.8 – Standardized status 0 codes

Code	Name	Description
0x01	Simulation active	The device is in simulation mode and the level output is not representative of the process.
0x02	Non-volatile memory defect	The Non-volatile memory check is invalid or maybe corrupt, or the battery of a battery-backed memory is defective.
0x04	Volatile memory defect	The RAM memory check is invalid or maybe corrupt.
0x08	Watchdog reset executed	A watchdog reset has been performed.
0x10	Voltage conditions out of range	A voltage condition is outside its allowable range.
0x20	Environmental conditions out of range	An internal or environmental condition is beyond acceptable limits.
0x40	Electronic defect	A hardware problem not related to the sensor has been detected.
0x80	Reserved	

F.2.8 Standardized status 1 codes

The Table F.9 is reserved for future use. Each bit of this 1-octet code is used as a Bit Field.

Table F.9 – Standardized status 1 codes

Code	Name	Description
—	—	—

F.2.9 Standardized status 2 codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.10.

Table F.10 – Standardized status 2 codes

Code	Name	Description
0x01	Sub-device list changed	The I/O system has lost communication with one of its sub-devices or discovered a new sub-device. This bit is reset by re-reading the sub-device list using command 84.
0x02	Duplicate master detected	The Adapter has discovered another master with the same address connected to its token-passing interface.

F.2.10 Standardized status 3 codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.11.

Table F.11 – Standardized status 3 codes

Code	Name	Description
0x01	Capacity denied	The device was unable to acquire the communication bandwidth required to support the publish messaging specified. This shall also be set if the network manager reduces the bandwidth allocated to the device.
0x02	Duplicate master detected	The Adapter has discovered another master with the same address connected to its token-passing interface.
0x04	Bandwidth allocation pending	The device has requested the bandwidth from the network manager and is awaiting network manager response.
0x08	Block transfer pending	The device has a data set (for example a waveform) awaiting transfer to the host application or the gateway.

F.2.11 Publish trigger mode codes

It indicates the trigger condition for publishing a message as shown in Table F.12.

Table F.12 – Publish trigger mode codes

Code	Name	Description
0	Continuous	The publish message is published continuously at (worst case) the minimum update period.
1	Window	The publish message is triggered when the source value deviates more than the specified trigger value.
2	Rising	The publish message is triggered when source value rises above the specified trigger value.
3	Falling	The publish message is triggered when the source value falls below the specified trigger value.
4	On-change	The publish message is triggered when any value in the message changes.

F.2.12 Event notification control codes

Each bit of this 4-bit code is used as a Bit Field as shown in Table F.13. It occupies the least significant four bits of one octet.

Table F.13 – Event notification control codes

Code	Description
0x00	Off
0x01	Enable event notification on token-passing data link layer
0x02	Enable event notification on TDMA data-link layer
0x03	Enable event notification on TDMA and token-passing data link layers

F.2.13 Event status codes

Each bit of this 4-bit code is used as a Bit Field as shown in Table F.14. It occupies the most significant four bits of one octet.

Table F.14 – Event status codes

Code	Definition
0x01	Configuration changed event pending
0x02	“Application process status” event pending
0x04	More status available event pending

F.2.14 Timetable request codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.15.

Table F.15 – TimeTable request codes

Code	Definition
0x01	Source
0x02	Sink
0x04	Intermittent

F.2.15 Timetable application domain codes

It is 1-octet enumeration data as shown in Table F.16.

Table F.16 – TimeTable application domain codes

Code	Definition
0	Publish
1	Event
2	Maintenance
3	Block transfer

F.2.16 Synchronous action control codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.17.

Table F.17 – Synchronous action control codes

Code	Name	Description
0x01	Command	When set to '1', the indicated command will be executed at the time indicated. When reset to '0', a "Device variable" is to be sampled.
0x10	One-shot	When set to '1', the synchronous action shall be performed one time only. When reset to '0', the sampling will trigger at the indicated time and repeat continuously thereafter.
0x80	Action enabled	When set to '1', the action is enabled. When action is "One-shot" then the enabled bit is reset after the action occurs.

F.2.17 Real-time clock flags codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.18.

Table F.18 – Real-time clock flags codes

Code	Name	Description
0x01	Non-volatile clock	When set to '1', the device contains a battery-backed clock. In this case, the clock does not need to be reset if there is a power failure.
0x02	Clock un-initialized	The real-time clock has never been set with the date and time. For example the clock is volatile and power was removed from and restored to the device.

F.2.18 Wireless timer codes

It is 1-octet enumeration data as shown in Table F.19.

Table F.19 – Wireless timer codes

Code	Timer_type
0	Discovery
1	Advertisement
2	Keep-alive
3	Path failure
4	Health report
5	Broadcast reply
6	Maximum PDU age

F.2.19 Device power source codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.20.

Table F.20 – Device power source codes

Code	Name	Description
0x01	Line power	Device is powered via a hardwired connection to a plant power source.
0x02	Battery power	Device operates solely off its (internal) battery.
0x04	Rechargeable battery power or power scavenging	The device has a short-term supply of energy (shall be at least one hour) that is continuously being replenished. The device's power is being replenished by harvesting and converting energy from the environment surrounding the device (for example solar, vibration, heat).
0x08 – 0x80	Reserved	

F.2.20 Link_type codes

It is 1-octet enumeration data as shown in Table F.21.

Table F.21 – Link_type codes

Code	Link_type
0	Normal
1	Discovery
2	Broadcast
3	Join

F.2.21 Link option flags codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.22.

Table F.22 – Link option flags codes

Code	Definition
0x01	Transmit
0x02	Receive
0x04	Shared

F.2.22 Superframe mode flags codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.23.

Table F.23 – Superframe mode flags codes

Code	Name	Description
0x01	Active	This flag is '0' (inactive) on reset.
0x80	Handheld superframe	This bit shall only be set for one superframe in the device.

F.2.23 Session_type codes

It is 1-octet enumeration data as shown in Table F.24.

Table F.24 – Session_type codes

Code	Session_type
0	Unicast
1	Broadcast
2	Join

F.2.24 Timetable deletion reason codes

It is 1-octet enumeration data as shown in Table F.25.

Table F.25 – TimeTable deletion reason codes

Code	Reason
0	Requested by peer
1	Timetable cannot be established
2	Network failure

F.2.25 Disconnect cause codes

It is 1-octet enumeration data as shown in Table F.26.

Table F.26 – Disconnect cause codes

Code	Cause
0	User-initiated
1	Communication failure

F.2.26 Wireless operation codes

It is 1-octet enumeration data as shown in Table F.27.

Table F.27 – Wireless operation codes

Code	Operation
0	Idle
1	Active search
2	Negotiating
3	Quarantined
4	Operational
5	Suspended
6	Deep sleep or Ultra-low power or Passive search

F.2.27 Join process status codes

Each bit of this 2-octet code is used as a Bit Field as shown in Table F.28.

Table F.28 – Join process status codes

Code	Name	Description
0x0001	Network packets heard	
0x0002	ASN acquired	
0x0004	Synchronized to slot time	
0x0008	Advertisement heard	
0x0010	Join requested	It is set to '1' on transmission of first join request.
0x0020	Join retrying	It is set to '1' after the first join request retry (i.e., when number of join requests is more than 2). It is reset to '0' when device is authenticated or when "Active search" is started.
0x0040	Join failed	It is set to '1' on transition from "Active search" mode to "Deep sleep" mode. It is reset to '0' on transition from "Deep sleep" mode to "Active search".
0x0080	Authenticated	It is set to '1' when network key, network manager session are established.
0x0100	Network joined	It is set to '1' when normal superframe and links are obtained.
0x0200	Negotiating network attributes	It is set to '1' when gateway session is obtained. Initial bandwidth requirements may not have been negotiated with network manager.
0x0400	Normal operation commencing	

F.2.28 Security type codes

It is 1-octet enumeration data as shown in Table F.29.

Table F.29 – Security type codes

Code	Security type
0	Session keyed
1	Join keyed
2	Handheld keyed

F.2.29 Device list codes

It is 1-octet enumeration data as shown in Table F.30.

Table F.30 – Device list codes

Code	Device list
0	Active devices
1	White-listed devices
2	Blacklisted devices

F.2.30 Network access mode codes

It is 1-octet enumeration data as shown in Table F.31.

Table F.31 – Network access mode codes

Code	Network access mode
0	Open – accept all devices
1	Use white-list devices for access
2	Use blacklist devices for access
3	Use white-list and blacklist devices for access
4	Lockdown – disable join links

F.2.31 Device profile codes

It is 1-octet enumeration data as shown in Table F.32.

Table F.32 – Device profile codes

Code	Device profile
1	HART™ process automation device
2	HART™ discrete device
3	Hybrid – process automation and discrete device
4	I/O system
129	Wireless process automation device
130	Wireless discrete device
131	Wireless hybrid – process automation and discrete device
132	Wireless gateway
141	Wireless process adapter
142	Wireless discrete adapter
144	Wireless-enabled handheld or portable maintenance tool

F.2.32 Device power status codes

It is 1-octet enumeration data as shown in Table F.33.

Table F.33 – Device power status codes

Code	Power status
0	Nominal
1	Low
2	Critically low
3	Recharging – low
4	Recharging – high

F.2.33 Neighbor flags codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.34.

Table F.34 – Neighbor flags codes

Code	Neighbor type
0x01	Time source

F.2.34 Notification mask codes

Each bit of this 2-octet code is used as a Bit Field as shown in Table F.35.

Table F.35 – Notification mask codes

Code	Name	Description
0x0001	Publish mode	
0x0002	Event notification	
0x0004	Application process status	
0x0008	Device configuration	
0x0010	Network topology	target device shall be gateway or network manager
0x0020	Network schedule	target device shall be gateway or network manager

F.2.35 Join mode codes

It is 1-octet enumeration data as shown in Table F.36.

Table F.36 – Join mode codes

Code	Join mode description
0	Don't attempt to join.
1	Join now and return to join mode 2 after joining.
2	Attempt to join immediately on power up or reset.

F.2.36 Device scheduling flags codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.37.

Table F.37 – Device scheduling flags codes

Code	Device type
0x01	Transient device
0x02	Non-routing device
0x04	Handheld device

F.2.37 Network optimization flags codes

Each bit of this 1-octet code is used as a Bit Field as shown in Table F.38.

Table F.38 – Network optimization flags codes

Code	Optimization
0x01	Low latency
0x02	Use line powered devices as backbone

IECNORM.COM: Click to view the full PDF of IEC 62591:2010

Annex G

(informative)

Application of publish data mode and event commands

G.1 Publish data mode operation

G.1.1 Overview

The Type 20 application layer protocol supports the publishing of cyclical process data. In publish data mode, a device is instructed to publish the response to a command continuously without any further Master action. This mode requires that the Master setup various parameters and then enable publishing. A device can be configured to publish several data sets (messages). If a device implements this mode, then it supports at least three publish data messages. The publish data mode commands are:

- Command 103 (see 8.2.3.27) to write publishing period;
- Command 104 (see 8.2.3.28) to write publishing trigger;
- Command 105 (see 8.2.3.29) to read publish data mode Configuration;
- Command 107 (see 8.2.3.31) to write Device variables to be published;
- Command 108 (see 8.2.3.32) to write Command number for publishing;
- Command 109 (see 8.2.3.33) to enable or disable publishing.

G.1.2 Configuring a device for publish data mode operation

The Master should use the following procedure to place a field device into publish data mode.

- a) Configure the command response by using Command 108 'to write Command number' to for publishing. Commands 1, 2, 3, 9 and 48 are available for publishing and Command 33 is available, if it is implemented in the field device.
- b) Assign Device variables to the response data slots for Commands 9 and 33 by using Command 107 'to write Device variables to be published'. Command 9 supports up to eight slots and Command 33 supports up to four slots.
- c) Configure the update period and the maximum update period for publishing the data by using Command 103 'to write publishing period'. The master can use Command 54 to determine the update rates for the desired Device variables. The data-link requirements may result in publishing more frequently than the configured update period.
- d) Configure the Trigger mode and level for publishing by using Command 104 'to write publishing trigger'. Publishing may be configured to occur continuously or based on the value of the device variable. Publishing will always occur at least as often as the maximum update period – see Figure 45, Figure 46 and Figure 47.
- e) Issue Command 109 'to enable or disable publishing'. The field device follows the procedure specified in 8.2.3.33.3 and shown in Figure G.1. The network manager also sends request to other network devices to setup or delete superframe and link for publishing.

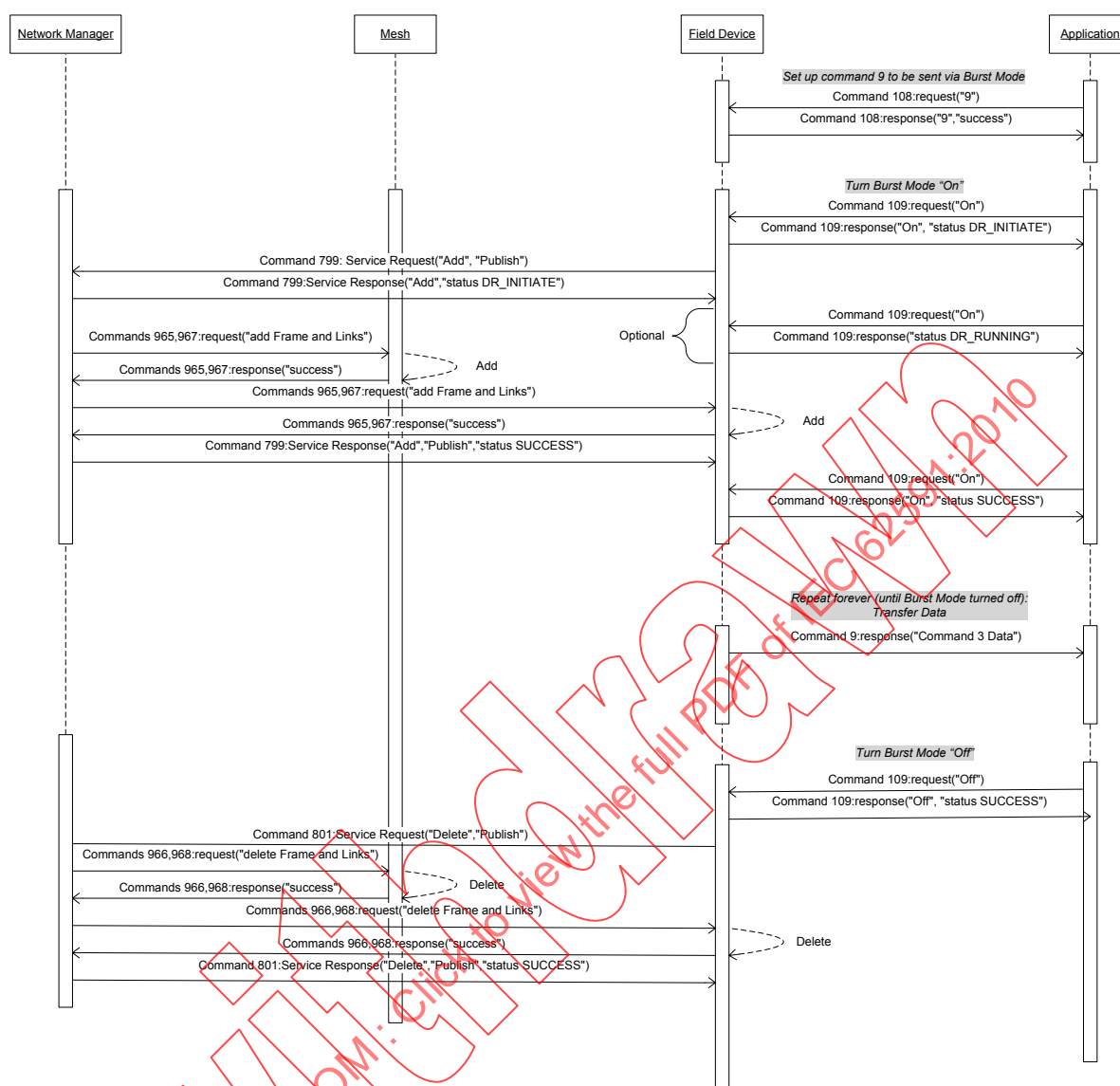


Figure G.1 – Enabling or disabling publishing sequence

G.1.3 Response for publish data mode operation

While the publishing is enabled, the field device (slave) transmits the responses to the command number set by Command 108 based on the data-link requirements and the attributes configured using the other publish data mode commands.

Once the device is in publish data mode, Command 108 can be used to change the publish data command response. A device may take one publish cycle before the response changes to the new command number. Other publish data mode commands may also be used to adjust publish data mode operation on-the-fly.

The wireless field device aggregates multiple pending messages into one transaction, whenever possible (see 8.2.2.5). If the publish data messages does not fit into a single transaction, then these can be sent in separate transactions.

G.1.4 Publish data mode update periods

The update period is constrained by the protocol – see 8.2.3.27.3. For the Physical layer specified by IEEE STD 802.15.4-2006, the minimum period is 0,1 s. The Update period

parameter of Command 103 determines the communication rate. The acquisition period for a Device variable can be different; it can be read using command 54.

G.2 Event notification operation

G.2.1 Overview

Event notification requires, and is built upon, publish data mode operation. If a device supports publish data mode, then it also supports all of event notification commands. The Type 20 application layer protocol offers two distinct methods to display events – the “Application process status” and the response to command 48.

Event notification publishes changes in the device’s status, independently from data publishing supported in other publish data mode commands. For events the status included in the “Application process status” octet, “Extended status” octet and command 48 can be used. It is possible to specify a subset of all these status bits that will trigger event notification. Event notification has a lower priority than other types of data transfer, but it requires a time stamp in order to indicate the first time when a notification occurred. Event notification is usually not required for monitoring or control; its use is rather infrequent. Event Notifications are aggregated with publish data messages, whenever possible.

The Event Notification commands are:

- Command 115 (see 8.2.3.34) to read the configuration of the Event notification,
- Command 116 (see 8.2.3.35) to select the bits that can trigger an Event notification,
- Command 117 (see 8.2.3.36) to control the timing of Event notifications,
- Command 118 (see 8.2.3.37) is to enable or disable Event notification, and
- Command 119 (see 8.2.3.38) to acknowledge the Event notification.

The device retains Event notification settings through a device reset, self test or the power being removed and reapplied. While event transitions do not need to be retained through power cycles, the device should buffer five transitions buffered; required minimum is three.

G.2.2 Configuring a device event notification operation

The Master should use the following procedure to configure Event notifications.

- Configure the status bits that will trigger Event notification by using Command 116 ‘to write Event notification bit mask’.
- Configure how long the event has to be active before an event is recognized by configuring De-bounce interval associate with that event. Configure how often the device has to send the event notification before it receives the acknowledgement. Both of these timings are set by using Command 117 ‘to write Event notification timing’.
- Enable the event notification by using Command 118 ‘to enable Event notification control’. This command also selects the interface to send the event notification.

G.2.3 Handling of Event Notification in a Device

Any change in the status bits for which bit mask is ‘1’ is considered as an event. The first occurrence of the event is captured and time stamped. If the event persists for duration equal to Event de-bounce interval, then the Event notification is generated and the Command 119 response transmitted. Command 119 is transmitted repeatedly at the rate indicated by the Event notification retry time until a Command 119 from a Master request acknowledges the event.

If there was no event generated during a Maximum update time, the device publishes the Command 119 response one time to the gateway. The latest values for “Application process

status”, the Configuration change counter and the Command 48 response octets are always included in the Command 119.

Command 116 is used to identify the bits that may trigger the Event notification. Command 119 only returns only those status octets, for which the Event mask was established using Command 116 (i.e., Command 116 may be truncated). The octets returned in Command 119 include the current “Application process status” and Command 48 data regardless of which bits are masked. The time stamp remains the same until an acknowledgement is received. If an acknowledgement is received, the time stamp is set to the time when the acknowledgement was received.

Event Notification and Time Stamps are not required to be maintained through power cycles or a device reset.

Withdawn
IECNORM.COM: Click to view the full PDF of IEC 62591:2010

Annex H (informative)

Network redundancy

The network redundancy is achieved by having more than one route from one device to another device, to the gateway and to the network manager. It is done by graph routing described in 6.5.1.3.2.

The network redundancy to the gateway is achieved by having more than one network access point as shown in Figure H.1 and as specified in Annex D.2. This figure also shows the relationship of the network manager to the wireless network. The network manager is responsible for establishing network routing.

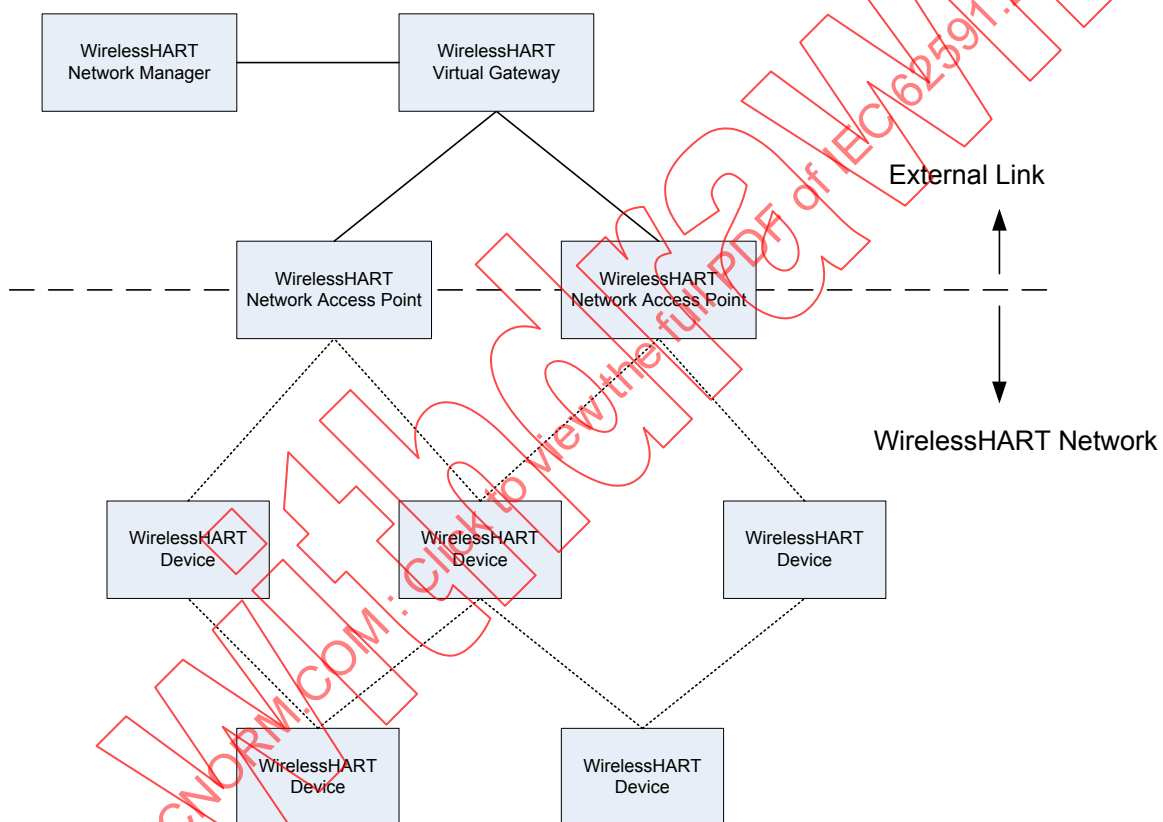


Figure H.1 – Network routing

If one of the access points fails, the communication between a network device and the gateway is routed via the other operational access point.

The redundancy of the gateway is outside the scope of this standard.

Annex I (informative)

Network manager implementation

I.1 Overview

The network manager is responsible for the overall management, scheduling, and optimization of the wireless network. The functions to be performed by a network manager in wireless network are:

- network formation and configuration,
- route setup and distribution of routes to the network devices,
- generation of network schedule and transfer to the network devices,
- channel management, and
- network diagnostics and adapting.

This annex describes the operation of these functions and guidelines on how these functions can be implemented.

I.2 Network manager model

I.2.1 General

The network manager data model is shown in Figure I.1. The data components are:

- the network schedule,
- the collection of network devices,
- the collection of neighbor tables,
- the collection of connection tables, and
- the collection of routes.

The network manager also maintains an association with the security manager.

When a new device joins the network, the network manager is responsible for managing the join process. As part of this join process, the network manager uses its model of the network along with algorithms that it has to optimize the network. After the schedule has been generated, the network and schedule information is transferred through the network in reverse order, i.e., from the gateways to each field device.

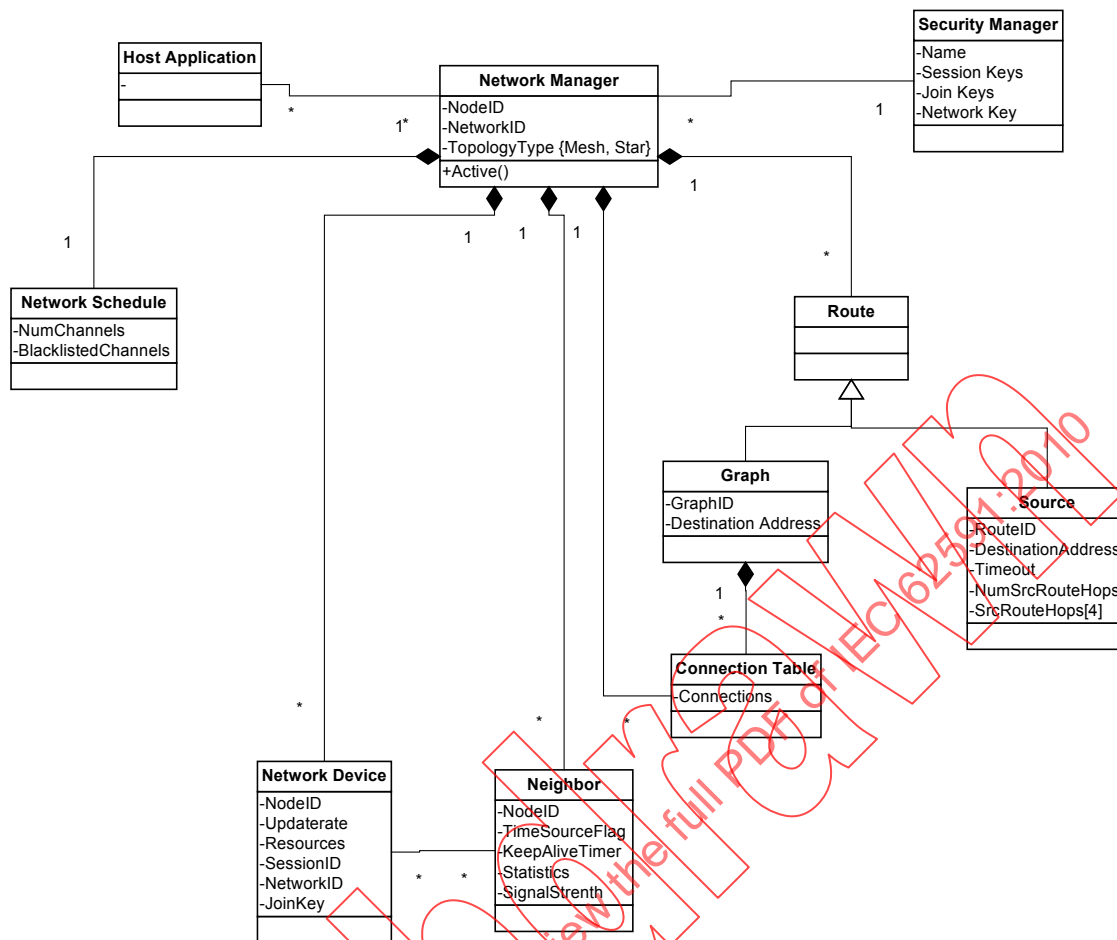


Figure I.1 – General model for network manager

The network manager generates and maintains all of the route information for the network. It uses the list of network devices, their reported neighbors and device capacities to generate a complete graph leading from each network device to the network manager. There may also be special purpose routes that are used to send commands and other settings from the gateway to network devices. There are broadcast routes that are used to send broadcast messages from the network manager through the gateway to all of the network devices.

NOTE Each graph should use a maximum of 4 neighbors as a potential next hop destination.

Special purpose routes are also created. These routes provide paths from the gateway to devices and from devices to gateway. These are referred to as downstream and upstream paths. For example upstream paths are used for transfer of periodic process data from the devices to gateway and downstream paths are used for transferring control information from the gateway to the actuators.

The overall routing information is assembled by the network manager using device, neighbor and diagnostic information reported by the network devices. After determining the routing information and communication requirements for all of the devices, the scheduling of network resources can be performed. The network manager contains one overall network schedule.

The network manager is responsible for adapting the network to changing conditions and for scheduling communication resources. As devices join and leave the network, the network manager updates its internal model of the network and uses this information to generate the schedule and routes. Network performance and diagnostic information is also used by the network manager to adapt the overall network to changes in topology and communication

requirements. After the overall schedule has been generated, the schedule is transferred through a series of commands from the network manager to the network devices.

I.2.2 Initializing the network manager

The network manager is responsible for the network_ID, which is required for each specific wireless network. One possible way to initialize the network_ID is through a gateway that is hosting the network manager. It is done by connecting this gateway to a handheld or a host. The following parameters have to be configured to initialize the network manager:

network_ID, network manager nickname, network manager address, gateway nickname, gateway address and gateway Join_key.

NOTE The network manager and the gateway addresses and nicknames are fixed. These are specified by this standard.

I.2.3 Types of network devices

Network device behavior depends upon its type. Device types and their data models are shown in Figure I.2.

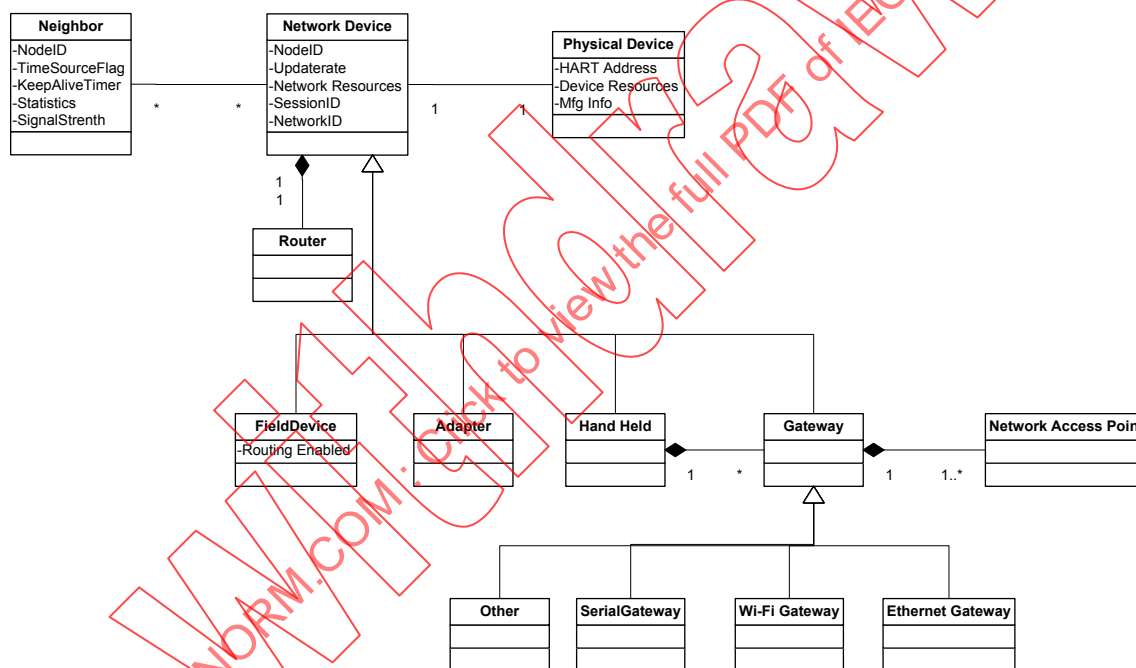


Figure I.2 – Device type data model

The root of the device data model is the network device. All network devices are globally identified by its 40-bit unique ID. The network manager contains a complete list of network devices connected to the network. It assigns a network unique 16-bit 'nickname' to each network device. All network devices also have attributes containing information on update rates, sessions and device resources, including items such as the size of the superframe table, etc. All network devices contain a list of neighbor devices that it has identified during its listening operations. Other attributes of a device depend upon the device type.

The most common type of network device is a field device. Field devices can be further differentiated by the type of measurement or control operation they. All field devices contain routing capabilities.

Handheld type of network device is used to configure devices, run diagnostics, perform calibrations, and manage network information inside other devices. If a handheld is used in a maintenance lab, it can connect directly to a field device via the maintenance port.

1.2.4 Network routing

```

classDiagram
    class NetworkManager {
        -NodeID
        -NetworkID
        -TopologyType {Mesh, Star, Star-Mesh}
        +Active()
    }
    class Route {
        <<abstract>>
    }
    class Graph {
        -GraphID
        -Destination Address
    }
    class Source {
        -RouteID
        -DestinationAddress
        -Timeout
        -NumSrcRouteHops
        -SrcRouteHops[4]
    }
    class Proxy {
        -64-bit Destination Address
    }
    class Neighbor {
        -NodeID
        -TimeSourceFlag
        -KeepAliveTimer
        -Statistics
        -SignalStrength
    }
    class ConnectionTable {
        -Connections
    }
    class Connection {
        -GraphId
        -NeighborAddress
    }
    class NetworkDevice {
        -NodeID
        -UpdateRate
        -Resources
        -SessionID
        -NetworkID
        -JoinKey
    }

    NetworkManager "1" *-- "*" Route
    NetworkManager "1" *-- "*" Neighbor
    NetworkManager "1" *-- "*" Graph
    NetworkManager "1" *-- "*" Source
    Route <|-- Graph
    Route <|-- Source
    Graph "1" *-- "*" ConnectionTable
    ConnectionTable "1" *-- "*" Connection
    Connection "1" *-- "*" NetworkDevice
    Source "1" *-- "*" NetworkDevice
    Proxy "1" *-- "*" NetworkDevice
  
```

The network manager contains a complete list of routes, connections and network devices. When a device is initially added to the network, the network manager stores all neighbor entries including signal strength information as reported from that network device. The network manager uses this information to build a complete network graph. The network graph is an optimized route map. During the optimization of the network graph, a large number of possible (but suboptimal) links are removed. The network graph is put together optimizing several attributes including reliability, hop count, reporting rates, power usage, and overall traffic flow. A key part of the topology is the list of connections that connect devices together.

One of the functions of the network manager is to configure graph and connection information in each network device. The network manager maintains a complete list of the graphs and connection information with which each network device has been configured. As the overall network adapts to changing information, the network manager is responsible for updating the overall topology, which includes adding and deleting information in each network device.

Using this knowledge, the network manager generates graph routes. It also generates source routes for the gateway. Source routes can be derived from graph routes assembled by the network manager. The graph route is like a tree. Single path from the root to the leaf is a source route. The network manager utilizes graph routes to generate schedules. This includes measurement information that is transferred from network devices to the gateway and control information that is transferred from the gateway to final control field device such as regulating valves, on-off valve, pump, fan, damper, as well as motors.

Every graph in a network is associated with a unique Graph_ID. To send a packet on a graph, the source network device includes a Graph_ID in the packet's network header. The packet travels along the paths corresponding to this Graph_ID until it reaches its destination, or is discarded. In order to be able to route packets along a graph, a device shall be configured with a connection table. The connection table contains entries that include the Graph_ID and neighbor address. Redundant paths may be setup by having more than one neighbor associated with the same Graph_ID. For routing a received packet, the device performs a lookup in the connection table by Graph_ID and sends that packet to any of the listed neighbors. After the neighbor acknowledges receipt of the packet (data-link level acknowledgement), the routing device may release it and remove the packet from the transmit buffer. If an acknowledgement is not received, the device attempts to retransmit the packet at its next available opportunity.

I.2.5 Network schedule

I.2.5.1 General

The network manager is responsible for allocating communication resources. The communication resources are divided up into slots and channel offsets and arranged in superframes. Communication resources are allocated based upon information from the communication requirements. The overall model for a network schedule is summarized in Figure I.4.

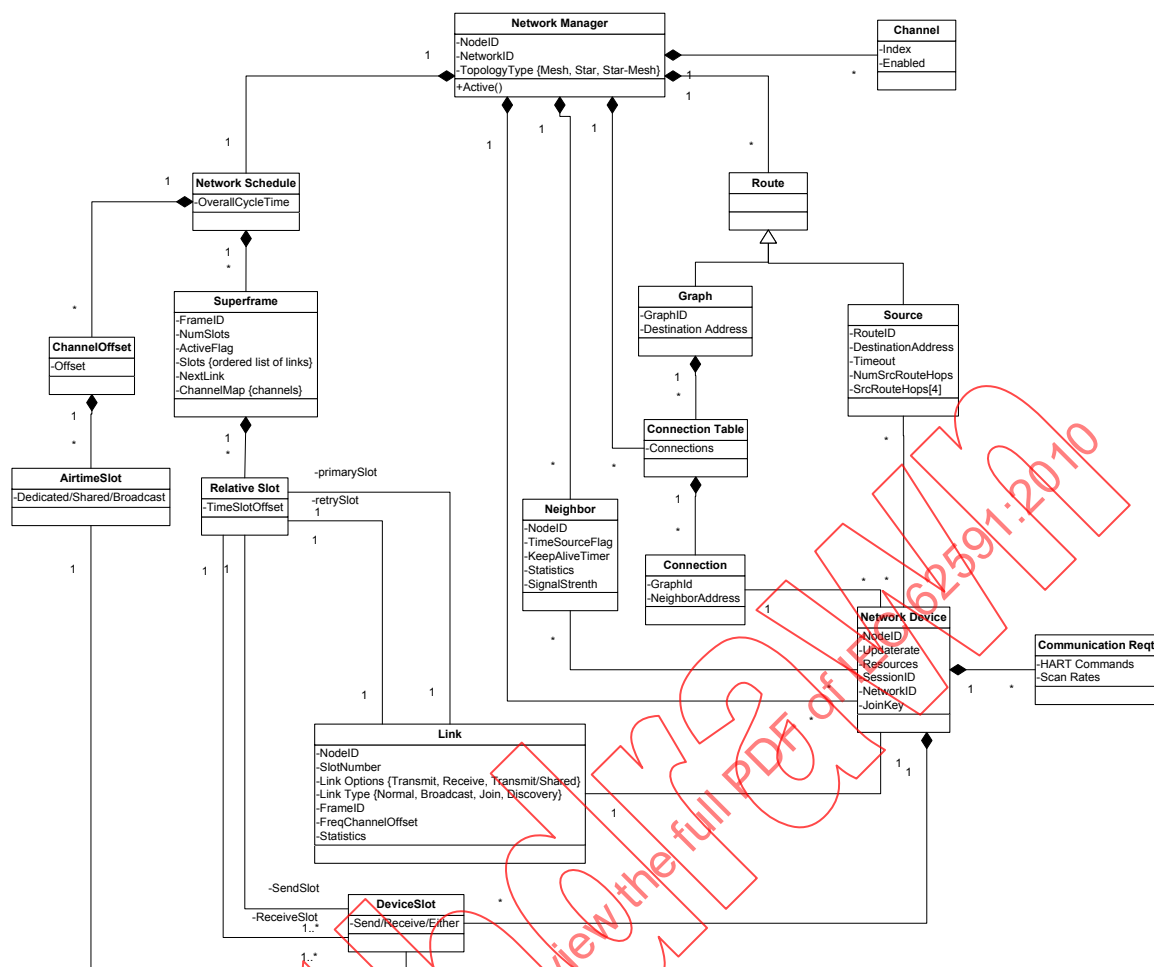


Figure I.4 – Network schedule

Each wireless network contains exactly one overall schedule that is created and managed by the network manager for that network. The schedule is organized into superframes. Each superframe is further subdivided into slots that repeat at the superframe cycle rate. The slots in a superframe are called relative slots; these slots are relative to the start of the superframe. Relative slot is not the same as absolute slot number which indicates the actual time that is being used for transmission of a specific packet.

Link objects are associated with specific network device ID; for each link there are slot allocations in one or more devices. If it is a dedicated slot then there is a send slot in one device and a receive slot in another device. If it is a shared slot then there is a receive slot in one device and one or more transmit slots in several devices. If it is a broadcast slot then there is one transmit slot in one device and receive slots in several devices. The link object also contains the Superframe_ID, relative slot number, link options (transmit, receive, shared), and Link_type (normal, broadcast, advertising, discovery).

The channel offset is used to calculate the specific radio frequency channel that is used for a particular slot based on a pseudorandom sequence.

The network manager combines the communication requirements with the superframe information to create a set of links for each device. The specific links loaded into each device are used by that device to determine when the device's radio needs to wake up, and after it wakes up whether it should transmit, receive, or either transmit and receive. The link does not determine what is communicated; a link is an opportunity to communicate. The device determines what it will communicate in each slot.

I.2.5.2 Superframe

A superframe is a collection of links assigned to time slots repeating in time. The number of slots in a given superframe (superframe size) determines how often each slot repeats, thus setting a communication schedule for devices that use the slots. A superframe is a product of both channels and time slots. When a superframe is created, it is associated with a Graph_ID. The network manager uses this association to allocate slots and configure links. In runtime the device determines how a link will be used.

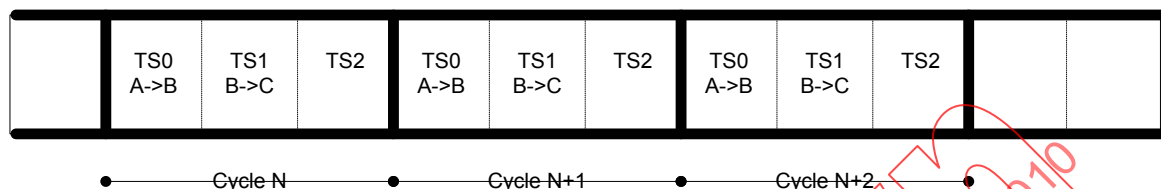


Figure I.5 – Example of a three-slot superframe

Every new superframe instance in time is called a superframe cycle. Figure I.5 shows how devices may communicate in a simple three slot superframe. Devices A and B communicate during slot 0; devices B and C communicate during slot 1; and slot 2 is not being used. Every three slots, the link schedule repeats.

The size of superframes should follow a harmonic chain, i.e. all periods should divide into each other. Examples of harmonic chains are 1, 2, 4, 8, 16, ... and 3, 6, 12, 24 and as well as any other period that conforms to the expression ab^n .

I.2.5.3 Multiple superframes in the network

A wireless network may contain several concurrent superframes of different sizes. Multiple superframes may be used to define a different communication schedule for various groups of devices or to run the entire network at different duty cycles. Additional superframes may also be allocated for different communication rates, publish data requirements, event notifications, and application layer commands issued by host applications.

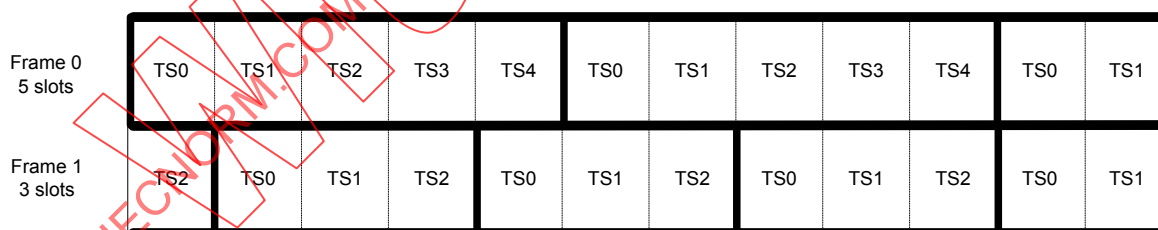


Figure I.6 – Multiple superframes in a network

A network device may participate in one or more superframes simultaneously, but not all devices need to participate in all superframes. By configuring a network device to participate in multiple overlapping superframes of different sizes, it is possible to establish different communication schedules and connectivity matrices that all work at the same time.

Key applications, such as asset management systems and device specific applications, often require considerable throughput for relatively short durations of time (measured in minutes) – used to call up configuration and diagnostic screens and respond to user requests. To support this temporary increase in demand for communication time slots, additional superframes may be used.

Superframes can be added, removed, activated and deactivated while the network is running. All superframes have identical start time. Cycle 0, slot 0 of every Superframe occurs at the beginning of epoch. The epoch for a specific network is the time when the network manager starts the network. Because of this, time slots in different superframes are always aligned, even though beginnings and ends of superframes may not be aligned, as shown in Figure I.6. Because all superframes begin at the same time, it is always possible to identify start time of a specific superframe cycle and time slot.

A network device with links in multiple superframes may encounter a link arbitration situation. This may happen when two or more superframes with assigned links coincide in the same absolute time slot. In these cases, the device shall operate on the link that has the numerically lowest frame ID. The rules for link arbitration are defined by this standard.

I.2.5.4 Time Slots

A time slot is a unit of fixed time duration commonly shared by all network devices in a network. The duration of a time slot is sufficient to send or receive one packet and an accompanying acknowledgement, including guard-band times for network-wide synchronization. The model also refers to a time slot as 'relative time slot'. This is because time slots in a superframe are always relative to the start of that superframe.

Dedicated links are shared by a pair of devices that communicate during an allocated time slot. Shared links can have more than one transmitter and only one receiver. For shared links, a defined back-off and retry mechanism handles collisions that may occur. Broadcast and multicast links have one transmitter and many receivers, but there is no data-link acknowledgement.

Time slots repeat in time at the rate corresponding to the size of its superframe. Time slots are assigned to devices through links. If a time slot is assigned to a device, the device can perform one of the following actions within the time slot, depending on the type of link:

- attempt to transmit a packet,
- wait to receive a packet, or
- remain idle.

A network device that has a transmit link or a transmit/receive link may send a packet during the associated time slot, if the destination of the packet matches the neighbor(s) on the other end of the link. A network device that has a receive link, or a transmit/receive link with no packet to send, listens for an incoming packet during the associated time slot. Time slots can also be shared by multiple devices. All devices that participate in either a dedicated or a shared link are awake and listening.

I.2.5.5 Links

When the network manager creates connections between devices, link assignments is available to support those connections. In many cases it will be necessary for the network manager to create new link assignments. In these cases the network manager will transfer the link assignments to each of the devices that require it. A link assignment specifies how the network device uses a time slot. Each link includes exactly one time slot, a channel offset, its type (transmit, receive or shared), neighbor device on the other end of the link and transmit/receive attributes.

Links may be transmit-only, receive-only, or transmit/receive. A network device that has a transmit link or a transmit/receive link may transmit a packet during the associated time slot, if the destination of the packet matches the neighbor(s) on the other end of the link. A network device that has a receive link, or a transmit/receive link with no packet to transmit, listens for an incoming packet during the associated time slot.

Transmit links may be shared by multiple devices, which is indicated to the network device by the shared flag in the link configuration. For a shared link, the device uses a collision-avoidance scheme with a Backoff to handle collision situations. Using shared links may be desirable when throughput requirements of devices are low, and/or traffic is irregular or traffic has bursts. In some situations, using shared links may decrease latency because the network device does not need to wait for dedicated links, but only if the probability of collision is low.

I.2.6 Security manager

The security manager is used by the network manager to allocate session keys. This is shown in Figure I.7. The network manager is responsible for propagating security keys.



Figure I.7 – Security manager

I.2.7 Detailed model of the network manager

I.2.7.1 General

The detailed model of the network manager is shown in Figure I.8 along with all of the other key components of the architecture. A memory buffer (Packet queue) and packet are also included in the diagram. These will be used to help walk through the model from the point-of-view of routing packets.

The left side of the drawing primarily deals with allocating communication resources. The right side deals with routing. The network manager uses routing to determine how to route packets and it uses communication requirements to determine what network resources shall be assigned. Using both of these, the network manager generates an overall network schedule, which in effect allocates communication resources in terms of superframes and links. After the schedule has been determined, the network manager transfers these configurations items to each network device by using application layer services such as write superframe, write link, write graph.

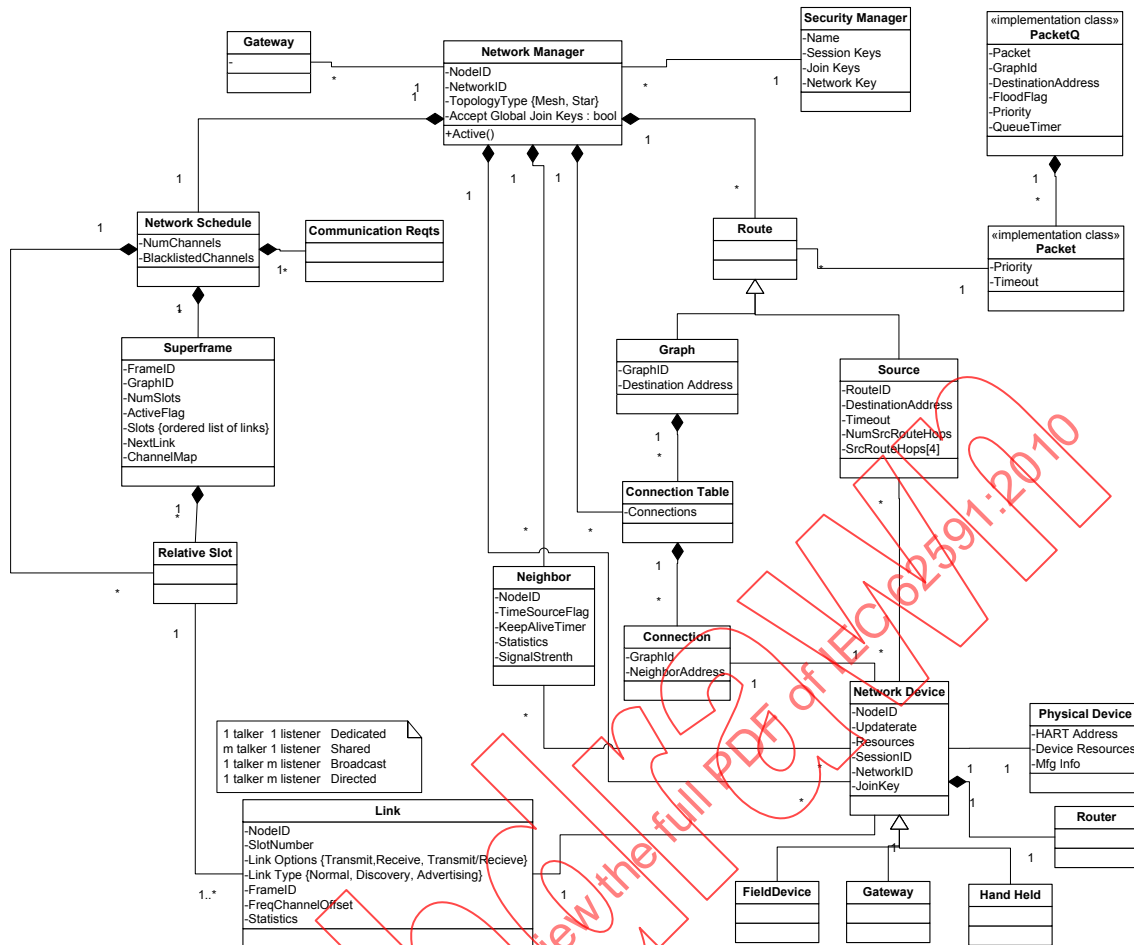


Figure I.8 – Detailed model of the network manager

When a device wakes up, it first looks at its list of links and selects the next link. If the link option is receive, the device immediately enters into a listen mode. If the link option is transmit or a transmit and receive, the device enters into transmit state. Since the link identifies which device can be transmitted to, the next step is to check if the device has a packet in its Packet queue that can be sent to the device on the other end of the Link. To find a match, the device looks at the routing information in each packet in the Packet queue. If a match is found then the packet is sent. If a match is not found and the Link_type is transmit/receive then the device listens for an incoming packet. If no match is found then the device is returned to sleep mode.

NOTE The above description is a simplified sequence of operation for illustrating how the link schedule is executed by a device.

I.2.7.2 Network addressing

When a network device joins the wireless network, the network manager assigns it a 16-bit address (nickname). The network header of every packet contains source and destination network device addresses, which do not change as packets are routed through the network. It is the responsibility of each network device to terminate packets in which the destination network device address matches the device's own address. Broadcast network device address may be used to address all network devices.

I.2.7.3 Network formation

A key attribute of a wireless network is its ability to self-organize. There are three components of network formation – advertising, joining and resource allocation. As part of advertising, network devices that are already part of the network send packets announcing the presence

of the network that they are part of. Advertisement packets include time synchronization information and a unique network_ID. A device trying to join the network listens for these packets and tries to match the advertised network_ID with their own. After at least one advertisement is heard, the new network device can attempt to join the network. The new network device is authenticated to join the network. After a device has joined the network it can request the network manager for allocating network resources.

I.3 Routing

I.3.1 General

A key part of the network manager's functions is to develop the overall routing for the network. In order to put together efficient and optimized routes the network manager needs information about the network, information about communication requirements, and information about the capabilities of the network devices. As this information is discovered, the network manager adjusts the connections in the network until it has a good working system. This subclause summarizes some of the key requirements and presents rules for determining routes.

I.3.2 Routing requirements

The requirements for the network manager are summarized in Table I.1.

Table I.1 – Routing requirements

Network function	Requirement
Create and manage network route	The network manager maintains an internal representation of the entire network, which in extreme cases could have every node connected to every other node. The network manager prunes the route information down into what it estimates as a reasonable representation of the network. This internal representation is used to generate graph and source routes.
Manage neighbor tables	The network manager collects network statistics and neighbor table information from each device through periodic health reports. The connection and signal level information is used to adjust the routes.
Health reports	The communication information is used to choose between existing connections and make decisions on forming new ones.
Build route tables for graph routing	Graph routing is ideal for both scheduled upstream and downstream communications. Upstream communications include process measurements and alarms. Downstream communications include control changes to actuators. There should not be any circular loops in any route
Builds source route lists for Source routing	The network manager builds the source routes. There should not be any circular loops in any route
Build downstream graph	The network manager generates a downstream broadcast and a unicast graph from the gateway to all other network devices.

I.3.3 Routing strategy

A basic routing strategy is summarized below.

- If there is a one hop path to the gateway it should be used.
- The minimum number of hops to be considered when constructing the graph is 2.
- The maximum number of hops to be considered when constructing the initial graph is 4.
- The ratio of the lowest signal strength on a two hop path to the signal strength on a corresponding one hop path for the two hop path should be considered instead of the one hop path.
- Use the same rule noted in d) for 3 and 4 hop paths.

- f) The signal level threshold to be used when building the graph should be 50% as a starting point. If no paths are found using the specified signal level threshold, then this threshold can be reduced to 0,75 of its previous value and the graph generation retried. This recursion should continue up to four times. If at least one route is still not possible, then it is considered that the node is unreachable.

NOTE The routing algorithm is not specified by this standard.

I.4 Scheduling

I.4.1 General

Scheduling communication resources is one of the important functions that the network manager performs. In order to put together efficient and optimized schedules, the network manager needs information about the network, information about communication requirements, and information about the capabilities of the network devices. As this information is discovered, the network manager continues to adjust the schedule until it has met the requirements. The scheduler in the network manager uses feedback from the operation of the system to tune the schedule. This subclause summarizes some of the key requirements, presents rules for determining routes, and outputs a simple example of a route.

I.4.2 Scheduling requirements

The requirements for the network manager are summarized in Table I.2.

Table I.2 – Scheduling requirements

Network function	Requirement
Assumptions	Network manager has reasonable representation of network graph. Each network device has been configured with a connection table. Network manager knows the update rate of each device. For redundancy, a datum is configured with one transmit and a retry on one path and another retry on another path.
Constraints	Maximum number of concurrent active channels is determined by the number of enabled channels and limited by black-listing. No device can be scheduled to listen twice in a slot. More than one device can transmit to the same device. A broadcast link and dedicated links to each of the listening devices can coexist. On multi-hop path, early hop are scheduled first. The supported update rates are defined as 2^n, where 'n' is a positive or a negative integer value. For example update rate selection can be 250 ms, 500 ms, 1 s, 2 s, 4 s, 8 s, 16 s, 32 s, 60 s or more. Base network management and publish data communications should not exceed 30% of the available communication bandwidth (100 slots/sec max). The network manager takes into account the service requirements. The final schedule (not counting the gateway) should have 50% free slots (i.e. allocated for retries, listens).
Data superframe	The data superframe length is determined by data scan rate. The network manager allocates slots starting with the fastest to the slowest scan rate. The network manager allocates one link for each en-route network device to the gateway, starting from the furthest end device. It allocates a 2nd dedicated slot on the same path to handle a retry. It allocates a 3rd shared slot on a separate path to handle another retry.
Management superframe	Management superframe has priority over data superframes. The network management should be 6 400 slots. The network manager traverses the graph by breath-first search, starting from the gateway, number the devices as $N_0, N_1, \dots, N_n$. At a minimum, every device shall have a slot for sending Keep-Alive DLPDU and there is a corresponding shared receive slot.
Join process	From the devices furthest from the gateway, the network manager allocates one link for each en-route network device to the gateway for join-request (no redundancy provided). Traversing the graph by breath-first search, the network manager allocates one link for each en-route network device from the gateway to the end network device for join-response (no redundancy provided). The network manager allocates advertise packets in each device. The number of advertise packets is inversely related to the number of hops away from the gateway.
Neighbor discovery	The network manager allocates discovery link common to all network devices. The discovery-interval timer is set to enable discovery.
Network management commands	It shares the network management links with join requests and responses.
Command request/response traffic	The network manager allocates shared slots to meet ad-hoc request and response traffic.
Gateway superframe	The gateway superframe should be allocated with a large ID value. The gateway superframe should be 40 slots long. All slots in the gateway's access points should be allocated. The network manager schedules all unallocated slots in the gateway. It alternates each of these slots as transmit, receive; receive slots are shared.

Network function	Requirement
Special purpose superframes	The network manager should allocate superframes to be used by gateway or client to address high throughput demand to satisfy asset management and other applications. This is allocated as a "maintenance" or "block transfer" service type. The network manager should allocate superframes to be used by the handheld and every field device for maintenance purpose. This superframe is used to provide the field device and the handheld with a high-speed connection. The network manager should allocate 4 slots per s (two links in each direction).

I.4.3 Scheduling strategy

A basic scheduling strategy is summarized below. The scheduling algorithm is not specified by this standard.

- 1) Scheduling strategy
 - a) Starting from slot 0, the devices are assigned to channel offsets.
 - b) Fastest publish data requirement is allocated first.
 - c) Publish data destination is always the gateway.
- 2) Data superframes
 - a) The data superframe length is determined by the data scan rate.
 - b) Slots are allocated starting with the fastest to the slowest scan rate.
 - c) Starting from the device furthest from the gateway, one link for each en-route network device to the gateway is allocated. A 2nd dedicated slot to handle a retry is also allocated.
 - d) Each transmission is also scheduled with a retry on another path, if one is available.
 - e) One network device can only be scheduled to receive once in a slot.
 - f) Event notification uses same slot allocation scheme as data. If there is publish data operation scheduled, then events can share the same slots. Events are sent infrequently. When event is sent, it can use the retry slots.
- 3) Management superframe
 - a) Management superframe has priority over data superframes.
 - b) The network management superframe should be 6 400 slots.
 - c) Advertisement slots are the slots that devices use to allow other devices trying to become part of the network to join through.
 - d) The graph should be traversed by breath-first search, starting from the gateway, number the devices as N0, N1, ... Nn.
 - e) Every device shall have a slot for a Keep-alive DLPDU. This slot should be shared receive slot on the parent side. If a network device has not sent any packet to its parent within Keep_alive_time interval, it sends a Keep-alive DLPDU after the expiration of that interval.
 - f) Each device shall have 3 slots every 15 minutes for health reports.
 - g) Each device shall have at least 1 shared slot every minute for request/response. If an asset management application is started, then the gateway shall allocate communication resources for that application.
- 4) Join request
 - a) Starting from the device furthest from the gateway, one link should be allocated for each en-route network device to the gateway (no redundancy provided).
- 5) Join response
 - a) One link should be allocated for each en-route network device from the gateway to end network device (no redundancy provided).

- 6) Network management commands and responses
 - a) The network management links should be shared with join requests and responses.
- 7) Command request/response
 - a) The links should be allocated in the same way as join requests.
- 8) Maintenance superframe
 - a) Slots for the maintenance superframe should be allocated. This same superframe is set-up in all devices. The superframe should be 1 s in duration and with 4 slots in it.
- 9) Gateway superframe
 - a) The gateway superframe has a Superframe_ID of 253.
 - b) The gateway superframe is a minimum of 40 slots long (400 ms).
 - c) The slots should be alternate transmit and receive and should all be shared.
- 10) Optimization parameters are:
 - a) number of hops to the gateway,
 - b) alternate paths,
 - c) latency,
 - d) power utilization and
 - e) overall throughput.

I.4.4 Networking scheduling example

A simple scheduling example is presented. This example covers a wireless network that consists of one gateway and three field devices as shown in Figure I.9. The gateway is identified as 'A' and the three field devices are identified as 'B', 'C', and 'D'. The field devices B and C communicate every second; field device 'D' communicates every 4 s.

Schedule

The scheduler first creates paths and then allocates communication bandwidth.

Step 1: Select path to the gateway

B: B->A

C: C->A

D: D->B->A, D->C->A

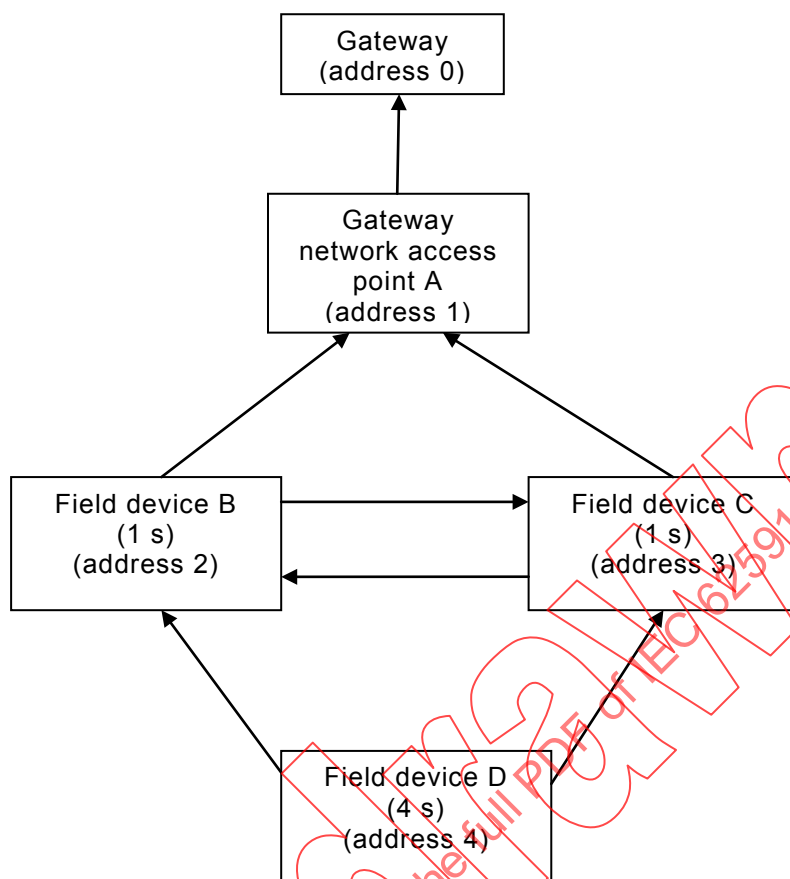


Figure I.9 – Four network device wireless network

Step 2: Data superframes – see Table I.3 and Table I.4.

Table I.3 – Frame ID 1: 1 s update rate, superframe length 100

Channel offset	Time Slot								
	0	1	2	3	4	5	6	7	8
0	B->A	B->A							
1			C->A	C->A					
2									

Table I.4 – Frame ID 4: 4 s update rate, superframe length 400

Channel offset	Time Slot								
	0	1	2	3	4	5	6	7	8
0			D->B	D->B	D->C				
1					B->A	B->A	C->A	C->A	
2									

Step 3: Management superframe – see Table I.5 to Table I.9.

Frame ID 0: Management superframe length 6 000 – once per minute.

Advertisement packets indicate the open links that can be used by joining devices.

Table I.5 – Frame ID 0: management superframe

[illegible]

Table I.6 – Join request (shared with management responses)

[illegible]

Table I.7 – Join response (shared with management requests)

[illegible]

Table 1.8 – Commands

[illegible]

Table I.9 – Command responses

[illegible]

Step 4: Create sub-schedules for each node (link table entries) – see Table I.10 to Table I.13.

Table I.10 – Device A

Frame ID	Time slot	Channel Offset	Device address	Link option	Link_type
1	0	0	B	Receive	Normal
1	1	0	B	Receive	Normal
1	2	1	C	Receive	Normal
1	3	1	C	Receive	Normal
4	4	1	B	Receive	Normal
4	5	1	B	Receive	Normal
4	6	1	C	Receive	Normal
4	7	1	C	Receive	Normal
0	8	0	*	Receive	Advertise
0	9	1	B	Receive	Normal
0	10	2	C	Receive	Normal
0	11	1	B	Receive	Normal
0	12	0	B	Transmit	Normal
0	13	0	C	Transmit	Normal
0	14	0	B	Transmit	Normal
0	16	2	C	Receive	Normal
0	17	1	B	Receive	Normal
0	18	1	B	Receive	Normal
0	19	0	B	Transmit	Normal
0	20	0	B	Transmit	Normal
0	21	0	C	Transmit	Normal

Table I.11 – Device B

Frame ID	Time slot	Channel Offset	Destination address	Link option	Link_type
1	0	0	A	Transmit	Normal
1	1	0	A	Transmit	Normal
4	2	0	D	Receive	Normal
4	3	0	D	Receive	Normal
4	4	1	A	Transmit	Normal
4	5	1	A	Transmit	Normal
0	6	1	*	Receive	Advertise
0	7	3	D	Receive	Normal
0	9	1	A	Transmit	Normal
0	11	1	A	Transmit	Normal
0	12	0	A	Receive	Normal
0	14	0	A	Receive	Normal
0	15	1	D	Transmit	Normal
0	16	3	D	Receive	Normal
0	17	1	A	Transmit	Normal
0	18	1	A	Transmit	Normal
0	19	0	A	Receive	Normal
0	20	0	A	Receive	Normal
0	21	1	D	Transmit	Normal

Table I.12 – Device C

Frame ID	Time slot	Channel Offset	Device address	Link option	Link_type
0	0	2	*	Receive	Advertise
1	2	1	A	Transmit	Normal
1	3	1	A	Transmit	Normal
4	4	0	D	Receive	Normal
4	6	1	A	Transmit	Normal
4	7	1	A	Transmit	Normal
0	10	2	A	Transmit	Normal
0	13	0	A	Receive	Normal
0	16	2	A	Transmit	Normal
0	21	0	A	Receive	Normal

Table I.13 – Device D

Frame ID	Time slot	Channel Offset	Device address	Link option	Link_type
0	0	3	*	Receive	Advertise
4	2	0	B	Transmit	Normal
4	3	0	B	Transmit	Normal
4	4	0	C	Transmit	Normal
0	7	3	B	Transmit	Normal
0	15	1	B	Receive	Normal
0	16	3	B	Transmit	Normal
0	21	1	B	Receive	Normal

I.5 Network manager interface

I.5.1 General

The network manager communicates to network devices through a series of application layer PDUs. The network manager uses AL services to read and write communication settings into network devices. For example the network manager can use these services to create frames and links in devices. In addition the network manager implements all of the data-link layer and the network layer configuration services.

Each service describes a related set of parameters and has an expected action associated with it. For example write, add, modify link contains the parameters needed to add or change a link. Delete link is used to remove a link from the link table in a network device. Several services may be combined into one transaction. For example the network manager could aggregate services for creating a superframe and a link.

The example in Figure I.10 shows two message sequences. In the first sequence, the network manager sends “read superframe” ALPDU requesting information on a specific superframe. The device responds by sending a response. In the second sequence, the network manager aggregates and sends APDUs to modify the superframe and add links. The network device responds by sending the aggregated results of the request APDUs.

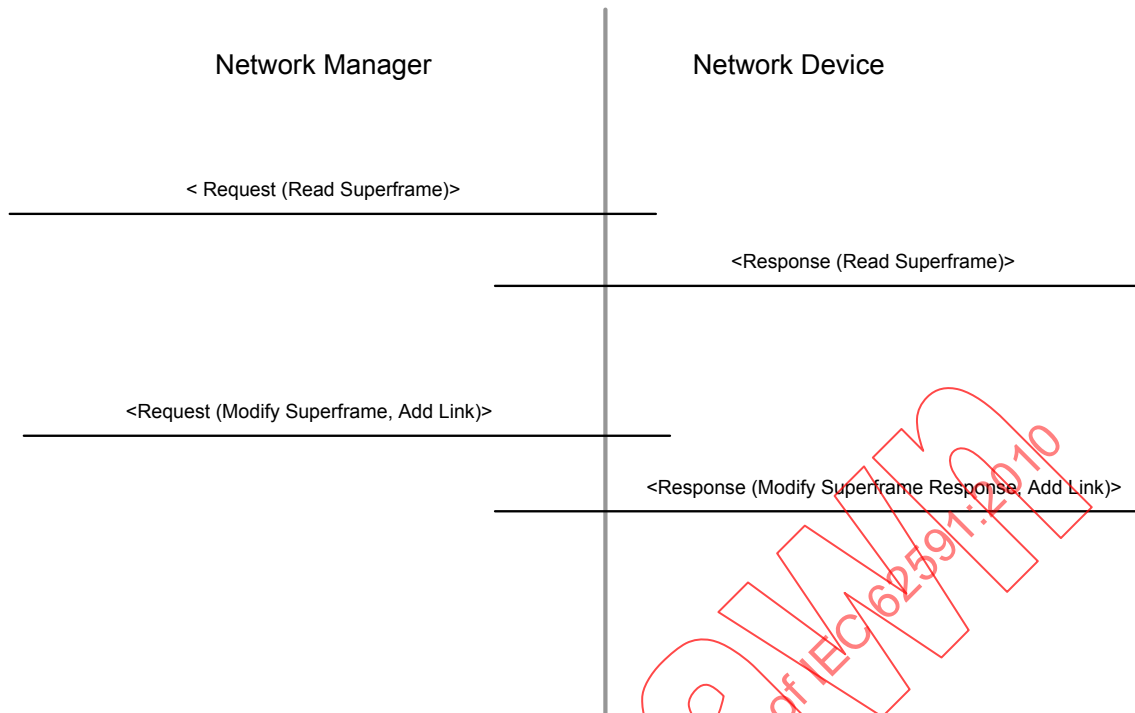


Figure I.10 – APDU sequences

Scenarios describing how to use the network manager interface are summarized in Table I.14.

Table I.14 – Network manager universal commands

Scenario	Description
Initializing a wireless network	Starting up a self-organizing wireless network
Allocating communication resources	Device sends a request to the network manager to increase its communication services.
Adjusting network schedule	Route is changed and schedule is updated.
Health reports	Device sends a health report to the network manager.
Path failure	A path failure is reported to the network manager.
Changing a session key	Network manager changes session keys.
Changing the network keys	Network manager changes the network keys.

I.5.2 Initializing a wireless network

A key characteristic of a wireless network is its ability to automatically start up and self-organize. Before a wireless network can form, a network manager and a gateway have to exist and both shall have created a private connection with each other. As part of its initialization sequence the network manager creates the following:

- network management superframe and
- network graph.

Then, the network manager activates the first superframe. This establishes the system epoch – ASN 0. The overall initialization sequence is shown in Figure I.11.

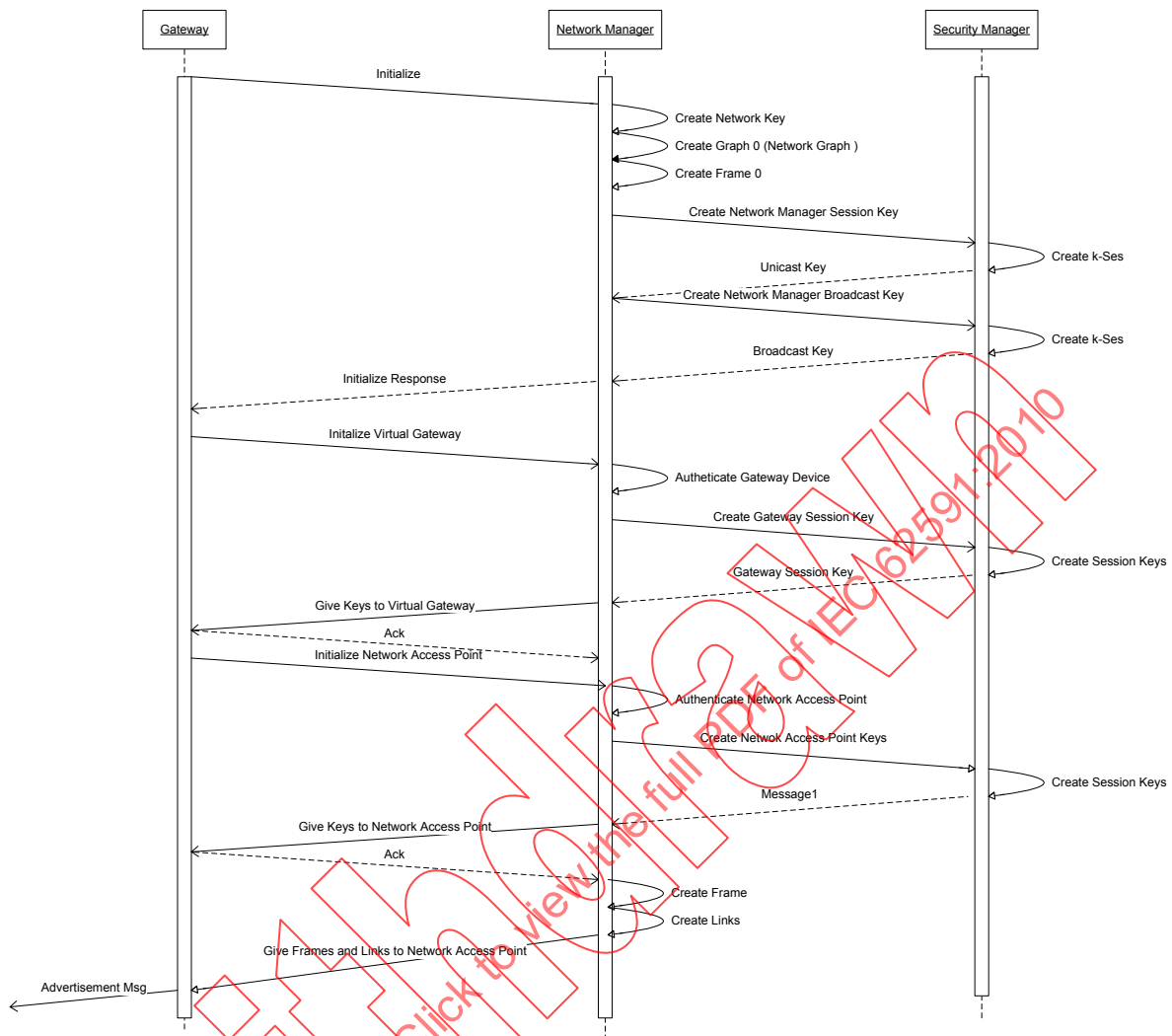


Figure I.11 – Initializing a wireless network

After the network access point starts to advertise, the other network devices can begin to join the network. As devices join, the network forms. There are three components of network formation: advertising, joining, and resource negotiation. As part of advertising, network devices that are already part of the network may send packets announcing the presence of the network. Advertisement packets include time synchronization information and a unique network_ID. A device that is trying to join listens for these packets. It then tries to match the advertised network_ID with network_ID set in it. After at least one Advertisement packet is heard, the new device can attempt to join the network. A new device joins the network by executing a join sequence.

I.5.3 Allocating communication resources

A device that joins the network does not start generating or receiving non-network-management data until an appropriate amount of network resource is allocated to it. For this purpose, 'Timetable' may be requested by a device, or may be created by the network manager and communicated to the device. A service is identified by service type (publish, block transfer, etc.).

To allocate resources for network service, the device sends write, add, modify request to the network manager. When the resources are allocated, the network manager returns a response code indicating what parameters were granted. If appropriate links and graphs

already exist in the network, the network manager may do nothing except send a response. Otherwise, additional links and/or graphs may be created or activated. This sequence is indicated in Figure I.12.

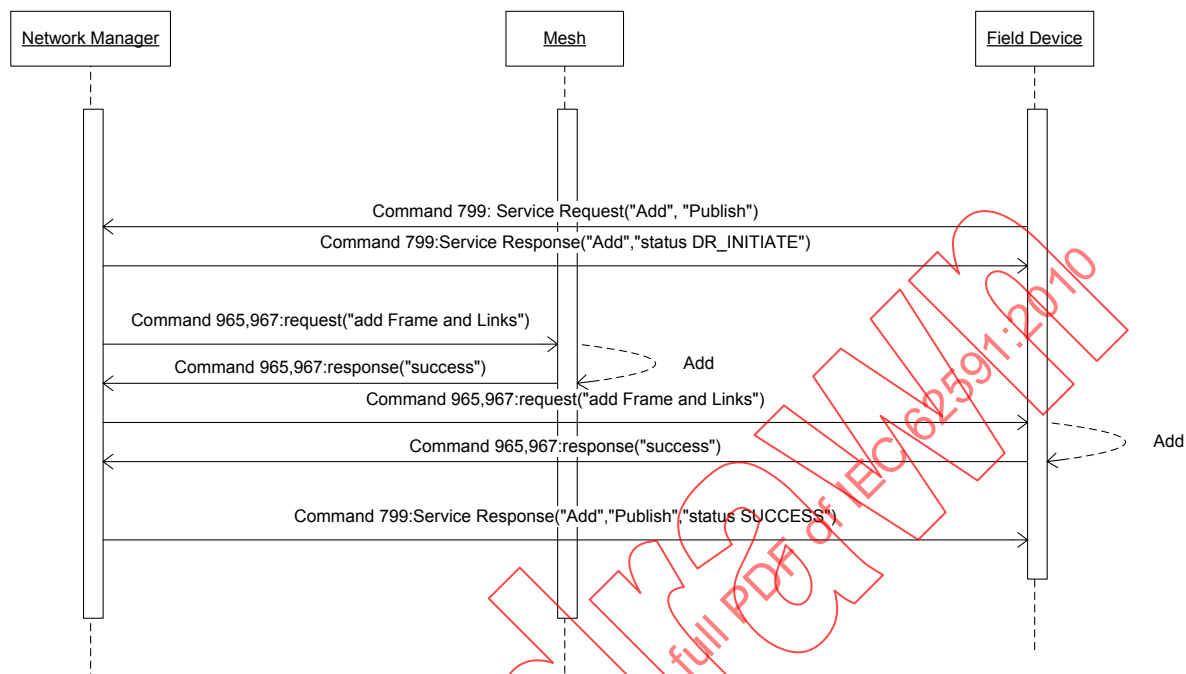


Figure I.12 – Allocating and using network resources

I.5.4 Adjusting network schedule

The overall network schedule is adjusted to address changes in routing, device resource usage (for example batteries running down), and network demand. This is illustrated in Figure I.13.

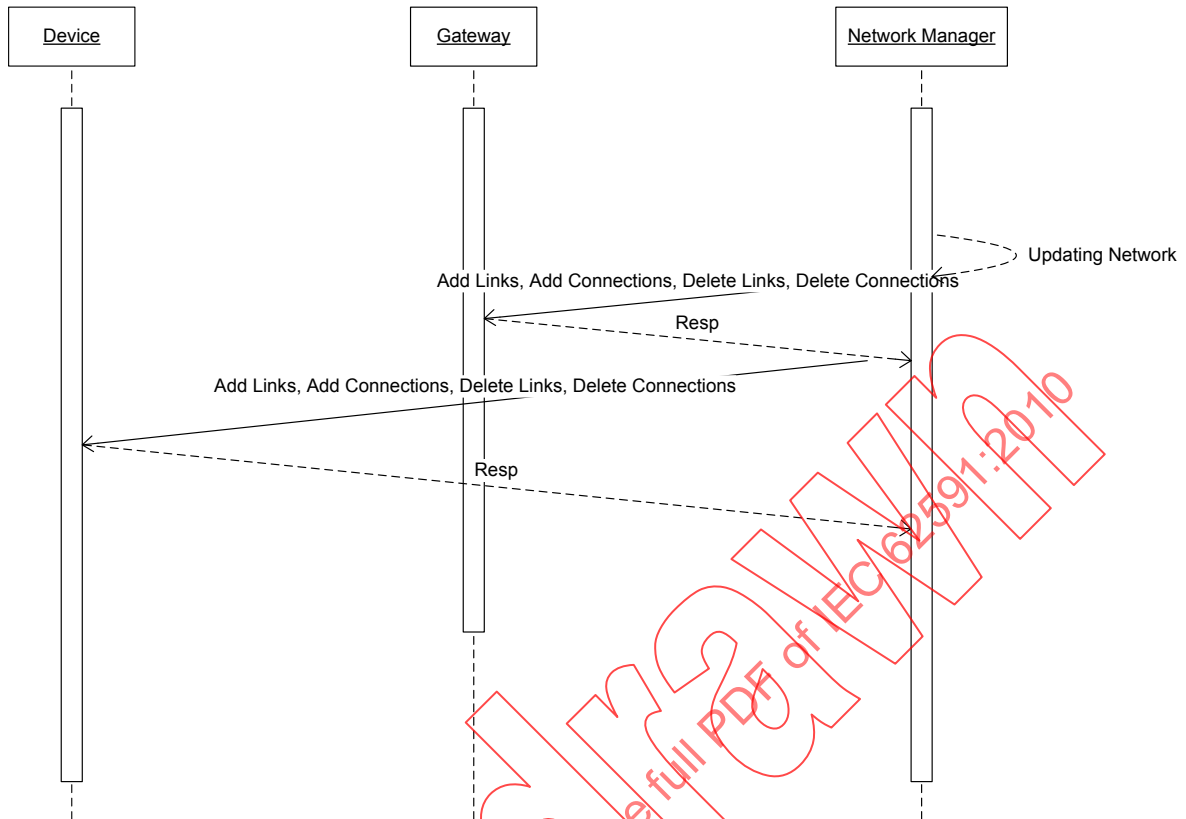


Figure I.13 – Adjusting network schedule

I.5.5 Health reports

Health reports are transferred from network devices to the network manager periodically. These messages are not acknowledged. This is illustrated in Figure I.14. Schedule and network optimization should be performed periodically as well.

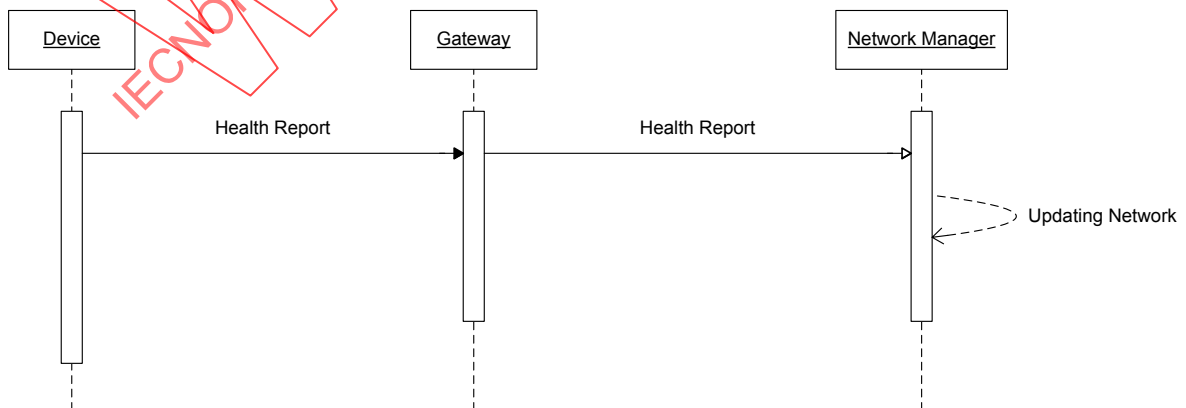


Figure I.14 – Health reports

Annex J (informative)

Gateway implementation

J.1 Overview

Gateway device connects the wireless network to a plant automation network, allowing data to flow between the two networks. It provides host applications access to the network devices. The functions to be performed by a gateway in wireless network are:

- The gateway implements a data cache to optimize the overall performance of the network and improve responsiveness to host applications.
- The gateway publishes device variable data that it is caching to other devices in the network.
- The gateway sends acknowledgement to the event notification APDU received by it.
- The gateway provides the tunneling function to transfer the APDU of this standard between the host connected to the gateway and the network connected to the gateway.
- The gateway checks the destination address in the APDU received from the host application. If the destination is the gateway itself, then it sends the response APDU as required by the AL protocol.
- If the destination address in the request APDU from the host application is for another network device, and if the response APDU is cached and is current, the gateway returns the cached response APDU from its real-time database.

The gateway provides a sink or source point for wireless network traffic. The network access point provides the physical connection into the wireless network.

J.2 Gateway model

J.2.1 General

The gateway has several distinct components as shown in Figure J.1. The gateway itself is a type of network device. It makes use of services from the network manager and the security manager to authenticate and join network devices.

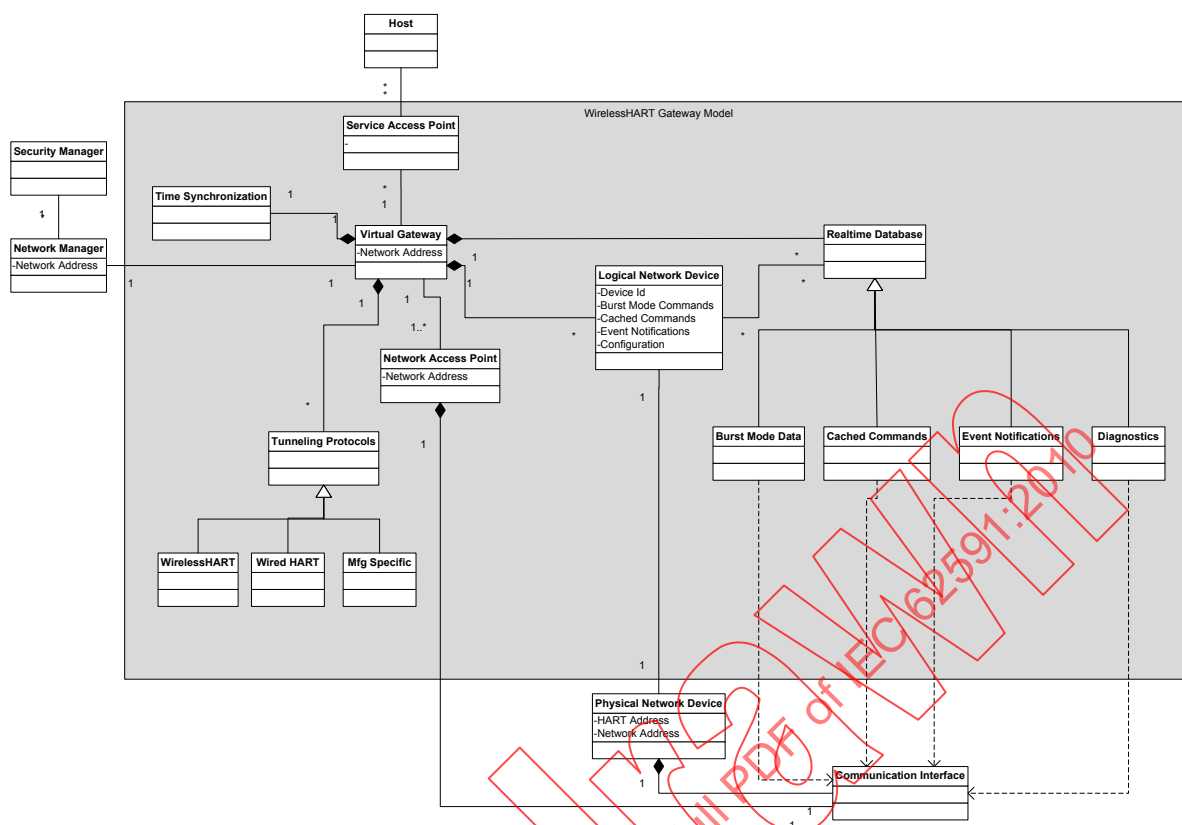


Figure J.1 – General model for gateway

J.2.2 Host interface

The gateway host interface is used to connect client applications outside of the wireless network with the devices connected to the wireless network. The gateway host interface can take many forms. Some of the common ones include:

- Ethernet-to-wireless gateway — a gateway that provides a bidirectional path between industrial Ethernet network and the wireless network.
- Wi-Fi-to-wireless gateway — a variation of an Ethernet-to-wireless gateway that uses IEEE 802.11 a/b/g radio to connect to the plant's network.
- Serial-to-wireless gateway — if plant automation servers and equipment support serial interfaces, a serial-to-wireless gateway can be used to connect to the serial interfaces of these devices.

The gateway is able to cache publish data command responses, commonly used read and write commands, and diagnostics data from the network devices and wireless adapters connected to the network. To take advantage of this caching, the gateway also behaves as a translator. The gateway peeks at requests from the host applications and, if the response data is cached and is current, returns the cached response messages from its real-time database. For example if a host issues HART™ command #0 request to a device in the network, the gateway checks to see if it has a cached command #0 response. If the gateway does not have the response for that device, it forwards the command to the device and returns the resulting response to the host.

J.2.3 Logical network device

The gateway maintains a list of logical network devices. The device information cached in the gateways in turn contains information about the devices such as the list of publish data commands, event notification messages, and cached commands responses that are currently stored there. The logical network device is shown in Figure J.2.

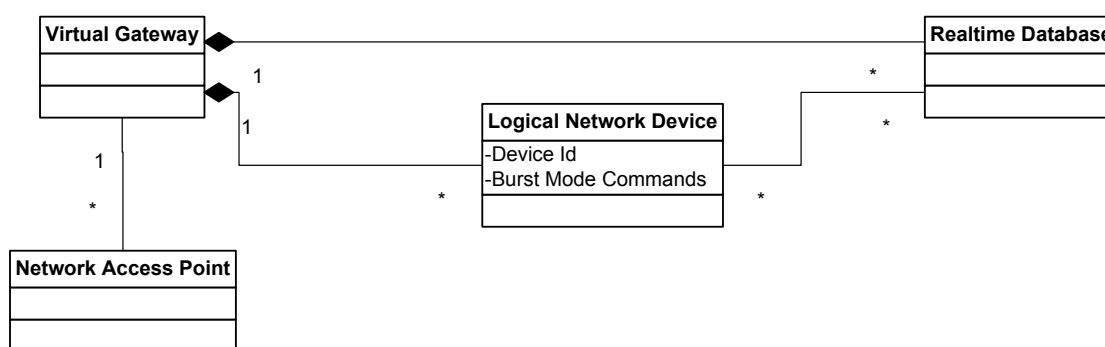


Figure J.2 – Logical network device

The logical network device plays an important role in modeling the system and later in commissioning the system. A logical network device can exist independent of an actual physical network device. It provides a network placeholder that can be used for off-line configuration, simulation, and on-line operation. It also provides the necessary separation for commissioning and device replacement.

J.2.4 Physical network device

A physical network device is an actual device in the network. It discovers neighbors and builds and maintains a neighbor table. This neighbor table is built by the physical device by listening for a specified amount of time on each channel. Each neighbor is recorded with its corresponding received signal strength. This association is shown Figure J.3.

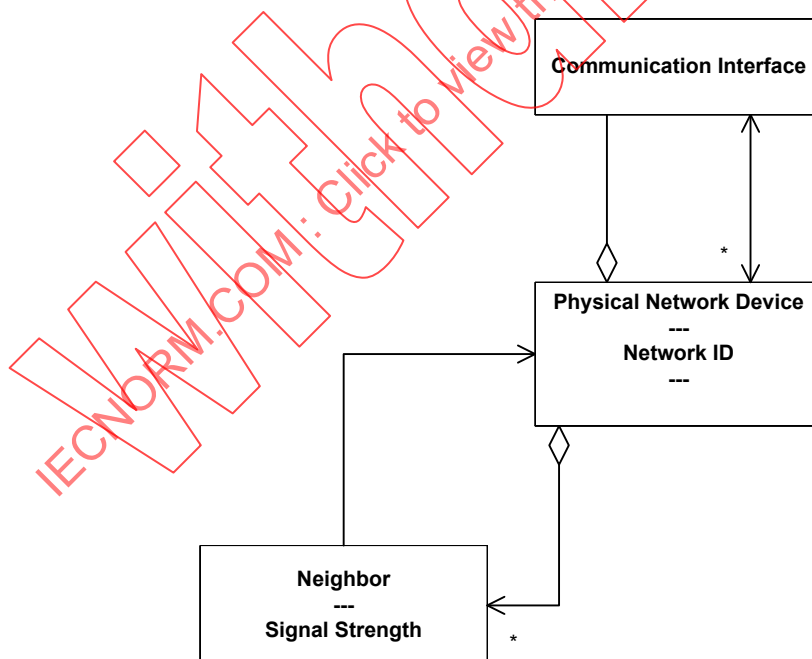


Figure J.3 – Physical network device

J.2.5 Communication interface

J.2.5.1 General

The communication interface provides the required services to communicate with network devices. In the implementation, these communication services are provided by the transport and network layers. The service types are summarized below:

- Request and response,
- Block data transfers,
- Publish data messages,
- Event notification, and
- Diagnostics.

J.2.5.2 Request and response

The request and response service provides a method for the network manager and host applications to send request messages to a network device and to receive its response message. If the response takes longer than a specified value, the gateway sends either a "delayed response" or a "timeout" response to the host application. In all cases, the host application expects a confirmed response message from the network device.

The network manager considers these communications as unscheduled and ad-hoc. Therefore, the network manager makes a base of bandwidth is available to accommodate request and response traffic. The gateway is responsible for requesting resources for this service, increasing and decreasing the bandwidth it requests as application demands vary.

J.2.5.3 Block transfers

Block transfers are supported both to the network devices and through an adapter to its sub-device. This supports transmission of large data sets with automatic segmentation and re-assembly the destination.

J.2.5.4 Publish data messages

The publish data messages are used to send data on a periodic schedule or on an exception basis. A publish data message consists of the response packet for the specified command. The publish data rate is configurable, based on application needs. For control applications, the publish data rate is determined by control loop or sequence execution requirements (for example the process time constant). In some cases, more than one host application will subscribe to the same published data. In these cases, the update rate is determined by the fastest requested data rate.

Publish data responses are cached inside the gateway in a real-time database. This provides a mechanism for multiple readers to access the same data; decouples the applications; and reduces network traffic.

J.2.5.5 Event notification

The event notification services provide support for the communications of events and alarms. This is a guaranteed message delivery service. The gateway is responsible for acknowledging and caching event notification messages.

J.2.5.6 Diagnostics

This service is used to transfer diagnostic information about the network to the host.

J.2.6 Cached response messages

J.2.6.1 Network status

Each network device maintains diagnostics. The diagnostics are periodically published via HART™ commands to the network manager. The network manager maintains the complete set of device and network diagnostics. Hosts can query the gateway or the network manager for network level diagnostics.

J.2.6.2 Publish data command response

The database caches all of the publish data command response messages.

J.2.6.3 Event notification command response

The database caches all of the event notification response messages.

J.2.6.4 Cached command response

The database caches the latest response messages for several HART™ commands.

J.2.6.5 Delayed response command response

For HART™ request and response commands, the gateway maintains a complete list of all outstanding commands that have been sent to devices for which a response has not yet been received in return. Delayed response commands are purged if they exceed a 24 hour timeout.

J.3 Gateway management

J.3.1 Addressing

At the wireless gateway and the host, each field device is identified by its 40-bit unique ID. In addition, the network manager maintains a unique 16-bit address (nickname) for each network device. Some devices, for example those connected through an adapter, do not have a nickname. In all cases the gateway maintains a table of 40-bit unique IDs, nicknames and device type information. This table is used by the gateway to translate addresses between host or clients and the network.

J.3.2 Retry mechanisms

Result of the host application request to the gateway may be classified as one of the following.

- Success – a valid response message is generated within the prescribed timeout window.
- Busy – the gateway received the request but is unable to respond at the present time.
- Delayed response – the gateway received the request and has started processing it.
- Error – the gateway received the message but detected an error that prevents it from being processed.

The number of times that a gateway retries before it returns Busy or an Error can be configured.

J.3.3 Power-on-reset

The gateway performs the following sequence when it is powered on.

- 1) Calculate the power outage time. If the power outage time is less than 15 minutes, sets an internal "WARM_START" flag. If the power outage time is greater than 15 minutes, sets an internal "COLD_START" flag. If power outage time cannot be determined, sets an internal "COLD_START" flag.
- 2) Looks for network access points and forms connections with it.
- 3) If a network manager is found with the same network_ID as configured in the gateway, it connects to that network manager.
- 4) Synchronizes its clock with an external time source such as a GPS receiver. The gateway will synchronize its clock with the external time source and, at least once per hour, broadcast a UTC time messages.

NOTE This is a separate mechanism from Keep-alive DLPDUs, which are used to keep the networks understanding of the ASN in-sync.

- 5) Initialize host interfaces - the gateway can now begin returning information to host applications.
- 6) If it is "WARM_START" then following sequence is executed.
 - a) Discovers the networks absolute slot number (ASN).
 - b) Checks the connections to all network devices in the graph table by sending commands 0 and 20 to each of the devices.
 - c) For each adapter, it executes the "Join sequence for adapter sub-devices". Updates the devices status to match the response returned from HART™ command #75 "Poll sub-device".
- 7) Commences normal operations.

J.3.4 Network access point reset

When a network access point is reset, it completely clears its memory and restarts. It looks for the gateway. After it has found the gateway, it joins the network through the gateway and then begins looking for neighbors on the network side.

NOTE The network access point can be reset by sending HART™ command #42 to it.

J.3.5 Gateway reset

Resetting the gateway is treated the same as its "COLD_START". The gateway sets its internal gateway reset flag, clears all of its buffers, and informs the network manager that it is being reset. The gateway reset flag is not cleared until network communications have been re-established. While the reset sequence is underway, the gateway responds to all commands to field devices with "Busy" code. Any existing delayed responses and block mode transfers are cleared and no new ones are accepted until the reset sequence has been completed. The following summarizes the actions taken by the gateway.

- 1) Sets reset flag.
- 2) Deletes any outstanding block mode transfers and sends back error response.
- 3) Deletes any outstanding delayed response and sends back error response.
- 4) Sends a HART™ command '03xxx reset device' to the network manager to inform the network manager that the gateway is being reset.
- 5) Clears all tables and buffers.
- 6) Initialize host interfaces.
- 7) Re-connects with the network manager.
- 8) Gets re-configured by the network manager.
- 9) Sends HART™ command #0 and command #20 to each device to build the device information cache.
- 10) For each adapter, executes the "Join sequence for adapter sub-devices".
- 11) Re-establishes periodic update messages with all devices.
- 12) Clear reset flag.

NOTE The gateway can be reset by sending HART™ command #42 to it.

J.3.6 Re-build publish data periodic data

When a gateway's REBUILD flag is set, all cached responses for a specified device are cleared. In the case of a wireless adapter, this implies that all buffers for all devices connected to that adapter will also be cleared. The gateway can use HART™ command #0 to test the connection to any network device.

J.3.7 Gateway self test

The Gateway may do the following after it receives HART™ command #41 (self test) from the host.

- 1) Verify the ROM checksum and set a flag, if this check fails.
- 2) Verify the non-volatile memory contents and set a flag, if this check fails.
- 3) Send HART™ command #41 response message

These ROM check and non-volatile memory verification functions may also be performed periodically by the gateway.

J.3.8 Adding new network device

Whenever the gateway receives a new network device in its route table from the network manager, the gateway sends that device a Command 0 and Command 20 requests. The response messages from the just added device are cached for that device in the gateway. The gateway can determine if the responding device is a wireless adapters (protocol bridge device), by examining bit 2 (protocol bridge device) of octet number 8 of the response.

J.3.9 Device configuration change status notifications

Whenever the gateway receives a configuration change status from a device, it sends HART™ command #0 and command #20 requests to that device. The response messages are then cached for that device in the gateway. The configuration change counter is stored as part of the command #0 response. If the device is a wireless adapter, then the gateway repeats the "Join sequence for adapter sub-devices" connected to the wireless adapter.

J.4 Gateway superframe

The gateway communicates with the network devices via its access points. The access point should have activity scheduled for every slot. Not utilizing every slot represents wasted opportunities. For example if the access points have nothing else to send or receive, it should advertise and perform shared listens.

The network manager should assign unused access points to advertise faster than ChannelSearchTime. In doing so, devices within the signal range of the access point and trying to join will be able to do so in shorter time.

Generally, a dedicated superframe should be assigned by the network manager (for example superframe number 253). By allocating a high-numbered Superframe_ID, other transmit and receive links can be used to transmit or receive higher priority messages.

J.5 Gateway change notification services

The gateway can return change notification messages to the host or the client, when a change in a network device is detected. The notification provides an indication that a value or status has changed, but does not include the actual changed value(s). When the host or the client receives a change notification, it is expected to issue a read the associated information request to the gateway.

NOTE Not all host interfaces can support change notifications.

The Table J.1 summarizes items for which the host or the client can receive change notification.

Table J.1 – Required command responses

Notification type	Fastest notification rate in s	Description
Publish data	0,250	The gateway checks the cache on a device-by-device basis for publish data changes (for example response to HART™ command #9). If there is any change in the data value, an item is added to the change notification list.
Event notification	1	Every 1 s the gateway checks the cache on a device-by-device basis for event notification updates. If there is any update, an item is added to the change notification list.
Application process status	5	Every 5 s the gateway checks the cache on a device-by-device basis for application process status changes. If there any change in the application process status, an item is added to the change notification list. If publish data is enabled, the rate will be increased up to the publish data rate, because the application process status is always sent with the publish data message.
Device configuration	60	Every 60 s the gateway checks the cache on a device-by-device basis for device configuration changes (monitors HART™ command #0 and command #20 responses; checks configuration change bit and configuration change counter). If there is any change in the device configuration, an item is added to the change notification list. Since this can also be handled with each publish data, the notification should only be sent if no publish data message has been sent.
Network topology	60	Every 60 s the gateway checks with the network manager to find if there has been any network topology change. If there is any change in the network topology, an item is added to the change notification list.
Network schedule	60	Every 60 s the gateway checks with the network manager to find if there has been any network schedule change. If there is any change in the network schedule, an item is added to the change notification list.

The Gateway supports at least eight (32 recommended) separate change notification lists. These change notification lists are for host-side support and there is separate list for every client.

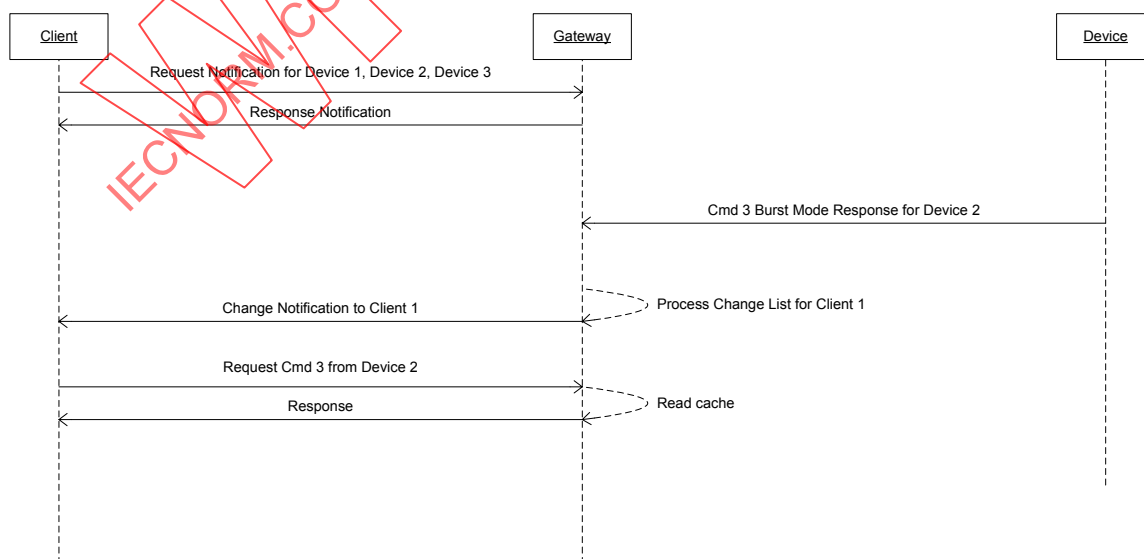


Figure J.4 – Managing notification services

Figure J.4 shows the sequence used by a client to request the gateway to monitor several devices. The overall sequence occurs in the following order.

- 1) Client-1 requests change notification messages for three devices – device-1, device-2, and device-3. The client does this by sending HART™ command #140 request to the gateway.
- 2) The gateway grants the request and sets up a change notification service for Client-1. The gateway returns a HART™ command response message indicating that the change notification request was successful.
- 3) Device-2 sends a HART™ command #3 response using publish data update. The gateway caches this response message for device-2 and sets the change notification bits for Device-2.
- 4) The gateway processes the change notification list for client-1. It adds HART™ command 3 for device-2 to its change notification list; sends a HART™ command #142 to client-1 indicating the changes that have occurred. It then clears the change notification bits for client-1.
- 5) Client-1 receives the change notification message from the gateway and reads the cached HART™ command #3 response from the gateway's cache.

J.6 HART™ commands interface

J.6.1 General

The gateway provides an interface between the host applications and the wireless network. The gateway:

- always uses extended addressing (except command #0);
- caches publish data response message data and "Application process status" data;
- caches selected HART™ command read and write messages;
- caches event notification response messages;
- supports client-side service requests for request and response, change notification, publish data, block mode transfers, delayed response messages and high throughput services;
- passes through command requests and responses addressed to and from devices in the network;
- automatically requests increments and decrements in communication bandwidth between the gateway and field devices in response to host application demands.

The gateway retains its configuration during reset and power failure, but does not retain the network schedule. The retained parameters include, but are not limited to:

- unique ID and nickname for all access points,
- network_ID,
- Join_key,
- tag, descriptor and date, and
- retry and timeout limits for busy and other errors.

J.6.2 Host to gateway command request and response

The HART™ commands from a host addressed to the gateway are:

- command #0 – read unique ID in short or extended format provided the address matches that of the gateway in either case,
- command #11 – read unique ID associated with tag, and command #21 – read unique ID associated with Long_tag, if the tag and address match those of the gateway,

- all other commands in extended 40-bit addressing format if the address matches that of the gateway,

If a communications error is detected, the gateway responds with a communication error response.

If a command number is unknown, the gateway responds with “command not implemented”.

During active reset operation, the gateway responds with “busy”.

J.6.3 Pass-through of HART™ command request and response

Pass-through service allows a HART™ command request from a host to a specific field device to pass-through the gateway and then through the network to the destination device and for the command response to return through the network to the gateway and pass-thru to the host. Following rules apply for this pass-through.

- All commands have to be in extended 40-bit addressing format.
- If the command matches one in the gateway cache, the gateway returns the cached response message.
- If the response is not cached, the gateway forwards the request to the field device. When the gateway receives the response, it returns the response from the field device to the host and if it is one of the commands that the gateway is expected to cache, then it caches the response.
- The host can flush completed delayed responses.
- If a communications error is detected, the gateway responds to the host on the device's behalf with the appropriate communication error response.

J.6.4 Caching publish data command response messages

The dynamic data cache in the gateway updates and records each time a device sends publish data message response including the data validity flags, byte count, response bytes, data bytes, total numbers of updates and the extended status bytes.

J.6.5 Gateway status error flag bits

The gateway supports the error flag bits shown in Table J.2.

Table J.2 – Gateway status flags

Error	Description
Cold-start	It is set when the gateway is powered-up or reset and the amount of time the gateway has been unavailable is more than 15 minutes. It is cleared when the gateway completes its start-up sequence.
Warm-start	It is set when the gateway is powered-up and the amount of time the gateway has been unavailable is less than 15 minutes. It is cleared when the gateway completes its start-up sequence. This shall be an internal state in the gateway.
Configuration-changed	It is set whenever the gateway parameters are changed by the host and it is cleared by receiving HART™ command #38 request.
Malfunction	OR of HARD_FAULT bits (see HART™ command #48 and variable definition).
More status available	HART™ command #48 should be read to get the status.

J.6.6 Gateway additional status flags

In responding to HART™ command #48, the gateway only reports the current state of its various flags.

J.6.7 Gateway capacities

The gateway also provides certain capacity to support devices. The guidelines for the capacities are summarized in Table J.3.

Table J.3 – Gateway capacity guidelines

Parameter	Tiny gateway (10 devices)	Small gateway (50 devices)	Large gateway (250 devices)
Minimum Number of Sessions	30	110	510
Minimum Number of Transport	30	110	510
Minimum Number of Route	15	60	128
Minimum Number of Neighbors	12	50	128
Minimum Number of Superframes	12	12	12
Minimum Number of Links	50	100	500
Minimum Number of Graphs	25	60	128
Minimum Number of Graph-Neighbor	40	200	1 000
Minimum Number of Packet Buffers	25	100	500
Minimum Number of Publish Data Messages	30	200	1 000
Minimum Number of Cached Messages	75	400	2 000
Minimum Number of Publish Event Messages	25	100	500
Total Number of Devices (Wireless Adapters + Field Devices)	10	50	250
Clients	4	8	32

J.6.8 Gateway commands

The gateway supports several gateway specific commands, supports pass-through messages, supports publish data and block mode transfers and caches several read and write commands. The commands are specified in 9.3.5.

Annex K (informative)

Handheld device

K.1 Overview

Handheld device is used in the installation and maintenance of a network device. It is portable equipment operated by the plant personnel. There are four approaches to connect a handheld device to the network device.

- a) Handheld device is connected through a plant automation network via another network such as Ethernet or wireless. This handheld device communicates with the network devices via the gateway device in the same way as external plant automation client does. It appears as another host application connected to the network.
- b) Handheld device is connected to the maintenance port in the network device. The handheld can communicate with the attached network device, but it cannot communicate with any other device in the network.
- c) Wireless network capable handheld device is connected to the network. The handheld device appears as another device in the network. In this mode the handheld can only communicate with the gateway and the network manager. This mode is used to write keys into the handheld device and to view diagnostic and system health information of the network. This mode of operation is referred to as "connected as a network device."
- d) Wireless network capable handheld device is connected to a field device via the network. The handheld device is restricted to communicate only with that field device. Special provisioning is used to ensure that the handheld is restricted to one hop and one device at a time. This mode of operation is referred to as "connected as a maintenance device."

K.2 Maintenance port connection

The handheld device can configure and interrogate the network device when connected to the maintenance port of that device. The field device allows reading of the device parameters and writing of parameters required for provisioning via the maintenance port. Any write request that can only be sent by the network manager is not available via the maintenance port. When connected via the maintenance port the handheld does not have access to the wireless network.

K.3 Network device connection

K.3.1 General

There are two reasons to connect a handheld device into the wireless network as a network device:

- a) to install session keys, and
- b) to view network diagnostics and health reports.

To join a wireless network as a network device, the user will need to first install the network_ID and the Join_key into the handheld itself. This can be done in one of several ways, including connecting the handheld up to the gateway via Ethernet to initialize the network and to allocate the initial keys for the handheld. The initializing of the network and the gateway are implementation specific. After the handheld has its network_ID and Join_key, it will join the network in the same way as any other device joins the network.

K.3.2 Install session keys

The handheld will need a session key to communicate with another network device. After joining to the network, the handheld has to send a request to the network manager to allocate session keys. There is separate session key for each device that the handheld will communicate with. The network manager installs these keys into the handheld and the devices. The session keys and their nonce counters are initialized to the same number at creation.

K.3.3 View network diagnostics and health reports

After the handheld has connected itself to the wireless network as a network device, it can request the network manager to allocate it additional communication resources to communicate with the gateway. The handheld can send requests to the gateway and to the network manager to read diagnostic and health report information. The handheld will not be able to communicate with other devices on the network when connected in this manner (device-to-device connections are not allowed).

K.4 Network connection as a maintenance device

After a network is up and running, a maintenance technicians may want to connect wireless network capable handheld device to an individual installed field device via the network to gather device specific information, run diagnostics, check device calibration, etc. To support these activities, the network manager will allow the handheld to connect up to one device at a time using a special superframe that has been installed into each device. The sequence for connecting up to the field device is as follows.

- a) The handheld is connected to the network either directly or through the gateway and it obtains sessions (including the keys) to communicate with the selected network device.
- b) The human user walks out into the plant and gets close to the selected network device.
- c) Handheld goes into the listen mode looking for device advertisement packets.
- d) Handheld locates the selected device and identifies links on which it can communicate with the selected device. It sends a command to this device requesting additional bandwidth, using one of these links.
- e) The device sends a request to the network manager to activate its maintenance superframe.
- f) The network manager activates the high speed handheld superframe.
- g) The device and the handheld now communicate using this high speed handheld superframe.

Several prerequisites are necessary to support this scenario.

- a) The network manager installs a well-known high-speed handheld superframe in each device and into the handheld.
- b) The handheld and the selected device get a peer-peer session key.
- c) The handheld uses HART™ commands #806 and #807 to activate and deactivate the maintenance superframe.

The handheld device is restricted to using only the maintenance superframe for communication with the selected device.

Bibliography

IEC 61158-6-9, *Industrial communications networks – Fieldbus specifications – Part 6-9: Application layer protocol specification – Type 9 elements*

ISO/IEC 8822, *Information technology – Open Systems Interconnection – Presentation service definition*

ISO/IEC 15408 series, *Information technology – Security techniques – Evaluation criteria for IT security*

IEEE 754-2008, *IEEE Standard for Binary Floating-Point Arithmetic*

IEEE 802.3, *LAN/MAN CSMA/CD (Ethernet) Access Method*

IEEE 802.11, *LAN/MAN Wireless LANS*

NIST Pub 800-38C *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*
available at <<http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C.pdf>>

NIST FIPS 140 series *Security Requirements for Cryptographic Modules*
available at <<http://csrc.nist.gov/publications/PubsFIPS.html>>

HCF_SPEC-183, *Common Tables Specification*
available at <www.hartcomm.org>

IECNORM.COM: Click to view the full text of IEC 62591:2010

IECNORM.COM: Click to view the full PDF of IEC 62591:2010

SOMMAIRE

AVANT-PROPOS	465
INTRODUCTION.....	467
1 Domaine d'application	469
2 Références normatives.....	469
3 Termes, définitions, symboles, abréviations et conventions	470
3.1 Termes et définitions du modèle de référence	470
3.2 Termes et définitions issus de l'ISO/CEI 7498-1	471
3.2.1 Termes et définitions issus de l'ISO/CEI 9545	471
3.2.2 Termes et définitions issus de l'ISO/IEC 8824	471
3.2.3 Termes et définitions issus de l'ISO/IEC 8825	471
3.3 Termes et définitions spécifiques	471
3.4 Abréviations	480
3.5 Conventions	483
3.5.1 Conventions utilisées pour le service	483
3.5.2 Conventions utilisées pour le protocole DL (diagramme d'état)	484
3.5.3 Conventions pour le service Couche Application	484
3.5.4 Conventions pour le protocole AL	487
4 Spécification de couche physique et définition de services	492
4.1 Définition de services de couche physique	492
4.1.1 Vue d'ensemble	492
4.1.2 Services de couche physique	493
4.2 Spécification de protocole de couche physique	498
5 Définition des services de couche liaison de données – Éléments de Type 20.....	498
5.1 Services de la couche liaison de données	498
5.1.1 Moyens des services de la couche liaison de données	498
5.1.2 QoS pour le service messagerie	499
5.1.3 Séquence de primitives	499
5.1.4 Service DL-TRANSMIT	503
5.1.5 Service DL-RECEIVE	505
5.1.6 Service DL-DISCONNECT	505
5.1.7 Service DL-PATH-FAILURE	506
5.1.8 Service DL-ADVERTISE	506
5.1.9 Service DL-NEIGHBOR	507
5.1.10 DLM-SET	507
5.1.11 DLM-GET	508
5.1.12 DLM-ACTION	509
5.1.13 DLM-EVENT.....	509
5.2 Services de couche réseau	510
5.2.1 Moyens des services de la couche réseau	510
5.2.2 QoS pour le service messagerie	510
5.2.3 Séquence de primitives	511
5.2.4 Service NL-TRANSMIT	511
5.2.5 NLM-SET	512
5.2.6 NLM-GET	513
5.2.7 NLM-ACTION	514
5.3 Services de la couche transport	515

5.3.1	Moyens des services de la couche transport.....	515
5.3.2	QoS pour le service messagerie	516
5.3.3	Séquence de primitives	516
5.3.4	Service TL-DATA-TRANSFER	518
5.3.5	Service TL-DATA-EXCHANGE	520
5.3.6	TLM-SET	521
5.3.7	TLM-GET	522
6	Spécification de protocole de couche liaison de données – Éléments de Type 20.....	523
6.1	Vue d'ensemble.....	523
6.1.1	Fonctionnement de la couche liaison de données	523
6.1.2	Fonctionnement de la couche réseau	527
6.1.3	Procédures relatives aux dispositifs.....	528
6.2	Commande de liaison logique.....	529
6.2.1	Structure générale de DLPDU	529
6.2.2	Structure, codage et éléments de procédure spécifiques à la DLPDU	533
6.2.3	Priorité de DLPDU et régulation de débit.....	538
6.2.4	Détection d'erreurs et sécurité	539
6.3	Contrôle d'accès au support	541
6.3.1	Fonction	541
6.3.2	Temporisation d'intervalle de temps	541
6.3.3	Tableaux et tampons de communication.....	544
6.3.4	Programmation de liaison	551
6.3.5	Fonctionnement de MAC	555
6.4	DL-management-information	570
6.4.1	Généralités.....	570
6.4.2	Attributs de la DL.....	570
6.4.3	Action de la DLM	570
6.5	Couche réseau.....	571
6.5.1	Mode à deux niveaux de NL	571
6.5.2	Structure de NPDU structure	574
6.5.3	Fonction de sécurité	578
6.5.4	Tableaux de la couche réseau	583
6.5.5	Machine NLE	586
6.5.6	Éléments de procédure NLE	589
6.5.7	Informations de gestion de NL	592
6.6	Couche transport.....	593
6.6.1	Généralités.....	593
6.6.2	Structure de TPDU	594
6.6.3	Tableau de tube de transport.....	595
6.6.4	Éléments de procédure TLE	596
6.6.5	Diagrammes d'états de TLE.....	599
6.6.6	Informations de gestion de TL	604
7	Définition des services de la couche application – Éléments de Type 20	604
7.1	Concepts communs.....	604
7.2	Data type ASE.....	604
7.2.1	Vue d'ensemble.....	604
7.2.2	Définition formelle des objets de types de données	607
7.2.3	Types de données définis pour la FAL	608
7.2.4	Spécification de service de Data type ASE	612

7.2.5	Résumé des types de données	612
7.3	Spécification de modèle de communication	613
7.3.1	Paramètres communs	613
7.3.2	ASE	614
7.3.3	AR	681
7.3.4	Résumé des classes	683
7.3.5	Service permis par le rôle AREP	684
8	Spécification de protocole de couche application – Éléments de Type 20	685
8.1	Syntaxe abstraite	685
8.2	Syntaxe de transfert	685
8.2.1	Champs communs aux APDU	685
8.2.2	Structure commune aux APDU	687
8.2.3	Structures des APDU spécifiques aux services application de dispositifs	690
8.2.4	Structures des APDU spécifiques aux services de gestion de couche	716
8.2.5	Structures d'APDU spécifiques à la passerelle et au gestionnaire de réseau	760
8.2.6	Structures d'APDU spécifiques à la configuration de gestion de réseau	771
8.2.7	Règles de codage des données	788
8.3	Procédure communes	793
8.3.1	Réponse différée	793
8.3.2	Temps de réponse	799
8.4	Machine protocolaire des services FAL (FSPM)	799
8.4.1	Généralités	799
8.4.2	Tableau d'états de la FSPM	800
8.4.3	Fonctions utilisées par la FSPM	802
8.4.4	Paramètres de primitives de FSPM/ARPM	803
8.5	Machines protocolaires de relation d'applications (ARPM)	804
8.5.1	Mise en correspondance de l'AREP à la couche transport	804
8.5.2	Machines protocolaires de relation d'applications (ARPM)	805
8.5.3	Définitions des primitives de diagramme d'états AREP	807
8.5.4	Fonctions de diagramme d'états d'AREP	808
8.6	Machine protocolaire de mise en correspondance de TL (TMPM)	808
8.6.1	États de TMPM	808
8.6.2	Diagrammes d'états de TMPM	809
8.6.3	Primitives échangées entre couche de transfert et TMPM	809
8.6.4	Fonctions utilisées par TMPM	810
9	Famille de profils de communication 9 (HART™)	810
9.1	Aperçu général	810
9.2	Profil 9/1	811
9.3	Profil 9/2	811
9.3.1	Déclaration de conformité	811
9.3.2	Types de dispositifs	812
9.3.3	Couche physique	815
9.3.4	Couche liaison de données	819
9.3.5	Couche application	820
Annexe A	(normative) Sécurité	830
Annexe B	(normative) Procédures sans fil	845

Annexe C (normative) Gestion du réseau.....	850
Annexe D (normative) Passerelle.....	854
Annexe E (normative) Codes de réponse de commande	861
Annexe F (normative) Tableaux communs	871
Annexe G (informative) Application du mode édition de données et commandes d'événements	885
Annexe H (informative) Redondance de réseau	890
Annexe I (informative) Mise en œuvre du gestionnaire de réseau	892
Annexe J (informative) Mise en œuvre d'une passerelle.....	923
Annexe K (informative) Dispositif tenu à la main	936
Bibliographie.....	939
Figure 1 – Exemple de classe Super trame.....	489
Figure 2 – Exemple de relation d'héritage	491
Figure 3 – Exemple d'agrégation de base	491
Figure 4 – Exemple d'agrégation de composition	492
Figure 5 – Séquences de services de couche physique	494
Figure 6 – Séquences sans erreurs de services message.....	501
Figure 7 – Séquences de service message avec erreurs.....	503
Figure 8 – Service message "réception seule"	503
Figure 9 – Séquences de services de transfert de données NLL	511
Figure 10 – Séquences sans erreurs du service transfert de données.....	517
Figure 11 – Séquences avec erreurs du service transfert de données.....	518
Figure 12 – Intervalle de temps TDMA et supertrame.....	524
Figure 13 – Saut de voie.....	525
Figure 14 – Réseau sans fil	528
Figure 15 – Structure de DLPDU.....	529
Figure 16 – Spécificateur d'adresse	530
Figure 17 – Construction de l'adresse EUI-64 de 8 octets	531
Figure 18 – Spécificateur de DLPDU.....	532
Figure 19 – Détails de temporisation d'un intervalle de temps	542
Figure 20 – Tableaux de DLE et relation	544
Figure 21 – Relations utilisées pour la programmation de liaisons	552
Figure 22 – Composants MAC	556
Figure 23 – Diagramme d'état TDMA	559
Figure 24 – Diagramme d'état XMIT.....	564
Figure 25 – Diagramme d'état RECV	567
Figure 26 – Diagramme d'état pour l'émission des ACK	569
Figure 27 – Routage par graphe	572
Figure 28 – Routage source.....	574
Figure 29 – Structure de NPDU.....	575
Figure 30 – Octet de commande de NPDU.....	576
Figure 31 – Octet de commande de sécurité	577

Figure 32 – Tableaux de NLE et relation	584
Figure 33 – Diagramme d'état NLE	588
Figure 34 – Décision de routage de la NPDU	591
Figure 35 – Structure de TPDU	594
Figure 36 – Octet de commande de TPDU	594
Figure 37 – Diagramme de transition relatif au diagramme d'états de transfert de données client	599
Figure 38 – Diagramme de transition relatif au diagramme d'états d'échange de données client	600
Figure 39 – Diagramme de transition relatif au diagramme d'états serveur	602
Figure 40 – Hiérarchie de la classe de types de données	606
Figure 41 – Format d'APDU de demande du maître à l'esclave	687
Figure 42 – Format d'APDU de réponse normale de l'esclave au maître	687
Figure 43 – Format d'APDU de réponse en cas d'erreur de commande de l'esclave au maître	688
Figure 44 – APDU de commande agrégée	689
Figure 45 – Mode de déclenchement à fenêtre	702
Figure 46 – Mode de déclenchement à fenêtre avec expiration du temps maximal de mise à jour	703
Figure 47 – Mode de déclenchement à niveau	704
Figure 48 – Codage sans identification	788
Figure 49 – Codage des données de type Integer	788
Figure 50 – Codage des données de type Unsigned	789
Figure 51 – Codage de données du type virgule flottante en simple précision	789
Figure 52 – Codage de données du type virgule flottante en double précision	790
Figure 53 – Codage des données de type Date	790
Figure 54 – Fonctionnement normal de DR	795
Figure 55 – Réponses de commande pendant le traitement de DR	797
Figure 56 – Esclaves avec plusieurs tampons de DR	799
Figure 57 – Diagramme de transitions d'états de la FSPM	800
Figure 58 – Diagramme de transitions d'états de l'ARPM client	806
Figure 59 – Diagramme de transitions d'états de l'ARPM serveur	807
Figure 60 – Diagramme de transitions d'états de la TMPM	808
Figure 61 – Composants de réseau sans fil	813
Figure A.1 – Séquence de rattachement	834
Figure A.2 – Procédure de rattachement de couche réseau	839
Figure A.3 – Procédure de rattachement de couche liaison de données	841
Figure A.4 – Opération de changement de clé réseau	843
Figure B.1 – Découverte de voisins	846
Figure B.2 – Défaillance de trajet	848
Figure B.3 – Dispositif quittant le réseau	849
Figure C.1 – Gestionnaire de réseau dans un réseau sans fil	851
Figure D.1 – Point d'accès réseau unique avec horloge	855
Figure D.2 – Plusieurs points d'accès réseau avec horloges	856

Figure D.3 – Point d'accès réseau unique ne fournissant pas d'horloge	857
Figure G.1 – Activation ou désactivation de la séquence d'édition	887
Figure H.1 – Routage de réseau	890
Figure I.1 – Modèle général pour gestionnaire de réseau	893
Figure I.2 – Modèle de données d'un type de dispositif	895
Figure I.3 – Routage de réseau	897
Figure I.4 – Programme de réseau	900
Figure I.5 – Exemple d'une supertrame de trois intervalles de temps	900
Figure I.6 – Multiples supertrames dans un réseau	901
Figure I.7 – Gestionnaire de sécurité	903
Figure I.8 – Modèle détaillé du gestionnaire de réseau	905
Figure I.9 – Réseau sans fil à quatre dispositifs de réseau	911
Figure I.10 – Séquences d'APDU	917
Figure I.11 - Initialisation d'un réseau sans fil	919
Figure I.12 – Allocation et utilisation de ressources réseau	920
Figure I.13 – Ajustement du programme de réseau	921
Figure I.14 – Rapports de santé	922
Figure J.1 – Modèle général pour passerelle	924
Figure J.2 – Dispositif de réseau logique	925
Figure J.3 – Dispositif de réseau physique	926
Figure J.4 – Gestion des services de notification	932
Tableau 1 – Conventions utilisées pour les diagrammes d'états	488
Tableau 2 – Notation des relations entre classes	490
Tableau 3 – Notation de la multiplicité	492
Tableau 4 – Primitives et paramètres de PH-ENABLE	495
Tableau 5 – Primitives et paramètres de PH-CCA	495
Tableau 6 – Primitives et paramètres de PH-DATA	496
Tableau 7 – Primitive et paramètres de PHM-SET	497
Tableau 8 – Primitive et paramètres de PHM-GET	497
Tableau 9 – Primitives et paramètres de DL-Transmit	504
Tableau 10 – Primitive et paramètres de DL-RECEIVE	505
Tableau 11 – Primitive et paramètres de DL-DISCONNECT	506
Tableau 12 – Primitive et paramètres de DL-PATH-FAILURE	506
Tableau 13 – Primitive et paramètres de DL-ADVISE	507
Tableau 14 – Primitive et paramètres de DL-NEIGHBOR	507
Tableau 15 – Primitive et paramètres de DLM-SET	507
Tableau 16 – Primitive et paramètres de DLM-GET	508
Tableau 17 – Primitive et paramètres de DLM-ACTION	509
Tableau 18 – Primitives et paramètres de NL-TRANSMIT	511
Tableau 19 – Primitive et paramètres de NLM-SET	513
Tableau 20 – Primitive et paramètres de NLM-GET	513
Tableau 21 – Primitive et paramètres de NLM-ACTION	514

Tableau 22 – Primitives et paramètres de TL-DATA-TRANSFER	518
Tableau 23 – Primitives et paramètres de TL-DATA-EXCHANGE	520
Tableau 24 – Primitive et paramètres de TLM-Set.....	522
Tableau 25 – Primitive et paramètres de TLM-Get	522
Tableau 26 – Allocation de network_ID	530
Tableau 27 – Contenu de la charge utile d'une DLPDU ACK	534
Tableau 28 – Code de réponse ACK	534
Tableau 29 – Contenu de la charge utile d'une DLPDU Advertise	536
Tableau 30 – Début de la supertrame	536
Tableau 31 – Le prochain champ après la taille de la dernière supertrame	536
Tableau 32 – Répété pour chaque liaison	536
Tableau 33 – Affectations des voies et de la fréquence.....	537
Tableau 34 – Définitions et valeurs de temporisation d'un intervalle de temps	542
Tableau 35 – Exigence en minimum de Tableaux de DLE et d'espace tampon	545
Tableau 36 – Attributs de supertrame	546
Tableau 37 – Attributs d'une liaison	546
Tableau 38 – Attributs de voisin et variables	548
Tableau 39 – Attributs d'un graphe	549
Tableau 40 – Enregistrement de paquets.....	550
Tableau 41 – Exemple d'ensembles de sélection de Backoff_cntr	554
Tableau 42 – Diagramme d'état TDMA.....	560
Tableau 43 – Transitions d'états de XMIT	564
Tableau 44 – Diagramme d'état RECV.....	567
Tableau 45 – Diagramme d'état pour l'émission des ACK	569
Tableau 46 – Attributs de la DL.....	570
Tableau 47 – Actions de DL.....	570
Tableau 48 – Valeur du Compteur.....	578
Tableau 49 – Attributs de Session	579
Tableau 50 – Nonce de NPDU	581
Tableau 51 – Exigence d'espace du au nombre minimum des NLE	584
Tableau 52 – Attributs de chemin.....	585
Tableau 53 – Attributs de chemins de source.....	585
Tableau 54 – Attributs de la table des temps	586
Tableau 55 – États de NLE	586
Tableau 56 – Construction de la NPDU.....	589
Tableau 57 – Actions de routage	591
Tableau 58 – Attributs de la NL.....	592
Tableau 59 – Actions de la NL	593
Tableau 60 – Attributs du tube de transport	595
Tableau 61 – Codage de commande de transport pour la demande "TL-DATA-TRANSFER request"	596
Tableau 62 – Codage de commande de transport pour la demande "TL-DATA-EXCHANGE request"	596

Tableau 63 – Codage de commande de transport pour la réponse "TL-DATA-EXCHANGE response"	597
Tableau 64 – Tableau d'états de transfert de données client	600
Tableau 65 – Tableau d'états d'échange de données client	601
Tableau 66 – Tableau d'états serveur	603
Tableau 67 – Résumé des types de données	612
Tableau 68 – Valeurs du code de réponse	613
Tableau 69 – Paramètres du service Read	617
Tableau 70 – Paramètres du service Write	618
Tableau 71 – Paramètres du service "Information report"	619
Tableau 72 – Paramètres du service Action	621
Tableau 73 – Paramètres du service AR get attributes	682
Tableau 74 – Paramètres du service AR set attributes	683
Tableau 75 – Résumé des classes	684
Tableau 76 – Services confirmés par la classe AREP	684
Tableau 77 – Services non confirmés de la classe AREP	684
Tableau 78 – Valeurs du code de réponse	685
Tableau 79 – Valeurs d'Extended status	686
Tableau 80 – Champ "Value" de Reset configuration changed flags	691
Tableau 81 – Codes de réponse spécifiques à la commande "Reset configuration changed flag"	691
Tableau 82 – Codes de réponse spécifiques à la commande "Perform self test"	692
Tableau 83 – Codes de réponse spécifiques à la commande "Perform device reset"	693
Tableau 84 – Champ Valeur de "Read additional device status"	693
Tableau 85 – Codes de réponse spécifiques à la commande "Read additional device status"	694
Tableau 86 – Champ Valeur de "Read additional device status"	694
Tableau 87 – Codes de réponse spécifiques à la commande "Read additional device status"	695
Tableau 88 – Champ Valeur de "Read device variable information"	695
Tableau 89 – Champ "Valeur" de réponse "Read device variable information"	696
Tableau 90 – Codes de réponse spécifiques à la commande "Read device variable information"	696
Tableau 91 – "Champ "Value" de "Write device variable"	697
Tableau 92 – Codes de réponse spécifiques à la commande "Write device variable"	697
Tableau 93 – Champ "Value" de "Read real-time clock"	698
Tableau 94 – Codes de réponse spécifiques à la commande "Read real-time clock"	698
Tableau 95 – Champ "Value" de "Write publish data period"	699
Tableau 96 – Codes de réponse spécifiques à la commande "Write publish data period"	699
Tableau 97 – Champ "Write publish data trigger"	700
Tableau 98 – Codes de réponse spécifiques à la commande "Write publish data trigger"	700
Tableau 99 – Source de déclenchement de message Publish data	701
Tableau 100 – Champ "Value" de la demande "Read publish data mode configuration"	704

Tableau 101 – Champ "Value" de la réponse "Read publish data mode configuration"	705
Tableau 102 – Codes de réponse spécifiques à la commande "Read publish data mode configuration"	705
Tableau 103 – Codes de réponses spécifiques à la commande "Flush delayed responses"	706
Tableau 104 – Champ "Value" de "Write publish data mode configuration"	707
Tableau 105 – Codes de réponse spécifiques à la commande "Write publish data device variables"	707
Tableau 106 – Champ "Value" de "Tableau"	708
Tableau 107 – Champ "Value" de "Write publish data mode command number"	708
Tableau 108 – Codes de réponse spécifiques à la commande "Write publish data mode command number"	708
Tableau 109 – Champ "Value" de "Write publish data mode control"	709
Tableau 110 – Codes de réponse spécifiques à la commande "Write publish data mode control"	709
Tableau 111 – Champ "Value" de la demande "Read event notification summary"	711
Tableau 112 – Champ "Value" de la réponse "Read event notification summary"	711
Tableau 113 – Codes de réponse spécifiques à la commande "Read event notification summary"	712
Tableau 114 – Champ "Value" de "Write event notification bit mask"	712
Tableau 115 – Codes de réponse spécifiques à la commande "Write event notification bit mask"	712
Tableau 116 – Champ "Value" de "Write event notification timing"	713
Tableau 117 – Codes de réponse spécifiques à la commande "Write event notification timing"	713
Tableau 118 – Champ "Value" de "Write event notification control"	714
Tableau 119 – Codes de réponse spécifiques à la commande "Write event notification control"	714
Tableau 120 – Champ "Value" de "Write event notification acknowledgement"	715
Tableau 121 – Codes de réponse spécifiques à la commande "Write event notification acknowledgement"	716
Tableau 122 – Champ "Value" de "Write Join_key"	716
Tableau 123 – Codes de réponse spécifiques à la commande "Write Join_key"	717
Tableau 124 – Champ "Value" de "Read join status"	717
Tableau 125 – Codes de réponse spécifiques à la commande "Read join status"	718
Tableau 126 – Champ "Value" de demande "Write Active_search_shed_time"	718
Tableau 127 – Champ "Value" de réponse de "Write Active_search_shed_time"	718
Tableau 128 – Codes de réponse spécifiques à la commande "Write Active_search_shed_time"	718
Tableau 129 – Champ "Value" de "Write Join mode configuration"	719
Tableau 130 – Codes de réponse spécifiques à la commande "Write Join mode configuration"	719
Tableau 131 – Champ "Value" de la réponse "Read Join mode configuration"	720
Tableau 132 – Codes de réponse spécifiques à la commande "Read Join mode configuration"	720
Tableau 133 – Champ "Value" de "Write network_ID"	721
Tableau 134 – Codes de réponse spécifiques à la commande "Write network_ID"	721

Tableau 135 – Champ "Value" de "Read network_ID"	722
Tableau 136 – Codes de réponse spécifiques à la commande "Read network_ID"	722
Tableau 137 – Champ "Value" de "Write Network_tag"	722
Tableau 138 – Codes de réponse spécifiques à la commande "Write Network_tag"	722
Tableau 139 – Champ "Value" de réponse "Read Network_tag"	723
Tableau 140 – Codes de réponse spécifiques à la commande "Read Network_tag"	723
Tableau 141 – Champ "Value" de "Read wireless device capabilities"	723
Tableau 142 – Codes de réponse spécifiques à la commande "Read wireless device capabilities"	724
Tableau 143 – Champ "Value" de réponse "Read battery life"	724
Tableau 144 – Codes de réponse spécifiques à la commande "Read battery life"	724
Tableau 145 – Champ "Value" de réponse de "Report device health"	725
Tableau 146 – Codes de réponse spécifiques à la commande "Report device health"	725
Tableau 147 – Champ "Value" de demande "Read neighbor health list"	725
Tableau 148 – Champ "Value" de réponse de "Read neighbor health list"	726
Tableau 149 – Codes de réponse spécifiques à la commande "Read neighbor health list" ..	726
Tableau 150 – Champ "Value" de réponse de "Read device nickname"	727
Tableau 151 – Codes de réponse spécifiques à la commande "Read device nickname"	727
Tableau 152 – Champ "Value" de demande "Read session list"	727
Tableau 153 – Champ "Value" de réponse de "Read session list"	727
Tableau 154 – Codes de réponse spécifiques à la commande "Read session list"	728
Tableau 155 – Champ "Value" de demande "Read superframe list"	728
Tableau 156 – Champ "Value" de réponse de "Read superframe list"	728
Tableau 157 – Codes de réponse spécifiques à la commande "Read superframe list"	729
Tableau 158 – Champ "Value" de demande "Read link list"	729
Tableau 159 – Champ "Value" de réponse de "Read link list"	729
Tableau 160 – Codes de réponse spécifiques à la commande "Read link list"	730
Tableau 161 – Champ "Value" de demande "Read graph list"	730
Tableau 162 – Champ "Value" de réponse de "Read graph list"	730
Tableau 163 – Codes de réponse spécifiques à la commande "Read graph list"	731
Tableau 164 – Champ "Value" de demande "Read neighbor attribute flag"	731
Tableau 165 – Champ "Value" de réponse de "Read neighbor attribute flag"	731
Tableau 166 – Codes de réponse spécifiques à la commande "Read neighbor attribute flag"	732
Tableau 167 – Champ "Value" de demande "Read neighbor signal level"	732
Tableau 168 – Champ "Value" de réponse de "Read neighbor signal level"	732
Tableau 169 – Codes de réponse spécifiques à la commande "Read neighbor signal level"	732
Tableau 170 – Champ "Value" de réponse de "Report path down alarm"	733
Tableau 171 – Codes de réponse spécifiques à la commande "Report path down alarm"	733
Tableau 172 – Champ "Value" de réponse de "Report source route failed alarm"	734
Tableau 173 – Codes de réponse spécifiques à la commande "Report source route failed alarm"	734
Tableau 174 – Champ "Value" de réponse de "Report graph route failed alarm"	734

Tableau 175 – Codes de réponse spécifiques à la commande "Report graph route failed alarm"	734
Tableau 176 – Champ "Value" de réponse de "Report transport layer failed alarm"	735
Tableau 177 – Codes de réponse spécifiques à la commande "Report transport layer failed alarm"	735
Tableau 178 – Champ "Value" de "Write UTC time mapping"	735
Tableau 179 – Codes de réponse spécifiques à la commande "Write UTC time mapping"	736
Tableau 180 – Champ "Value" de réponse de "Read UTC time mapping"	736
Tableau 181 – Codes de réponse spécifiques à la commande "Read UTC time mapping"	736
Tableau 182 – Champ "Value" de "Write timer interval"	737
Tableau 183 – Codes de réponse spécifiques à la commande "Write timer interval"	737
Tableau 184 – Champ "Value" de demande "Read timer interval"	737
Tableau 185 – Champ "Value" de réponse de "Read timer interval"	738
Tableau 186 – Codes de réponse spécifiques à la commande "Read timer interval"	738
Tableau 187 – Champ "Value" de "Write radio power output"	738
Tableau 188 – Codes de réponse spécifiques à la commande "Write radio power output"	738
Tableau 189 – Champ "Value" de réponse de "Read radio power output"	739
Tableau 190 – Codes de réponse spécifiques à la commande "Read radio power output"	739
Tableau 191 – Champ "Value" de "Request TimeTable"	740
Tableau 192 – Codes de réponse spécifiques à la commande "Request TimeTable"	740
Tableau 193 – Champ "Value" de demande "Read TimeTable list"	741
Tableau 194 – Champ "Value" de réponse de "Read TimeTable list"	741
Tableau 195 – Codes de réponse spécifiques à la commande "Read TimeTable list"	742
Tableau 196 – Champ "Value" de demande "Delete TimeTable"	742
Tableau 197 – Champ "Value" de réponse de "Delete TimeTable"	742
Tableau 198 – Codes de réponse spécifiques à la commande "Delete TimeTable"	743
Tableau 199 – Champ "Value" de demande "Read route list"	743
Tableau 200 – Champ "Value" de réponse de "Read route list"	743
Tableau 201 – Codes de réponse spécifiques à la commande "Read route list"	744
Tableau 202 – Champ "Value" de demande "Read source route"	744
Tableau 203 – Champ "Value" de réponse de "Read source route"	744
Tableau 204 – Codes de réponse spécifiques à la commande "Read source route"	745
Tableau 205 – Champ "Value" de réponse de "Read CCA mode"	745
Tableau 206 – Codes de réponse spécifiques à la commande "Read CCA mode"	745
Tableau 207 – Champ "Value" de "Write CCA mode"	745
Tableau 208 – Codes de réponse spécifiques à la commande "Write CCA mode"	746
Tableau 209 – Champ "Value" de réponse de "Read handheld superframe"	746
Tableau 210 – Codes de réponse spécifiques à la commande "Read handheld superframe"	747
Tableau 211 – Champ "Value" de "Write handheld superframe"	747
Tableau 212 – Codes de réponse spécifiques à la commande "Write handheld superframe"	747

Tableau 213 – Champ "Value" de réponse de "Read packet hop-to-live"	748
Tableau 214 – Codes de réponse spécifiques à la commande "Read packet hop-to-live"	748
Tableau 215 – Champ "Value" de "Write packet hop-to-live"	748
Tableau 216 – Codes de réponse spécifiques à la commande "Write packet hop-to-live"	748
Tableau 217 – Champ "Value" de réponse de "Read Join_priority"	749
Tableau 218 – Codes de réponse spécifiques à la commande "Read Join_priority"	749
Tableau 219 – Champ "Value" de "Write Join_priority"	749
Tableau 220 – Codes de réponse spécifiques à la commande "Write Join_priority"	750
Tableau 221 – Champ "Value" de réponse de "Read receive Priority_threshold"	750
Tableau 222 – Codes de réponse spécifiques à la commande "Read receive Priority_threshold"	750
Tableau 223 – Champ "Value" de "Write receive Priority_threshold"	751
Tableau 224 – Codes de réponse spécifiques à la commande "Write receive Priority_threshold"	751
Tableau 225 – Champ "Value" de demande "Read device list"	751
Tableau 226 – Champ "Value" de réponse de "Read device list"	752
Tableau 227 – Codes de réponse spécifiques à la commande "Read device list"	752
Tableau 228 – Champ "Value" de demande "Add device list entry"	753
Tableau 229 – Champ "Value" de réponse de "Add device list entry"	753
Tableau 230 – Codes de réponse spécifiques à la commande "Add device list entry"	753
Tableau 231 – Champ "Value" de demande "Delete device list entry"	754
Tableau 232 – Champ "Value" de réponse de "Delete device list entry"	754
Tableau 233 – Codes de réponse spécifiques à la commande "Delete device list entry"	754
Tableau 234 – Champ "Value" de réponse de "Read channel blacklist"	755
Tableau 235 – Codes de réponse spécifiques à la commande "Read channel blacklist"	755
Tableau 236 – Champ "Value" de "Write channel blacklist"	755
Tableau 237 – Codes de réponse spécifiques à la commande "Write channel blacklist"	756
Tableau 238 – Champ "Value" de réponse de "Read back-off exponent"	756
Tableau 239 – Codes de réponse spécifiques à la commande "Read back-off exponent"	756
Tableau 240 – Champ "Value" de "Write back-off exponent"	757
Tableau 241 – Codes de réponse spécifiques à la commande "Write back-off exponent"	757
Tableau 242 – Champ "Value" de "Write network access mode"	758
Tableau 243 – Codes de réponse spécifiques à la commande "Write network access mode"	758
Tableau 244 – Champ "Value" de réponse de "Read network access mode"	758
Tableau 245 – Codes de réponse spécifiques à la commande "Read network access mode"	759
Tableau 246 – Champ "Value" de demande "Request Session"	759
Tableau 247 – Champ "Value" de réponse de "Request Session"	759
Tableau 248 – Codes de réponse spécifiques à la commande "Request Session"	759
Tableau 249 – Champ "Value" de demande "Read network device identity"	760
Tableau 250 – Champ "Value" de réponse de "Read network device identity"	760

Tableau 251 – Codes de réponse spécifiques à la commande "Read network device identity"	760
Tableau 252 – Champ "Value" de demande "Read device's neighbor health"	760
Tableau 253 – Champ "Value" de réponse de "Read device's neighbor health"	761
Tableau 254 – Codes de réponse spécifiques à la commande "Read device's neighbor health"	761
Tableau 255 – Champ "Value" de demande "Read network topology information"	762
Tableau 256 – Champ "Value" de réponse de "Read network topology information"	762
Tableau 257 – Codes de réponse spécifiques à la commande "Read network topology information"	762
Tableau 258 – Champ "Value" de demande "Read publish data message list"	763
Tableau 259 – Champ "Value" de réponse de "Read publish data message list"	763
Tableau 260 – Codes de réponse spécifiques à la commande "Read publish data message list"	763
Tableau 261 – Champ "Value" de "Flush cached"	763
Tableau 262 – Codes de réponses spécifiques à la commande "Flush cached responses"	764
Tableau 263 – Champ "Value" de "Write update notification bit mask for a device"	764
Tableau 264 – Codes de réponse spécifiques à la commande "Write update notification bit mask for a device"	764
Tableau 265 – Champ "Value" de demande "Read update notification bit mask for a device"	765
Tableau 266 – Champ "Value" de réponse de "Read update notification bit mask for a device"	765
Tableau 267 – Codes de réponse spécifiques à la commande "Read update notification bit mask for a device"	765
Tableau 268 – Champ "Value" de demande "Report change notification"	765
Tableau 269 – Champ "Value" de réponse de "Report change notification"	766
Tableau 270 – Codes de réponse spécifiques à la commande "Report change notification"	766
Tableau 271 – Champ "Value" de demande "Read network device statistics"	766
Tableau 272 – Champ "Value" de réponse de "Read network device statistics"	766
Tableau 273 – Codes de réponse spécifiques à la commande "Read network device statistics"	767
Tableau 274 – Champ "Value" de demande "Read network device identity"	767
Tableau 275 – Champ "Value" de réponse de "Read network device identity"	768
Tableau 276 – Codes de réponse spécifiques à la commande "Read network device identity"	768
Tableau 277 – Champ "Value" de "Write network device's scheduling flags"	768
Tableau 278 – Codes de réponse spécifiques à la commande "Write network device's scheduling flags"	768
Tableau 279 – Champ "Value" de demande "Read network device's scheduling flags"	769
Tableau 280 – Champ "Value" de réponse de "Read network device's scheduling flags"	769
Tableau 281 – Codes de réponse spécifiques à la commande "Read network device's scheduling flags"	769
Tableau 282 – Champ "Value" de "Read network constraints"	770
Tableau 283 – Codes de réponse spécifiques à la commande "Read network constraints"	770

Tableau 284 – Champ "Value" de "Write network constraints"	770
Tableau 285 – Codes de réponse spécifiques à la commande "Write network constraints"	771
Tableau 286 – Champ "Value" de "Disconnect device"	771
Tableau 287 – Codes de réponse spécifiques à la commande "Disconnect device"	771
Tableau 288 – Champ "Value" de "Write network key"	772
Tableau 289 – Codes de réponse spécifiques à la commande "Write network key"	772
Tableau 290 – Champ "Value" de "Write device nickname"	772
Tableau 291 – Codes de réponse spécifiques à la commande "Write device nickname"	773
Tableau 292 – Champ "Value" de demande "Write session"	773
Tableau 293 – Champ "Value" de réponse de "Write session"	773
Tableau 294 – Codes de réponse spécifiques à la commande "Write session"	774
Tableau 295 – Champ "Value" de demande "Delete session"	774
Tableau 296 – Champ "Value" de réponse de "Delete session"	775
Tableau 297 – Codes de réponse spécifiques à la commande "Delete session"	775
Tableau 298 – Champ "Value" de demande "Write superframe"	775
Tableau 299 – Champ "Value" de réponse de "Write superframe"	775
Tableau 300 – Codes de réponse spécifiques à la commande "Write superframe"	776
Tableau 301 – Champ "Value" de demande "Delete superframe"	776
Tableau 302 – Champ "Value" de réponse de "Delete superframe"	777
Tableau 303 – Codes de réponse spécifiques à la commande "Delete superframe"	777
Tableau 304 – Champ "Value" de demande "Write link"	777
Tableau 305 – Champ "Value" de réponse de "Write link"	777
Tableau 306 – Codes de réponse spécifiques à la commande "Write link"	778
Tableau 307 – Champ "Value" de demande "Delete link"	779
Tableau 308 – Champ "Value" de réponse de "Delete link"	779
Tableau 309 – Codes de réponse spécifiques à la commande "Delete link"	779
Tableau 310 – Champ "Value" de demande "Write graph connection"	780
Tableau 311 – Champ "Value" de réponse de "Write graph connection"	780
Tableau 312 – Codes de réponse spécifiques à la commande "Write graph connection"	780
Tableau 313 – Champ "Value" de demande "Delete graph connection"	781
Tableau 314 – Champ "Value" de réponse de "Delete graph connection"	781
Tableau 315 – Codes de réponse spécifiques à la commande "Delete graph connection"	781
Tableau 316 – Champ "Value" de "Write neighbor attribute flag"	781
Tableau 317 – Codes de réponse spécifiques à la commande "Write neighbor attribute"	782
Tableau 318 – Champ "Value" de "Write network suspend"	782
Tableau 319 – Codes de réponse spécifiques à la commande "Write network suspend"	782
Tableau 320 – Champ "Value" de demande "Write TimeTable"	783
Tableau 321 – Champ "Value" de réponse de "Write TimeTable"	783
Tableau 322 – Codes de réponse spécifiques à la commande "Write TimeTable"	784
Tableau 323 – Champ "Value" de demande "Write route"	784
Tableau 324 – Champ "Value" de réponse de "Write route"	785
Tableau 325 – Codes de réponse spécifiques à la commande "Write route"	785

Tableau 326 – Champ "Value" de demande "Delete route"	785
Tableau 327 – Champ "Value" de réponse de "Delete route"	785
Tableau 328 – Codes de réponse spécifiques à la commande "Delete route"	786
Tableau 329 – Champ "Value" de demande "Write source route"	786
Tableau 330 – Champ "Value" de réponse de "Write source route"	786
Tableau 331 – Codes de réponse spécifiques à la commande "Write source route"	787
Tableau 332 – Champ "Value" de demande "Delete source route"	787
Tableau 333 – Champ "Value" de réponse de "Delete source route"	787
Tableau 334 – Codes de réponse spécifiques à la commande "Delete source route"	787
Tableau 335 – Codage des données de type Integer16	788
Tableau 336 – Codage des données de type Unsigned16	789
Tableau 337 – Codage du type Date type	790
Tableau 338 – Codage des données de type Bit Field de 1 octet	791
Tableau 339 – Codage des données de type Bit Field de 2 octets	791
Tableau 340 – Jeu de caractères Packed ASCII	792
Tableau 341 – Sous-ensemble de caractères ISO Latin-1	793
Tableau 342 – Codes de réponse liés au mécanisme DRM	794
Tableau 343 – Tableau d'états de la FSPM – transactions client	800
Tableau 344 – Tableau d'états de la FSPM – transactions serveur	802
Tableau 345 – Fonction TransportType ()	802
Tableau 346 – Fonction PriorityType ()	802
Tableau 347 – Fonction FormReqApdu ()	802
Tableau 348 – Fonction FormRspApdu ()	803
Tableau 349 – Fonction RetrieveCommand ()	803
Tableau 350 – Fonction CommandErr ()	803
Tableau 351 – Fonction ResponseCode ()	803
Tableau 352 – Fonction DeviceStat ()	803
Tableau 353 – Fonction Value ()	803
Tableau 354 – Paramètres utilisés avec les primitives échangées entre la FSPM et l'ARPM	803
Tableau 355 – États ARPM Client	805
Tableau 356 – Tableau d'états de l'ARPM Client	806
Tableau 357 – États ARPM Serveur	807
Tableau 358 – Tableau d'états de l'ARPM Serveur	807
Tableau 359 – Primitives émises de l'ARPM vers la TPM	808
Tableau 360 – Primitives émises par la TPM vers l'ARPM	808
Tableau 361 – Descriptions d'états de TPM	808
Tableau 362 – Tableau d'états de la TPM – transactions Client	809
Tableau 363 – Tableau d'états de la TPM – transactions Serveur	809
Tableau 364 – Primitives échangées entre couche réseau et TPM	810
Tableau 365 – CPF 9: vue d'ensemble des jeux de profils	810
Tableau 366 – CP 9/2: Sélection de protocoles PhL	817
Tableau 367 – Spécifications de l'émetteur-récepteur	818

Tableau 368 – Attributions de fréquences	818
Tableau 369 – Attributs de la PIB de la PHY	819
Tableau 370 – CP 9/2: Sélection des services de DLL	820
Tableau 371 – CP 9/2: Sélection de protocoles de DLL	820
Tableau 372 – CP 9/2: Sélection de services d'AL	821
Tableau 373 – CP 9/2: Sélection des services d'application de dispositif	822
Tableau 374 – CP 9/2: Sélection de services de gestion de couche	823
Tableau 375 – CP 9/2: Sélection de services de passerelle et de gestionnaire de réseau	824
Tableau 376 – CP 9/2: Sélection de services de configuration de la gestion de réseau	824
Tableau 377 – CP 9/2: Sélection de protocoles d'AL	825
Tableau 378 – Valeurs de statut de processus d'application	825
Tableau 379 – CP 9/2: Sélection des APDU spécifiques aux services application de dispositifs	826
Tableau 380 – CP 9/2: Sélection des APDU spécifiques aux services de gestion de couche	827
Tableau 381 – CP 9/2: Sélection des APDU spécifiques aux services de passerelle et de gestionnaire de réseau	828
Tableau 382 – CP 9/2: Sélection des APDU spécifiques aux services configuration de gestion de réseau	829
Tableau C.1 – Exigences relatives au gestionnaire de réseau	852
Tableau D.1 – Messages de réponse mis en cache	859
Tableau E.1 – Valeurs du code de réponse	861
Tableau F.1 – Valeurs réservées	871
Tableau F.2 – Codes de commande du mode édition (Publish mode control codes)	873
Tableau F.3 – Write device variable codes	873
Tableau F.4 – Codes de la famille de variables Device	873
Tableau F.5 – Codes de classification de variables Device	874
Tableau F.6 – Codes saturés de voie analogique	874
Tableau F.7 – Codes fixes de voie analogique	875
Tableau F.8 – Codes Standardized status 0	875
Tableau F.9 – Codes Standardized status 1	875
Tableau F.10 – Codes Standardized status 2	876
Tableau F.11 – Codes Standardized status 3	876
Tableau F.12 – Codes du mode déclenchement d'édition	876
Tableau F.13 – Codes de commande de notifications d'événements	877
Tableau F.14 – Codes de statut d'événement	877
Tableau F.15 – Codes de demande d'indicateur horaire (Timetable)	877
Tableau F.16 – Codes de domaine d'application d'indicateur horaire (Timetable)	877
Tableau F.17 – Codes de commande d'action synchrone	878
Tableau F.18 – Codes de fanions d'horloge temps réel	878
Tableau F.19 – Codes de temporisateur sans fil	878
Tableau F.20 – Codes de source d'énergie du dispositif	879
Tableau F.21 – Codes Link_type	879
Tableau F.22 – Codes de fanions options de liaison	879

Tableau F.23 – Codes de fanions mode de supertrame.....	879
Tableau F.24 – Codes Session_type.....	880
Tableau F.25 – Codes de raison de la suppression de Timetable (indicateur horaire)	880
Tableau F.26 – Codes de cause de déconnexion	880
Tableau F.27 – Codes de fonctionnement sans fil	880
Tableau F.28 – Codes de statut du processus de rattachement	881
Tableau F.29 – Codes de type de sécurité	881
Tableau F.30 – Codes de liste de dispositifs	881
Tableau F.31 – Codes de mode d'accès au réseau	882
Tableau F.32 – Codes de profil de dispositifs.....	882
Tableau F.33 – Codes de statut d'énergie du dispositif	882
Tableau F.34 – Codes de fanions relatifs aux voisins.....	883
Tableau F.35 – Codes de masques de notification	883
Tableau F.36 – Codes de mode de rattachement	883
Tableau F.37 – Codes de fanions programmation de dispositif.....	883
Tableau F.38 – Codes de fanions optimisation de réseau.....	884
Tableau I.1 – Exigences du routage.....	906
Tableau I.2 – Exigences de programmation	907
Tableau I.3 – ID de trame 1: fréquence de mise à jour 1 s, longueur de supertrame 100.....	911
Tableau I.4 – ID de trame 4: fréquence de mise à jour 4 s, longueur de supertrame 400.....	912
Tableau I.5 – ID de trame 0: supertrame de gestion.....	912
Tableau I.6 – Demande de rattachement (partagée avec des réponses de gestion)	912
Tableau I.7 – Réponse de rattachement (partagée avec des demandes de gestion).....	912
Tableau I.8 – Commandes	913
Tableau I.9 – Réponses de commande	913
Tableau I.10 – Dispositif A.....	913
Tableau I.11 – Dispositif B.....	914
Tableau I.12 – Dispositif C.....	915
Tableau I.13 – Dispositif D.....	915
Tableau I.14 – Commandes universel de gestionnaire de réseau	917
Tableau J.1 – Réponses de commande requises	931
Tableau J.2 – Fanions de statut de la passerelle	934
Tableau J.3 – Lignes directrices pour les capacités de la passerelle.....	935

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

**RÉSEAUX DE COMMUNICATIONS INDUSTRIELS –
RÉSEAU DE COMMUNICATIONS SANS FIL
ET PROFILS DE COMMUNICATION –
WirelessHART™**

AVANT-PROPOS

- 1) La Commission Electrotechnique Internationale (CEI) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, la CEI – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de la CEI"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec la CEI, participent également aux travaux. La CEI collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
- 2) Les décisions ou accords officiels de la CEI concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de la CEI intéressés sont représentés dans chaque comité d'études.
- 3) Les Publications de la CEI se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de la CEI. Tous les efforts raisonnables sont entrepris afin que la CEI s'assure de l'exactitude du contenu technique de ses publications; la CEI ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
- 4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de la CEI s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de la CEI dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de la CEI et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
- 5) La CEI elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de la CEI. La CEI n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
- 6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
- 7) Aucune responsabilité ne doit être imputée à la CEI, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de la CEI, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de la CEI ou de toute autre Publication de la CEI, ou au crédit qui lui est accordé.
- 8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.

La Norme internationale CEI 62591 a été établie par le sous-comité 65C: Réseaux de communications industrielles du comité d'études 65 de la CEI: Mesure, commande et automation dans les processus industriels.

La présente norme annule et remplace le IEC/PAS 62591 publiée en 2009. Cette première édition constitue une révision technique.

La présente version bilingue, publiée en 2011-07 correspond à la version anglaise.

Le texte anglais de cette norme est issu des documents 65C/587/FDIS et 65C/595/RVD.

Le rapport de vote 65C/595/RVD donne toute information sur le vote ayant abouti à l'approbation de cette Norme.

La version française de cette norme n'a pas été soumise au vote.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 2.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de la CEI sous «<http://webstore.iec.ch>» dans les données relatives à la publication recherchée. A cette date, la publication sera

- reconduite,
- supprimée,
- remplacée par une édition révisée, ou
- amendée.

IMPORTANT – Le logo "colour inside" qui se trouve sur la page de couverture de cette publication indique qu'elle contient des couleurs qui sont considérées comme utiles à une bonne compréhension de son contenu. Les utilisateurs devraient, par conséquent, imprimer cette publication en utilisant une imprimante couleur

INTRODUCTION

La présente Norme internationale fournit la spécification, les définitions et le profil pour un réseau de communications sans fil. Elle complète la IEC 61158-5-20, la IEC 61158-6-20 où un certain nombre de commandes sont spécifiées et elle complète la IEC 61784-1 où une commande universelle de profil de communication CP (Communication Profile) 9/1, est spécifiée.

Le présent document suit la structure et les conventions de la série IEC 61158 (par exemple pour la séparation des définitions de service DL et la spécification du protocole DL) ainsi que les conventions de la IEC 61784-1. La série IEC 61158 spécifie différents réseaux de communication. Ceux-ci sont structurés en différents Types. Le Type 20 est attribué aux technologies de la Hart™¹ Communication Foundation (HCF). Pour les autres affectations des numéros de Type, voir IEC/TR 61158-1.

NOTE La série IEC 61158 ne contient pas de spécification de DLL pour le Type 20. Cependant, la présente norme inclut une spécification de DLL.

La IEC 61784-1 et la IEC 61784-2 fournissent des familles de profils de communication (CPF) et, au sein d'une famille, un à n Profils de communication. Le numéro de CPF attribué pour les technologies de la HCF est CPF 9. Pour les autres attributions de numéros de CPF, voir IEC/TR 61158-1.

Un nouveau nom de projet CEI 62591 a été attribué aux améliorations du Type 20 et au CP 9/2 associé et, de ce fait, le présent document contient les sous-parties équivalentes de la série IEC 61158 et de la IEC 61784-1 organisées ici dans différents articles.

Le protocole du Type 20 prend en charge des communications numériques bidirectionnelles pour des dispositifs de mesure et de commande de processus. Les applications comprennent l'interrogation de variables de processus, l'accès cyclique aux données de processus, le réglage de paramètres et le diagnostic, et ce, à distance. Le présent document définit la spécification qui comprend le protocole de communications de terrain du Type 20 pour les dispositifs sans fil. La spécification du protocole de Type 20 repose largement sur le Modèle de communications OSI à sept (7) couches.

La Commission Électrotechnique Internationale (CEI) attire l'attention sur le fait qu'il est déclaré que la conformité avec les dispositions du présent document peut impliquer l'utilisation d'un brevet concernant les revendications des brevets énumérés ci-après dans les Articles normatifs.

US08/04750 & PCT 12/100986 [HCF] Combined Wired and Wireless Communications with Field Devices
US08/04777 & PCT 12/100927 [HCF] Increasing Reliability and Reducing Latency in a Wireless Network
US08/04677 & PCT 12/100995 [HCF] Adaptive Scheduling in a Wireless Network
US08/04716 & PCT 12/101000 [HCF] Wireless Protocol Adaptor
US08/04775 & PCT 12/101005 [HCF] Synchronizing Timeslots in a Wireless Communication Protocol
US08/04751 & PCT 12/101011 [HCF] Suspending Transmissions in a Wireless Communication Network
US08/04678 & PCT 12/101021 [HCF] Enhancing Security in a Wireless Network
US08/04746 & PCT 12/101037 [HCF] Wireless Gateway in a Process Control Environment
US08/04676 & PCT 12/101043 [HCF] Efficient Addressing in Wireless HART Protocol

¹ HART™ et WirelessHART™ sont les appellations commerciales de la HART™ Communication Foundation (HCF). HCF est une organisation à but non lucratif appuyant la HART™ Communication. Cette information est donnée à l'intention des utilisateurs de la présente Norme internationale et ne signifie nullement que la CEI approuve ou recommande l'emploi de produits de ces marques. La conformité au présent document n'exige pas l'utilisation de l'appellation commerciale. L'utilisation des appellations commerciales HART™ et WirelessHART™ requiert la permission du détenteur de l'appellation concernée.

US08/04745 & PCT 12/101049 [HCF] Support for Network Management and Device Communications in a Wireless Network

US08/04749 & PCT 12/101054 [HCF] Priority Based Scheduling and Routing in a Wireless Network

US08/04740 & PCT 12/101071 [HCF] Routing Packets on a Network Using Directed Graphs

US08/04761 & PCT 12/101074 [HCF] Scheduling Communication Frames in a Wireless Network

La CEI ne prend pas position quant à la preuve, à la validité et à la portée de ces droits de propriété.

Le détenteur de ces droits de propriété a donné l'assurance à la CEI qu'il consent à négocier des licences avec des demandeurs du monde entier, soit sans frais soit à des termes et conditions raisonnables et non discriminatoires. À ce propos, la déclaration du détenteur des droits de propriété est enregistrée à la CEI. Des informations peuvent être demandées à:

[HCF] Hart Communication Foundation
9390 Research Boulevard
Suite I-350
Austin TX 78759, USA

L'attention est d'autre part attirée sur le fait que certains des éléments du présent document peuvent faire l'objet de droits de propriété autres que ceux qui ont été mentionnés ci-dessus. La CEI ne saurait être tenue pour responsable de l'identification de ces droits de propriété en tout ou partie.

L'ISO (www.iso.org/patents) et la CEI (http://www.iec.ch/tctools/patent_decl.htm) maintiennent des bases des données, consultables en ligne, des droits de propriété pertinents à leurs normes. Les utilisateurs sont encouragés à consulter ces bases de données pour obtenir l'information la plus récente concernant les droits de propriété.

RÉSEAUX DE COMMUNICATIONS INDUSTRIELS – RÉSEAU DE COMMUNICATIONS SANS FIL ET PROFILS DE COMMUNICATION – WirelessHART™

1 Domaine d'application

La présente Norme internationale spécifie un réseau de communication type 20 supplémentaire en ajout à la IEC 61158-5-20 et à la IEC 61158-6-20 ainsi qu'un Profil de communication CP 9/2 en ajout à la famille de profils de communications CPF 9 de la IEC 61784-1.

La présente norme spécifie ce qui suit:

- Définition de service de la couche physique et spécification de protocole,
- Service et protocole de la couche liaison de données,
- Service et protocole de la couche application,
- Gestion de réseau
- Sécurité,
- Profil de communication,
- Procédures sans fil et
- Passerelle.

2 Références normatives

Les documents de référence suivants sont indispensables pour l'application du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

CEI 60559, *Arithmétique binaire en virgule flottante pour systèmes à microprocesseurs*

IEC/TR 61158-1:2010, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-5-20:2010, *Industrial communication networks – Fieldbus specifications – Part 5-20: Application layer service definition – Type 20 elements*

IEC 61158-6-20:2010, *Industrial communication networks – Fieldbus specifications – Part 6-20: Application layer protocol specification – Type 20 elements*

IEC 61784-1:2010, *Industrial Communication Networks – Profiles – Part 1: Fieldbus profiles*

IEC 61784-2:2010, *Industrial Communication Networks – Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3*

ISO/CEI 7498-1, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base: Le modèle de base*

ISO/IEC 8824, *Technologies de l'information – Interconnexion de systèmes ouverts – Spécification de la notation de syntaxe abstraite numéro 1 (ASN.1)* (disponible uniquement en anglais)

ISO/IEC 8859-1, *Technologies de l'information – Jeux de caractères graphiques codés sur un seul octet – Partie 1: Alphabet latin n°. 1* (disponible uniquement en anglais)

ISO/CEI 9545, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Structure de la couche application*

ISO/CEI 10731, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de Référence de Base – Conventions pour la définition des services OSI*

IEEE 802-2001, *IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture*

IEEE 802.15.4-2006, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*

3 Termes, définitions, symboles, abréviations et conventions

Pour les besoins du présent document, les termes suivants tels que définis dans ces publications s'appliquent.

3.1 Termes et définitions du modèle de référence

La présente norme est basée en partie sur les concepts développés dans l'ISO/CEI 7498-1 et l'ISO/CEI 7498-3 et utilise les termes suivants y définis.

entité DL (N=2)	[7498-1]
DL protocol-data-unit	[7498-1]
DL service-data-unit	[7498-1]
DLS user-data	[7498-1]
entité NL (N=3)	[7498-1]
NL protocol-data-unit	[7498-1]
NL service-data-unit	[7498-1]
NLS user-data	[7498-1]
TL-entity (N=4)	[7498-1]
TL protocol-data-unit	[7498-1]
TL service-data-unit	[7498-1]
TLS user-data	[7498-1]

3.2 Termes et définitions issus de l'ISO/CEI 7498-1

- a) syntaxe abstraite
- b) entité d'application
- c) processus d'application
- d) unité de données de protocole application
- e) élément de service application
- f) invocation d'entité d'application
- g) invocation de processus d'application
- h) transaction d'application
- i) contexte de présentation
- j) système ouvert réel
- k) syntaxe de transfert

3.2.1 Termes et définitions issus de l'ISO/CEI 9545

- a) association d'applications (application-association)
- b) contexte d'application (application-context)
- c) nom de contexte d'application (application context name)
- d) invocation d'entité d'application (application-entity-invocation)
- e) type d'entité d'application (application-entity-type)
- f) invocation de processus d'application (application-process-invocation)
- g) type de processus d'application (application-process-type)
- h) éléments de service application (application-service-element)
- i) élément de service de contrôle d'application (application control service element)

3.2.2 Termes et définitions issus de l'ISO/IEC 8824

- a) identificateur d'objet
- b) type
- c) valeur
- d) type simple
- e) type structuré
- f) type de composant
- g) tag (marqueur)
- i) true (vrai)
- j) false (faux)
- k) type integer (entier)
- m) type octet string (chaîne d'octets)
- n) type null
- o) type sequence (séquence)
- p) sequence of type (séquence de type)
- q) type choice (choix)
- r) type tagged (marqué)
- s) any type (tout type)
- t) module
- u) production

3.2.3 Termes et définitions issus de l'ISO/IEC 8825

- a) encodage (d'une valeur de donnée)
- b) valeur de donnée
- c) octets d'identificateur (le singulier est utilisé dans la présente norme)
- d) octet(s) de longueur (le singulier et le pluriel sont utilisés dans la présente norme)
- e) octets de contenu

3.3 Termes et définitions spécifiques

3.3.1

nombre d'intervalles absolus

nombre de tous les intervalles de temps qui se sont produits depuis que le réseau a été formé

NOTE Ce nombre est toujours incrémenté et n'est jamais remis à un nombre fixe quelconque ou à zéro. Sa valeur courante est toujours le numéro de séquence de l'intervalle de temps courant. Sa valeur maximale est $(2^{40} - 1)$ après quoi il repasse par zéro.

3.3.2

acquitter

donner une réponse explicite de liaison de données pour la réception réussie d'une DLPDU adressée et non diffusée à partir d'un dispositif source de DLE et la seconde DLPDU d'une transaction à deux DLPDU

3.3.3

adaptateur

dispositif de réseau qui permet la connexion et la communication avec un dispositif de terrain qui n'a pas de connexion directe à un réseau du Type 20

3.3.4

DLPDU avertise

DLPDU avertissant de nouveaux dispositifs dans le réseau

NOTE Lorsqu'un dispositif souhaite se rattacher à un réseau, il se met à l'écoute pour détecter ces DLPDU et utilise alors l'information contenue dans la DLPDU pour se synchroniser au réseau et lancer le processus de rattachement.

3.3.5

voie analogique

signal électrique à variations continues reliant un dispositif de terrain au reste du système d'acquisition ou de contrôle de données

NOTE 1 Certains dispositifs de terrain prennent en charge plusieurs voies analogiques (entrée ou sortie).

NOTE 2 Chaque voie analogique émet une seule variable dynamique à destination ou en provenance du dispositif de terrain.

3.3.6

application

fonction ou structure de données pour laquelle les données sont consommées ou produites

[IEC 61158-6-9]

3.3.7

domaine d'application

classification du processus d'application reposant sur l'exigence relative à la priorité et à la capacité de communication

3.3.8

objet d'application

classe d'objets qui gère et fournit l'échange de messages pendant le mode exécution à travers le réseau et à l'intérieur du dispositif de réseau

NOTE Plusieurs types de classes d'objets d'application peuvent être définis.

[IEC 61158-6-9]

3.3.9

relation d'applications

association coopérative entre deux ou plusieurs invocations d'entités d'application (application-entity-invocation) à des fins d'échange d'informations et de coordination de leur fonctionnement conjoint

NOTE Cette relation est activée soit par l'échange d'unités de données de protocole d'application, soit suite à des activités de pré configuration.

[IEC 61158-6-9]

3.3.10**point d'extrémité de relation d'applications**

contexte et conduite d'une relation d'applications tels que vus et maintenus par l'un des processus d'application impliqués dans la relation d'applications

NOTE Chaque processus d'application impliqué dans la relation d'applications maintient son propre point d'extrémité de relation d'applications.

3.3.11**temps ASN**

temps représenté sous la forme de nombre d'intervalles de temps absolus; toute instance dans le temps est égale au nombre d'intervalles de temps absolus en l'instance en question

3.3.12**attribut**

description d'une caractéristique visible de l'extérieur d'un objet

NOTE Les attributs d'un objet contiennent de l'information relative à des parties variables d'un objet. Typiquement, ils fournissent des informations relatives au statut ou régissent le fonctionnement d'un objet. Des attributs peuvent aussi avoir une incidence sur le comportement d'un objet. Les attributs se répartissent en attributs de classes et attributs d'instances.

3.3.13**comportement**

indication de la façon dont un objet réagit à des événements particuliers

NOTE Sa description comprend la relation entre les valeurs d'attributs et les services.

3.3.14**diffusion**

processus d'envoi d'une PDU à tous les dispositifs qui sont reliés au réseau et sont capables de recevoir la transmission

3.3.15**adresse de diffusion**

adresse utilisée par un maître pour envoyer une commande à tous les dispositifs

3.3.16**occupé**

état d'un dispositif qui est actif sur différentes tâches et ne peut pas exécuter une commande à l'instant en question

3.3.17**voie**

bande de fréquences RF utilisée pour transmettre un signal modulé transportant des paquets

3.3.18**mise de voie en liste noire**

méthode pour mettre une voie RF hors usage

3.3.19**saut de voie**

variation régulière de la fréquence d'émission ou de réception pour combattre les perturbations et les évanouissements

3.3.20**décalage de voies**

valeur spécifique à la liaison fournie par le gestionnaire de réseau qui est utilisée pour calculer la voie à utiliser lors du saut de voie

3.3.21

classe

ensemble d'objets, qui représentent tous le même type de composant système

NOTE Une classe est une généralisation de l'objet; un modèle pour définir des variables et des méthodes. Tous les objets dans une classe ont une forme et un comportement identiques, mais contiennent en général des données différentes dans leurs attributs

3.3.22

attribut de classe

attribut qui est partagé par tous les objets au sein de la même classe

3.3.23

code de classe

identificateur unique attribué à chaque classe d'objets

3.3.24

service spécifique à une classe

service défini par une classe d'objets particulière pour accomplir une fonction requise qui n'est pas accomplie par un service commun

NOTE Un objet spécifique à une classe est unique pour la classe d'objets qui le définit.

3.3.25

évaluation de voie libre

ensemble d'essais effectués pendant que la voie RF est utilisée afin d'éviter de lancer une transaction

NOTE Elle est accomplie en écoutant la voie avant d'envoyer la première DLPDU d'une transaction. Si un signal est détecté, la transaction est différée jusqu'à un intervalle de temps TDMA plus tardif.

3.3.26

client

- a) objet qui utilise les services d'un autre objet (serveur) pour accomplir une tâche
- b) émetteur d'un message auquel un serveur réagit, tel que le rôle d'un point d'extrémité d'AR dans lequel il émet des APDU de demande de service confirmées vers un seul point d'extrémité d'AR comme serveur

[IEC 61158-6-9]

3.3.27

trajet d'acheminement

flux unidirectionnel d'unités APDU à travers une relation d'applications

3.3.28

cyclique

terme utilisé pour décrire des événements qui se produisent d'une manière régulière et répétitive

3.3.29

dispositif

entité contenant une mise en œuvre d'un réseau de terrain de Type 20

3.3.30

ID de dispositif

numéro de série pour un dispositif qui est unique parmi toutes les instances d'un même type de dispositif

3.3.31**profil de dispositif**

ensemble d'informations et de fonctionnalité dépendant du dispositif qui assure la cohérence entre des dispositifs similaires relevant du même type de dispositif

3.3.32**type de dispositif étendu**

type selon le fabricant d'un dispositif tel que spécifié dans la IEC 61158-6-20:2010, Tableau 7

3.3.33**variable de dispositif**

donnée définie de manière unique au sein d'un dispositif de terrain qui est toujours associée à une information de processus cyclique

NOTE Une valeur de variable de dispositif varie en fonction des changements et des variations du processus auquel le dispositif est relié.

3.3.34**découverte**

processus consistant à trouver un nouveau voisin en écoutant la DLPDU spéciale qui indique la présence du dispositif

3.3.35**liaison de découverte**

liaison qui est utilisée pour émettre et recevoir la DLPDU spéciale qui indique la présence du dispositif

3.3.36**variable dynamique**

connexion entre un processus et une voie analogique

NOTE Un dispositif peut contenir des variables primaires, secondaires, tertiaires et quaternaires. Celles-ci sont collectivement appelées les "variables dynamiques".

3.3.37**point d'extrémité**

l'une des entités en communication impliquées dans une connexion

3.3.38**erreur**

discordance entre une valeur ou un état calculé(e), observé(e) ou mesuré(e) et la valeur ou l'état spécifié(e) ou théoriquement correct(e)

3.3.39**code d'erreur**

identification d'un type spécifique d'erreur dans une classe d'erreurs

3.3.40**dispositif de terrain**

dispositif de réseau qui est relié au processus ou à l'équipement de l'installation

NOTE Il se connecte directement au capteur ou actionneur ou accomplit une fonction de commande de processus et il est directement connecté à la couche physique de Type 20.

3.3.41**voie de fréquence**

attribution du spectre de fréquences dans une bande de fréquences donnée

3.3.42

passerelle

dispositif de réseau contenant au moins une interface d'hôte telle qu'une interface série ou Ethernet, faisant office de point d'entrée ou de point de sortie permettant la communication entre des applications d'hôtes et des dispositifs de terrain

3.3.43

graphe

ensemble de trajets orientés parmi des dispositifs de réseau tel que deux chemins au moins soient possibles pour aller d'un dispositif source vers un dispositif de destination

3.3.44

Graph_ID

identificateur utilisé pour indiquer une entrée de graphe spécifique

3.3.45

tenu à la main

- a) application d'hôte résidant sur un dispositif portatif
- b) qualificatif appliqué à d'autres termes tels que clé, super trame

3.3.46

saut

mouvement d'un paquet directement entre deux voisins adjacents dans une transaction de réseau sans la participation d'aucun autre nœud du réseau

3.3.47

nombre de sauts de vie résiduelle

champ dans l'en-tête réseau de chaque paquet qui spécifie le nombre de sauts résiduels qu'un paquet peut parcourir avant d'être rejeté

3.3.48

rattachement

processus par lequel un dispositif de réseau est authentifié et autorisé à participer au réseau

NOTE Un dispositif est considéré rattaché ("Joined") lorsqu'il a la clé du réseau, une session de gestionnaire de réseau et une super trame normale (de non-rattachement) et des liaisons.

3.3.49

join_key

clé de sécurité utilisée pour démarrer le processus de rattachement

3.3.50

latence

temps mis par un paquet pour franchir la connexion réseau, de l'expéditeur vers le destinataire

3.3.51

liaison

pleine spécification de communication entre dispositifs adjacents dans un réseau et comprenant les paramètres de communication nécessaires pour déplacer une DLPDU d'un saut

NOTE Une liaison est une fonction d'appariement d'adresses de source et de destination, d'affectation de décalages d'intervalle de temps et de voie, de sens de communication, de communication spécialisée ou partagée, et de type. Des liaisons sont attribuées à des super trames comme partie intégrante du processus de programmation.

3.3.52**commande de liaison logique**

la plus élevée des deux sous-couches de la couche liaison de données et utilisée pour la détection d'erreur, le contrôle de débit, le verrouillage de trame et l'adressage

3.3.53**long_tag**

chaîne restreinte à 32 caractères du jeu de l'alphabet latin 1 de l'ISO servant à identifier un dispositif de terrain

3.3.54**courant de boucle**

valeur mesurée en mA en série avec le dispositif de terrain

NOTE Le courant de boucle est un signal analogique quasi continu de 4 mA à 20 mA utilisé pour communiquer une seule valeur entre le système de commande et le dispositif de terrain. Les dispositifs en mode tension utilisent des "Volts CC" comme unités d'étude lorsque des valeurs de "courant de boucle" sont utilisées.

3.3.55**port de maintenance**

interface dans le dispositif réseau qui est utilisé pour l'approvisionnement

NOTE Cette interface est utilisée pour écrire la "Join_key" et l'identificateur réseau et pour surveiller le processus de rattachement.

3.3.56**informations de gestion**

informations accessibles par le réseau qui appuient la gestion du fonctionnement du système de réseau de terrain, y compris la couche application

NOTE La gestion inclut les fonctions telles que la commande, la surveillance et le diagnostic.

3.3.57**ID de fabricant**

énumération de deux octets identifiant le fabricant qui a produit un dispositif

NOTE Un fabricant est tenu d'utiliser la valeur qui lui est attribuée et n'est pas autorisé à utiliser la valeur attribuée à un autre fabricant.

3.3.58**maître**

dispositif qui lance une activité de communication en envoyant une APDU de demande à un dispositif et en attendant une PDU de réponse

3.3.59**contrôle d'accès au support**

la plus basse des deux sous-couches de liaison de données et utilisée pour contrôler l'accès à la voie de communication

3.3.60**multidiffusion**

envoi d'un paquet à plus d'un dispositif sur le réseau en utilisant une seule émission

3.3.61**voisin**

nœud adjacent dans le réseau tel que le niveau de signal reçu (RSL²) qui en est issu suggère que la communication est possible dans au moins un sens

² Les abréviations sont données en 3.4.

3.3.62

tableau de voisins

liste de tous les dispositifs que le dispositif considère comme voisins

NOTE Il conserve aussi les attributs du voisin.

3.3.63

réseau

série de nœuds reliés par quelque type de support de communication

NOTE Les trajets de connexion entre les nœuds d'une paire quelconque peut comprendre des répéteurs, routeurs et passerelles.

3.3.64

dispositif de réseau

dispositif ayant une connexion de couche physique directe au réseau

NOTE Chaque dispositif de réseau a une adresse unique qui est utilisée dans la communication avec le dispositif. Les dispositifs de réseau incluent dispositif de terrain, point d'accès (à savoir passerelle), adaptateur et appareil tenu à la main.

3.3.65

network_ID

identificateur utilisé pour indiquer un réseau auquel sont raccordés tous les dispositifs d'intercommunication

3.3.66

gestionnaire de réseau

entité qui est responsable de la configuration du réseau, de programmer la communication entre les dispositifs de réseau, de la gestion des tableaux de routage et de surveiller et rapporter la santé du réseau

3.3.67

surnom

identificateur attribué à un dispositif tel qu'il soit unique au sein du réseau auquel le dispositif est connecté

3.3.68

nœud

dispositif logique ou physique adressable attaché au réseau

3.3.69

nonce

nombre non répétitif dont la construction le rend unique au paquet courant et assure que d'anciennes communications ne peuvent pas être réutilisées dans des attaques par jeu

NOTE Le nonce est nécessaire pour maintenir la sécurité des paquets et assurer l'authenticité de l'expéditeur et l'intégrité des paquets.

3.3.70

paquet

bits agrégés et formatés qui sont émis ensemble simultanément à travers le support physique

[IEEE 802.15.4, modifiée]

3.3.71

trajet

liaison de communication directe entre deux dispositifs voisins dans le réseau maillé

3.3.72**charge utile de données**

contenu d'un message de données qui est en cours d'émission

3.3.73**tube**

connexion de communication virtuelle, sécurisée et bidirectionnelle entre dispositifs homologues

3.3.74**adresse d'interrogation**

nombre entier utilisé pour identifier un dispositif

NOTE L'adresse d'interrogation est utilisée pour construire une adresse d'un seul octet.

3.3.75**point d'extrémité d'AR prédéfini**

point d'extrémité d'AR qui est défini localement dans le dispositif sans utilisation du service création ("create")

NOTE Les relations AR prédéfinies qui ne sont pas préétablies sont établies avant d'être utilisées.

3.3.76**point d'extrémité d'AR préétabli**

point d'extrémité d'AR qui est mis dans un état établi pendant la configuration des AE qui commandent ses points d'extrémité

3.3.77**chemin**

séquence de trajets unidirectionnels d'un dispositif à un autre reliant un dispositif source à un dispositif de destination

3.3.78**ID de chemin (route ID)**

identificateur utilisé pour indiquer un chemin spécifique

3.3.79**serveur**

<communication> rôle d'un AREP dans lequel il retourne une APDU de réponse de service confirmée au client qui a émis la demande

3.3.80**serveur**

<système> objet qui fournit des services à un autre objet (client)

3.3.81**service**

opération ou fonction accomplie par un objet et/ou une classe d'objets à la demande d'un autre objet et/ou d'une autre classe d'objets

NOTE Un jeu de services communs est défini et des dispositions pour la définition de services spécifiques à l'objet sont fournies. Les services spécifiques à un objet sont ceux qui sont définis par une classe d'objets particulière pour accomplir une fonction requise qui n'est pas accomplie par un service commun.

3.3.82**ID de session (session ID)**

identificateur utilisé pour indiquer une entrée de session spécifique

3.3.83

esclave

dispositif qui déclenche une activité de communication seulement après qu'il a reçu une PDU de demande provenant d'un dispositif maître et il est tenu d'envoyer une réponse à la demande en question

3.3.84

intervalle de temps

durée fixe qui peut être utilisée pour la communication entre voisins

3.3.85

super frame (superframe)

ensemble d'intervalles de temps qui se répètent à une fréquence constante; chaque intervalle de temps a une liaison qui lui est associée

3.3.86

superframe_ID

identificateur utilisé pour indiquer une entrée de super frame spécifique

3.3.87

marqueur

chaîne ASCII de huit caractères utilisée pour identifier un dispositif de terrain

3.3.88

accès multiple par répartition dans le temps

technique de contrôle d'accès au support qui utilise des intervalles de temps lorsque les communications entre dispositifs peuvent avoir lieu

NOTE Il assure des communications déterministes sans collision.

3.3.89

transaction

échange de trames consécutives connexes entre deux entités homologues de contrôle d'accès au support, requis pour une émission réussie

NOTE Une transaction est constituée soit (a) d'une émission de PhPDU à partir d'un dispositif source, soit (b) d'une PhPDU provenant du dispositif source suivie d'une seconde PhPDU d'acquiescement de niveau liaison provenant du dispositif de destination.

3.3.90

monodiffusion

envoi d'un paquet vers un seul dispositif dans le réseau

3.3.91

ID propre (ID unique)

identificateur attribué à un dispositif qui est unique parmi toutes les instances des dispositifs conformes à la présente norme et conformes au Profil de communication CP 9/2 utilisant un nom d'organisation identique comme spécifié en 9.3.1

3.4 Abréviations

ACK	Acknowledge (Acquiescer)
AE	Application entity (Entité d'application)
AL	Application layer (Couche application)
AP	Application process (Processus d'application)
APDU	Application protocol data unit (Unité de donnée de protocole application)
APO	Application Object (Objet d'application)

AR	Application relationship (Relation d'applications)
AREP	Application relationship endpoint (Point d'extrémité de relation d'applications)
ARPM	Application Relationship Protocol Machine (Machine protocolaire de relation d'applications)
ASCII	American Standard Code for Information Interchange (Code américain normalisé pour l'échange d'information)
ASE	Application Service Element (Élément de service application)
ASN	Absolute Slot Number (Nombre d'intervalles absolus)
BACK	Burst acknowledge (Acquittement de rafale)
CCA	Clear channel assessment (Évaluation de voie libre)
Cnf	Confirmation
CP	Communication Profile (Profil de communication)
CPF	Communication Profile Family (Famille de profils de communication)
CRC	Contrôle de redondance cyclique
dBm	Abréviation pour le rapport de puissance en décibels (dB) de la puissance mesurée, référencé à 1 mW (1 mW), 0 dBm = 1 mW, 10 dBm = 10 mW, 20 dBm = 100 mW, 30 dBm = 1 W
DL-	Data-link layer (Couche liaison de données) (comme préfixe)
DLE	DL-entity (entité DL) (l'instance locale active de la couche liaison de données)
DLL	Data link layer (Couche Liaison de données)
DLM	DL-management (Gestion DL)
DLMS	DL-management service (Service de gestion DL)
DLPDU	Data-link protocol data unit (Unité de donnée de protocole de liaison de données)
DLS	DL-service (Service DL)
DLSDU	DL-service-data-unit (unité de donnée de service DL)
DR	Delayed response (Réponse différée)
DRM	Delayed response mechanism (Mécanisme de réponse différée)
EUI-64	Extended unique identifier (Identificateur propre étendu) (long de 64 bits)
FAL	Fieldbus application layer (Couche application de réseau de terrain)
FSMP	FAL Service Protocol Machine (Machine protocolaire de services FAL)
HCF	HART™ Communication Foundation
HTL	Hop to live (nombre de sauts de vie résiduelle)
ID	Identificateur
CEI	Commission Électrotechnique Internationale
Ind	Indication
UIT-T	Union Internationale des Télécommunications – Secteur de la Normalisation des Télécommunications
LLC	Logical link control (commande de liaison logique)
LSB	Least significant byte (Octet de poids faible)
MAC	Medium access control (Contrôle d'accès au support)
MIC	Message integrity code (Code d'intégrité de message)
MSB	Most significant byte (Octet de poids fort)

NL	Network layer (Couche réseau)
NL-	Network layer (de couche réseau) (comme préfixe)
NLE	NL-entity (entité NL) (l'instance locale active de la couche réseau)
NLM	NL-management (gestion réseau)
NLMS	NL-management service (Service de gestion réseau)
NLS	NL-service (Service NL)
NLSDU	NL-service-data-unit (Unité de donnée de service NL)
NPDU	NL-protocol-data-unit (Unité de donnée de protocole NL)
NSDU	NL-service-data-unit (Unité de donnée de service NL)
O-QPSK	Offset Quadrature Phase-Shift Keying (Modulation par déplacement de phase quadrivalente décalée)
OSI	Open Systems Interconnect (Interconnexion de systèmes ouverts)
OUI	Organizationally Unique Identifier (Identificateur propre à une organisation)
PDU	Protocol data unit (Unité de données de protocole)
PhE	PhL-entity (Entité PhL) (l'instance locale active de la couche physique)
PhL-	Physical layer (de couche physique) (comme préfixe)
PhPDU	PhL-protocol-data-unit (Unité de données de protocole PhL) [ISO/CEI 7498-1]
PhS	Physical layer service (Service de couche physique) [ISO/CEI 7498-1]
QoS	Quality-of-service (Qualité de service)
req	Request (Demande)
RF	Radiofréquence
RSL	Receive signal level (Niveau de signal reçu)
Rsp	Response (Réponse)
RSV	Réservé
SAP	Service access point (Point d'accès au service)
SN	Sign bit (Bit de signe)
SOM	Start of message (Début de message)
STX	Start of a transaction (Début d'une transaction)
TDMA	Time Division Multiple Access (Accès multiple par répartition dans le temps, AMRT)
TL	Transport layer (Couche transport)
TL-	Transport layer (de couche transport) (comme préfixe)
TLE	TL-entity (Entité TL) (l'instance locale active de la couche transport)
TLM	TL-management (gestion TL)
TLMS	TL-management service (Service de gestion TL)
TLS	TL-service (Service TL)
TMPM	Transport layer Mapping Protocol Machine (Machine protocolaire de mappage de couche transport)
TPDU	TL-protocol-data-unit (Unité de donnée de protocole TL)
TSDU	TL-service-data-unit (Unité de donnée de service TL)
VFD	Virtual field device (Dispositif de terrain virtuel)

3.5 Conventions

3.5.1 Conventions utilisées pour le service

La présente norme utilise les conventions descriptives données dans l'ISO/CEI 10731.

Le modèle de service, les primitives de service et les diagrammes de séquence temporelle utilisés sont des descriptions totalement abstraites; ils ne constituent pas une spécification pour une mise en œuvre.

Les primitives de service, utilisées pour représenter les interactions utilisateur de service/fournisseur de service (voir ISO/CEI 10731), acheminent des paramètres qui indiquent les informations disponibles dans l'interaction entre utilisateur et fournisseur.

La présente norme utilise un format de tableau pour décrire les paramètres de composants des primitives de service des couches Physique, Liaison de données, Réseau, Transport et Application. Les paramètres qui s'appliquent à chaque groupe de primitives sont consignés en tableaux dans toute la suite de la présente norme. Chaque tableau comporte jusqu'à six colonnes, contenant le nom du paramètre de service, et une colonne chacune pour les primitives et les sens de transfert de paramètres utilisés par le service:

- les paramètres d'entrée de la primitive de demande ("request");
- les paramètres de sortie de la primitive de demande ("request");
- les paramètres de sortie de la primitive d'indication;
- les paramètres d'entrée de la primitive de réponse ("response"); et
- les paramètres de sortie de la primitive de confirmation ("confirm").

Un paramètre (ou une partie de celui-ci) est énuméré dans chaque rangée de chaque tableau. Pour les services locaux, seule est utilisée la primitive "request" avec paramètres d'entrée et de sortie. Dans les colonnes appropriées de la primitive de service, un code est utilisé pour spécifier le type d'usage du paramètre sur la primitive et le sens de paramètres spécifiés dans la colonne:

M Le paramètre est obligatoire pour la primitive;

U Le paramètre est une option de l'utilisateur et peut ou peut ne pas être fourni, cela dépendant de l'usage dynamique de l'utilisateur DLS. Lorsqu'il n'est pas fourni, une valeur par défaut est supposée pour le paramètre;

S Un seul paramètre dans un groupe de paramètres utilisant ce code est requis.

C Le paramètre est conditionné à d'autres paramètres ou à l'environnement de l'utilisateur DLS;

CU Le paramètre est une option utilisateur conditionnelle et peut ou peut ne pas être autorisé, cela dépendant d'autres paramètres ou de l'environnement de l'utilisateur DLS. Lorsqu'il est autorisé, il peut ou peut ne pas être fourni, cela dépendant de l'usage dynamique de l'utilisateur DLS. Lorsqu'il est autorisé mais n'est pas fourni, une valeur par défaut est supposée pour le paramètre.

(blanc/vide) Le paramètre n'est jamais présent.

Certaines entrées sont en plus qualifiées par des éléments désignés entre parenthèses. Ceux-ci peuvent être :

a) une contrainte spécifique au paramètre

- (=) indique que le paramètre équivaut du point de vue de la sémantique au paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau;
- (≤) indique que le jeu de valeurs de paramètres a un ordre implicite et que la valeur du paramètre est inférieure ou égale à celle du paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau (c'est-à-dire à gauche d'une ou deux colonnes);

- (≥) indique que le jeu de valeurs de paramètres a un ordre implicite et que la valeur du paramètre est supérieure ou égale à celle du paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau (c'est-à-dire à gauche d'une ou deux colonnes);
- b) une indication qu'une certaine note s'applique à l'entrée;
- (n) indique que la note n suivante contient des informations complémentaires relatives au paramètre et à son utilisation.

3.5.2 Conventions utilisées pour le protocole DL (diagramme d'état)

La présente norme utilise un diagramme pour représenter un diagramme d'état. La convention suivante est utilisée.

- Les rectangles étiquetés représentent les états dans lesquels un composant peut se trouver.
- Les transitions d'état sont des lignes orientées. Elles montrent l'état qu'un composant quitte et l'état auquel il passe.
- Une transition est étiquetée avec les événements qui l'ont provoquée et les actions correspondantes (elle peut être vide). Les événements (au-dessus de la ligne) provoquant la transition sont séparés des actions résultantes (en dessous de la ligne) par une ligne horizontale.
- Un tilde (" ~ ") signifie le complément à un ou la négation.

3.5.3 Conventions pour le service Couche Application

3.5.3.1 Vue d'ensemble

La Couche Application de réseau de terrain (FAL) est définie comme étant un jeu d'éléments de service Application (ASE) orientés objets. Chaque ASE est spécifié dans un paragraphe distinct. Chaque spécification d'ASE est constituée de deux parties, à savoir sa spécification de classe et sa spécification de service.

La spécification de classe définit les attributs de la classe. La spécification de service définit les services qui sont fournis par l'ASE.

3.5.3.2 Conventions pour les définitions de classes

Les définitions de classes sont décrites à l'aide de modèles. Chaque modèle est constitué d'une liste d'attributs de la classe. Un exemple du modèle est montré ci-dessous:

FAL ASE:		Nom de l'ASE	
CLASS:		Nom de la classe	
CLASS ID:		#	
PARENT CLASS:		Nom de la classe parent	
ATTRIBUTES:			
1	(o)	Attribut clé:	identificateur numérique
2	(o)	Attribut clé:	nom
3	(m)	Attribut:	nom d'attribut(valeurs)
4	(m)	Attribut:	nom d'attribut(valeurs)
4.1	(s)	Attribut:	nom d'attribut(valeurs)
4.2	(s)	Attribut:	nom d'attribut(valeurs)
4.3	(s)	Attribut:	nom d'attribut(valeurs)
5.	(c)	Contrainte:	expression de la contrainte
5.1	(m)	Attribut:	nom d'attribut (valeurs)
5.2	(o)	Attribut:	nom d'attribut (valeurs)
6	(m)	Attribut:	nom d'attribut (valeurs)
6.1	(s)	Attribut:	nom d'attribut (valeurs)

6.2 (s) Attribut: nom d'attribut (valeurs)

SERVICES:

- 1 (o) OpsService: nom du service
- 2. (c) Contrainte: expression de la contrainte
- 2.1 (o) OpsService: nom du service
- 3 (m) MgtService: nom du service

- (1) La rubrique "FAL ASE:" est le nom de l'élément ASE de la couche FAL (FAL ASE) qui fournit les services pour la classe spécifiée.
- (2) La rubrique "CLASS:" est le nom de la classe spécifiée. Tous les objets définis à l'aide de ce modèle seront une instance de cette classe. La classe peut être spécifiée par la présente norme ou par un utilisateur de la présente norme.
- (3) La rubrique "CLASS ID:" est un numéro qui identifie la classe spécifiée. Ce numéro est unique au sein du FAL ASE qui fournira les services pour cette classe. Lorsqu'il est qualifié par l'identité de son FAL ASE, il identifie sans ambiguïté la classe relevant du domaine d'application de la FAL. La valeur "NULL" indique que la classe ne peut pas être instanciée. Les Class ID (identificateurs de classe) entre 1 et 255 sont réservés par la présente norme pour identifier des classes normalisées. Ils ont été attribués pour conserver la compatibilité avec des normes nationales existantes. Les CLASS ID entre 256 et 2048 sont alloués pour identifier les classes définies par l'utilisateur.
- (4) La rubrique "PARENT CLASS:" est le nom de la classe parent pour la classe spécifiée. Tous les attributs définis pour la classe parent et hérités par celle-ci sont hérités pour la classe définie, et ils n'ont donc pas à être redéfinis dans le modèle pour cette classe.

NOTE La classe parent "TOP" indique que la classe définie est une définition de classe initiale. La classe parent "TOP" est utilisée comme point de départ à partir duquel toutes les autres classes sont définies. L'usage de "TOP" est réservé pour les classes définies par la présente norme.

- (5) L'étiquette "ATTRIBUTES" (Attributs) indique que les entrées suivantes sont des attributs définis pour la classe.
 - a) Chacune des entrées d'attribut contient un numéro de ligne dans la colonne 1, un indicateur obligatoire (m) / facultatif (o) / conditionnel (c) / sélecteur (s) dans la colonne 2, une étiquette de type d'attribut dans la colonne 3, un nom ou une expression conditionnelle dans la colonne 4, et, facultativement, une liste de valeurs énumérées dans la colonne 5. Dans la colonne suivant la liste de valeurs, la valeur par défaut pour l'attribut peut être spécifiée.
 - b) Les objets sont normalement identifiés par un identificateur numérique et/ou par un nom d'objet. Dans les modèles de classe, ces attributs clés sont définis sous l'attribut clé.
 - c) Le numéro de ligne définit la séquence et le niveau d'imbrication de la ligne. Chaque niveau d'imbrication est identifié par période. L'imbrication est utilisée pour spécifier
 - i) des champs d'un attribut structuré (4.1, 4.2, 4.3),
 - ii) des attributs conditionnés à un énoncé de contrainte (5). Les attributs peuvent être obligatoires (5.1) ou facultatifs (5.2) si la contrainte est vraie. Tous les attributs facultatifs n'exigent pas des énoncés de contraintes comme le fait l'attribut défini en (5.2).
 - iii) les champs sélection d'un attribut de type choix (6.1 et 6.2).
- (6) L'étiquette "SERVICES" indique que les entrées suivantes sont des services définis pour la classe.
 - a) Un (m) dans la colonne 2 indique que le service est obligatoire pour la classe, alors qu'un (o) indique qu'il est facultatif. Un (c) dans cette colonne indique que le service est conditionnel. Lorsque tous les services définis pour une classe le sont comme

étant facultatifs, l'un au moins doit être sélectionné quand une instance de la classe est définie.

- b) L'étiquette "OpsService" désigne un service opérationnel (1).
- c) L'étiquette "MgtService" désigne un service de gestion (2).
- d) Le numéro de ligne définit la séquence et le niveau d'imbrication de la ligne. Chaque niveau d'imbrication est identifié par période. L'imbrication dans la liste de services sert à spécifier des services conditionnés à un énoncé de contrainte.

3.5.3.3 Conventions pour les définitions des services

3.5.3.3.1 Généralités

La présente norme utilise les conventions descriptives données dans l'ISO/CEI 10731.

Le modèle de service, les primitives de service et les diagrammes de séquence temporelle utilisés sont des descriptions totalement abstraites; ils ne constituent pas une spécification pour une mise en œuvre.

3.5.3.3.2 Paramètres de service

Les primitives de service sont utilisées pour représenter les interactions entre utilisateur de service et fournisseur de service (ISO/CEI 10731). Elles acheminent des paramètres qui indiquent des informations disponibles dans l'interaction entre utilisateur et fournisseur.

NOTE 1 Voir la NOTE en 3.5.3.3.3 relative à la non-inclusion des paramètres de service qui sont appropriés à la spécification de protocole ou à la spécification d'interface de programmation ou à la spécification de mise en œuvre, mais pas à une définition de service abstrait.

La présente norme utilise un format de tableau pour décrire les paramètres de composants des primitives de service. Les paramètres qui s'appliquent à chaque groupe de primitives de service sont consignés en tableaux dans toute la suite de la présente norme. Chaque tableau comporte jusqu'à six colonnes: une colonne pour le nom du paramètre de service et une colonne chacune pour les primitives et les sens de transfert de paramètres utilisés par le service. Les six colonnes possibles sont

- 1) le nom de paramètre;
- 2) les paramètres d'entrée de la primitive "request";
- 3) les paramètres de sortie de la primitive "request";

NOTE 2 Il s'agit d'une fonctionnalité rarement utilisée. Sauf spécification contraire, les paramètres de primitive "request" sont des paramètres d'entrée.

- 4) les paramètres de sortie de la primitive "indication";
- 5) les paramètres d'entrée de la primitive "response"; et
- 6) les paramètres de sortie de la primitive "confirm".

NOTE 3 Les primitives "request", "indication", "response" et "confirm" sont aussi respectivement appelées primitives "requestor.submit", "acceptor.deliver", "acceptor.submit", et "requestor.deliver" (voir ISO/CEI 10731).

Un paramètre (ou un composant de celui-ci) est énuméré dans chaque rangée de chaque tableau. Dans les colonnes appropriées de la primitive de service, un code est utilisé pour spécifier le type d'usage du paramètre sur la primitive spécifiée dans la colonne:

- M Le paramètre est obligatoire pour la primitive;
- U Le paramètre est une option de l'utilisateur et peut ou peut ne pas être fourni, cela dépendant de l'usage dynamique de l'utilisateur du service. Lorsqu'il n'est pas fourni, une valeur par défaut est supposée pour le paramètre;
- C Le paramètre est conditionné à d'autres paramètres ou à l'environnement de l'utilisateur du service;
- (blanc/vidé) Le paramètre n'est jamais présent ;

S Le paramètre est un élément sélectionné.

Certaines entrées sont en plus qualifiées par des éléments désignés entre parenthèses. Ceux-ci peuvent être

- a) une contrainte spécifique au paramètre:
"(") indique que le paramètre équivaut du point de vue de la sémantique au paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau;
- b) une indication qu'une certaine note s'applique à l'entrée;
"(n)" indique que la note "n" suivante contient des informations complémentaires relatives au paramètre et à son utilisation.

3.5.3.3.3 Procédures de service

Les procédures sont définies en termes des

- interactions entre entités d'application via l'échange d'unités de données de protocole d'application de réseau de terrain, et
- interactions entre un fournisseur de services de couche application et un utilisateur de service de couche application dans le même système via l'invocation de primitives de service de couche application.

Ces procédures sont applicables à des instances de communication entre systèmes qui prennent en charge des services de communications à contrainte temporelle au sein de la couche application de réseau de terrain.

NOTE La sous-série de normes CEI 61158-5 définit les jeux de services abstraits. Il ne s'agit ni de spécifications de protocole ni de spécifications de mise en œuvre ni de spécifications d'interface de programmation concrète. Par conséquent, il s'agit de restrictions concernant la mesure dans laquelle les procédures de service peuvent être mandatées dans les parties de la sous-série CEI 61158-5. Les aspects de protocole qui peuvent varier parmi différentes spécifications de protocole ou différentes mises en œuvre quiinstancient les mêmes services abstraits ne sont pas appropriés à être inclus dans ces définitions de service, excepté au niveau d'abstraction qui est nécessairement commun à toutes ces expressions. Dans le présent document, l'Article 7 est l'équivalent de la sous-série CEI 61158-5.

Par exemple, le moyen par lequel des fournisseurs de service appariant des unités de données protocolaires (PDU) de demande et de réponse est approprié pour la spécification dans la norme de spécification de protocole de la sous-série CEI 61158-6 mais pas dans une norme de définition de services abstraits de la série CEI 61158-5. De même, les méthodes de mise en œuvre locales par lesquelles un fournisseur de service ou utilisateur de service apparie des primitives de demande et de confirmation, ou des primitives d'indication et de réponse, sont appropriées pour une spécification de mise en œuvre ou pour une spécification d'interface de programmation, mais pas pour une norme de service abstrait ou pour une norme de protocole, excepté à un niveau d'abstraction qui est nécessairement commun à tous les modes de réalisation de la norme de spécification. Dans tous les cas, la définition abstraite n'est pas autorisée à spécifier outre mesure la réalisation d'instanciation plus concrète.

Des informations supplémentaires sur les procédures de service conceptuelles d'une mise en œuvre d'un protocole qui réalise les services de l'une des définitions de service abstrait de la sous-série CEI 61158-5 peuvent être consultées dans le rapport CEI/TR 61158-1:2007, 9.6.

3.5.4 Conventions pour le protocole AL

3.5.4.1 Concept général

La couche FAL est définie comme un jeu d'éléments ASE orientés objet. Chaque ASE est spécifié dans un paragraphe distinct. Chaque spécification d'ASE est constituée de trois parties: ses définitions de classe, ses services et sa spécification de protocole. Les deux premières parties sont contenues dans la CEI 61158-5. La spécification de protocole pour chacun des ASE est définie dans la présente norme.

Les définitions de classe définissent les attributs pris en charge par chaque ASE. Les attributs sont accessibles à partir d'instances de la classe en utilisant les services d'ASE de gestion spécifiés dans la CEI 61158-5. La spécification de services définit les services qui sont fournis par l'ASE.

La présente norme utilise les conventions descriptives données dans l'ISO/CEI 10731.

3.5.4.2 Conventions pour les définitions de classes

Les définitions de mise en correspondance de la couche liaison de données sont décrites à l'aide de modèles. Chaque modèle est constitué d'une liste d'attributs de la classe. La forme générale du modèle est définie dans la CEI 61158-5-20:2010, 3.8.2.

3.5.4.3 Conventions utilisées dans les diagrammes d'états

Les diagrammes d'états sont décrits au Tableau 1.

Tableau 1 – Conventions utilisées pour les diagrammes d'états

#	État courant	Événement / condition => action	Prochain état
Nom de cette transition	L'état courant auquel s'applique cette transition d'états	Événements ou conditions qui déclenchent cette transition d'état => Les actions qui sont entreprises lorsque les événements ou conditions ci-dessus sont rempli(e)s. Les actions sont toujours indentées sous les événements ou conditions	Le prochain état après que les actions dans cette transition ont été entreprises

Les conventions utilisées dans les diagrammes d'états sont les suivantes:

:= La valeur d'un élément à gauche est remplacée par la valeur d'un élément à droite. Si un élément à droite est un paramètre, il provient de la primitive montrée comme un événement d'entrée.

xxx Un nom de paramètre.

Exemple:

Identif:= reason

signifie que la valeur d'un paramètre 'reason' est attribuée à un paramètre appelé 'Identif.'

"xxx" indique une valeur fixe.

Exemple:

Identif:= "abc"

signifie que la valeur "abc" est attribuée à un paramètre nommé 'Identif.'

= Une condition logique pour indiquer qu'un élément à gauche est égal à un élément à droite.

< Une condition logique pour indiquer qu'un élément à gauche est inférieur à un élément à droite.

> Une condition logique pour indiquer qu'un élément à gauche est supérieur à un élément à droite.

<> Une condition logique pour indiquer qu'un élément à gauche n'est pas égal à un élément à droite.

&& "AND" (ET) logique

|| "OR" (OU) logique

Cette construction permet l'exécution d'une séquence d'actions dans une boucle au sein d'une même transition. La boucle est exécutée pour toutes les valeurs de start_value à end_value.

Exemple:
 for (Identifieur:= start_value to end_value)
 actions
endfor

Cette construction permet l'exécution d'actions alternatives dépendant d'une certaine condition (qui pourrait être la valeur d'un certain identificateur ou le résultat d'une action précédente) au sein d'une même transition.

Exemple:
 If (condition)
 actions
else
 actions
endif

Il est fortement recommandé aux lecteurs de se référer aux paragraphes concernant les définitions des attributs des points d'extrémité de relation d'applications (AREP), aux fonctions locales et aux définitions des FAL-PDU pour appréhender les machines protocolaires. Les lecteurs sont censés avoir une connaissance suffisante de ces définitions et celles-ci sont utilisées sans autres explications.

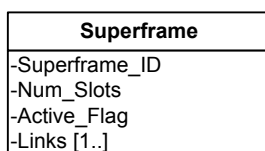
3.5.4.4 Diagramme de classes

3.5.4.4.1 Généralités

Le langage UML est utilisé dans le présent document pour montrer le diagramme de classes. Le but d'un diagramme de classes est de montrer les objets et leurs relations dans un système. Dans un diagramme de classes, les objets sont des classes (par opposition à un diagramme d'instance où les objets sont des instances de classes). Il existe plusieurs formes de relations, notamment les associations, généralisation/spécialisation, agrégation et dépendance.

3.5.4.4.2 Classes

Les classes contiennent trois compartiments disposés verticalement. Le compartiment supérieur est le nom de la classe. Le compartiment médian énumère les attributs de la classe. Le compartiment inférieur énumère les opérations de la classe. Pour tracer un élément classe sur un diagramme de classes, le compartiment supérieur est obligatoire et les deux compartiments inférieurs sont facultatifs. Le compartiment inférieur n'est pas indispensable dans un diagramme dont le but est seulement de montrer la relation entre des classes. Il n'est pas nécessaire de montrer tous les attributs dans tous les dessins où la classe est utilisée. La Figure 1 montre une classe Superframe avec des opérations supprimées. Dans cette figure, le nom de la classe est 'Superframe'. La classe est illustrée avec trois attributs, Superframe_ID, NumSlots et Active.



Légende

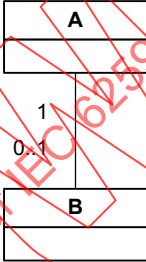
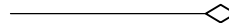
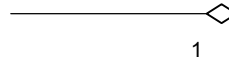
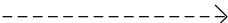
Anglais	Français
Superframe	Superframe

Figure 1 – Exemple de classe Superframe

3.5.4.4.3 Relations

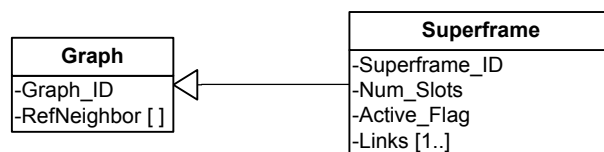
Les formes des relations utilisées dans le présent document sont récapitulées au Tableau 2.

Tableau 2 – Notation des relations entre classes

Relation	Fonction	Notation
Association bidirectionnelle	Une association bidirectionnelle est indiquée par une ligne en trait continu entre les deux classes.	
Association unidirectionnelle	Dans une association unidirectionnelle, deux classes sont reliées, mais une seule classe sait que la relation existe. Cela est indiqué par une ligne fléchée entre les deux classes.	
Association indiquant la multiplicité	La multiplicité spécifie le nombre d'instances d'une classe qui peuvent être reliées à une instance d'une autre classe. Dans cet exemple, 0..1 indique qu'une instance d'un 'A', si elle existe, peut avoir zéro ou une instance d'un 'B' qui lui est associée (B peut ne pas avoir été encore créé).	
Généralisation	Une relation plus spécifique et une description plus générale, utilisées pour l'héritage	
Agrégation de base	Relation partie-tout: le cycle de vie d'une classe partie est indépendant du cycle de vie de la classe entière.	
Agrégation composée	Relation partie-tout: le cycle de vie d'une classe partie est dépendant de l'objet contenant.	
Dépendance	Montre une relation entre deux éléments de modèle; elle indique qu'un objet dépend d'un autre objet pour un certain comportement.	

3.5.4.4.4 Héritage

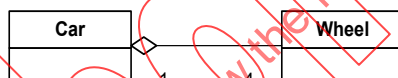
Un important concept en conception orientée objet est l'héritage. L'héritage se réfère à la capacité d'une classe (classe enfant) à hériter de la fonctionnalité identique d'une autre classe (superclasse) et ensuite à ajouter sa propre nouvelle fonctionnalité. Pour modéliser une relation d'héritage sur un diagramme de classes, une ligne en trait continu est tirée de la classe enfant (la classe héritant du comportement) avec une pointe de flèche fermée (ou triangle) pointant vers la superclasse. La Figure 2 montre une classe Superframe modélisée comme un type spécialisé de graphe (Graph).

**Légende**

Anglais	Français
Graph	Graphe
Superframe	Supertrame

Figure 2 – Exemple de relation d'héritage**3.5.4.4.5 Agrégation de base**

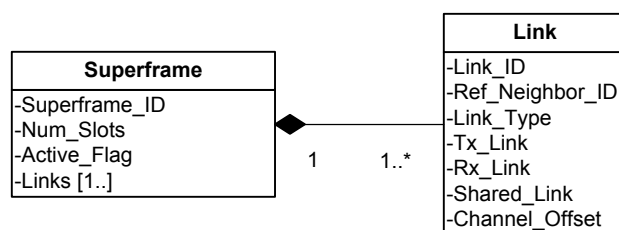
Un autre concept important en conception orientée objet est l'agrégation. L'agrégation sert à modéliser une relation "tout à ses parties". Il existe deux formes d'agrégation: l'agrégation de base et l'agrégation de composition. Dans les relations d'agrégation de base, le cycle de vie d'une classe partie est indépendant du cycle de vie de la classe entière. Comme montré à la Figure 3, Voiture (Car) est une entité entière et Roue (Wheel) est une partie de la Voiture globale. La Roue peut être créée à l'avance et elle peut rester dans un entrepôt avant d'être placée sur une voiture au cours de l'assemblage.

**Légende**

Anglais	Français
Car	Voiture
Wheel	Roue

Figure 3 – Exemple d'agrégation de base**3.5.4.4.6 Agrégation de composition**

L'autre forme d'agrégation est l'agrégation de composition, habituellement appelée tout simplement "composition". La Figure 4 montre la relation entre une supertrame et ses liaisons. Dans ce cas, une liaison ne peut pas exister sans la supertrame. La composition est tracée avec une forme de losange rempli.



Légende

Anglais	Français
Superframe	Supertrrame
Link	Liaison

Figure 4 – Exemple d'agrégation de composition

3.5.4.4.7 Multiplicité

La multiplicité montre le nombre d'instances d'une classe qui peuvent être reliées à une instance d'une autre classe. Le Tableau 3 énumère les exemples d'indicateurs de multiplicité avec leurs significations.

Tableau 3 – Notation de la multiplicité

Valeurs de multiplicité potentielles	
Indicateur	Signification
0..1	Zéro ou un
1	Un seul
0..*	Zéro ou plus
*	Zéro ou plus
1..*	Un ou plus
1..2	Un ou deux
4	Seulement quatre
0..5	Zéro à cinq

4 Spécification de couche physique et définition de services

4.1 Définition de services de couche physique

4.1.1 Vue d'ensemble

Le Type 20 utilise les fréquences radioélectriques (RF) pour communiquer entre dispositifs. La distance entre deux dispositifs peut être de 100 mètres ou plus, cela dépendant de la puissance de l'émetteur, du type d'antenne et de l'absence d'obstacle entre les deux dispositifs. Les dispositifs conformes au Type 20 peuvent être alimentés en énergie par le secteur, une batterie d'accumulateurs, des dispositifs de récupération d'énergie (par exemple solaires, vibrations, etc.) ou une combinaison de ceux-ci.

La présente norme définit la couche physique (PhL) de Type 20. Elle est utilisée pour les dispositifs mettant en œuvre la communication de réseaux maillés sans fil telle que définie par la présente norme. La PhL est responsable de la méthode de signalisation, de l'intensité du signal, de la sensibilité des dispositifs, et de l'environnement pour envoyer des bits à travers les supports réseau. Cette spécification de PhL sans fil repose sur la couche physique

DSSS de 2 450 MHz et l'utilisation de la modulation par déplacement de phase quadrivalente décalée (O-QPSK) selon l'IEEE STD 802.15.4-2006. La présente norme définit les services que la couche physique fournit aux couches supérieures de la présente norme pour envoyer et recevoir des données sur les supports. Elle définit également les paramètres de gestion locaux utilisés pour commander le fonctionnement de la PhL.

La PhL de la présente norme traite de caractéristiques radio constituant une connexion réseau. Elle s'associe à toute partie de la structure réseau qui ne traite d'informations d'aucune manière. Elle est chargée d'envoyer les bits à travers les supports réseau. Elle ne définit pas la nature d'un bit ou la manière de l'utiliser, mais simplement la manière dont il est envoyé. La PhL est chargée d'émettre et de recevoir les données.

4.1.2 Services de couche physique

4.1.2.1 Moyens des services de la couche physique

La PhL fournit les services suivants:

Service de commande – elle prend en charge l'activation ou la désactivation de l'entité de PhL (PhE) et l'accomplissement d'une évaluation de voie libre du support sous-jacent,

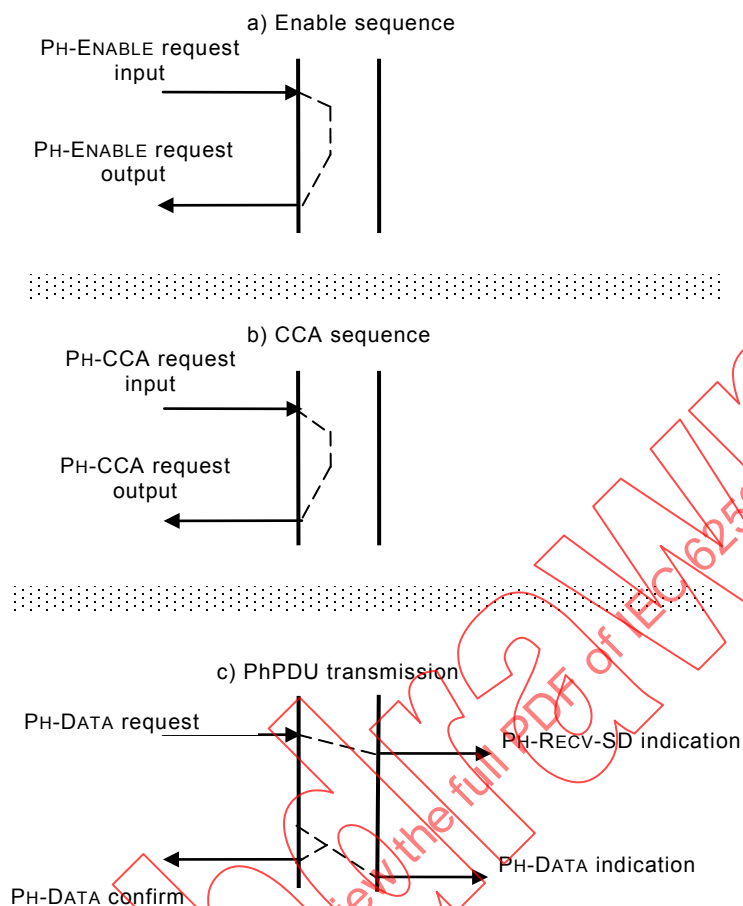
Service message – elle prend en charge le transfert de données utilisateur du service de couche physique (PhS) depuis un utilisateur source du PhS jusqu'à un utilisateur destination du PhS, et

Services de gestion de PhL – elle prend en charge la configuration du PhL.

4.1.2.2 Séquence de primitives

Les relations que les primitives de chaque type à l'interface d'utilisateur du PhS et d'une PhE ont avec des primitives à l'interface d'utilisateur du PhSet des autres PhE pour des cas exempts d'erreurs sont récapitulées dans les diagrammes de la Figure 5.

Avant d'utiliser la PhE, l'utilisateur du PhS doit l'activer comme montré dans la séquence a). La séquence c) montre la transmission de PhPDU de l'entité PhL (PhE) source vers la PhE de destination. La transmission commence par une évaluation de voie libre comme montré dans la séquence b). À la destination, la réception commence par une indication PH-RECV-SD après que le délimiteur de début a été reçu par la PhE. La réception sans erreurs se termine avec une indication "PH-DATA indication". S'il y a une erreur de réception, la PhE à la destination l'indique par un paramètre de statut et ne délivre pas de données d'utilisateur de services de couche physique (PhS).



Légende

Anglais	Français
Ph-Data confirm	Confirmation de données de couche physique
Ph-Data request	Demande de données de couche physique
Ph-Data indication	Indication de données de couche physique
c) PhPDU transmission	c) Émission de PhPDU
Ph-Enable request output	Sortie de demande d'activation de couche physique t
Ph-Enable request input	Entrée de demande d'activation de couche physique t
a) Enable sequence	a) Séquence d'activation
Ph-CCA request output	Sortie de demande d'évaluation de voie libre de couche physique
Ph-CCA request input	Entrée de demande d'évaluation de voie libre de couche physique
b) CCA sequence	b) Séquence d'évaluation de voie libre (CCA)
Ph-Recv-SD indication	Ph-Recv-SD indication
DL-Transmit confirm	DL-Transmit confirm
DL-Transmit request	DL-Transmit request
DL-Transmit indication	DL-Transmit indication

Figure 5 – Séquences de services de couche physique

4.1.2.3 Service PH-ENABLE

4.1.2.3.1 Types de primitives et paramètres

Le Tableau 4 indique les types de primitives et les paramètres nécessaires au service PhL-Enable. Ce service est utilisé pour activer ou pour désactiver la PhE et la placer dans le mode émission ou réception sur la voie de fréquence spécifiée. Le service "request" (c'est-à-dire demande) n'est couronné de succès que si la voie est dans l'état de repos.

Tableau 4 – Primitives et paramètres de PH-ENABLE

Nom de paramètre	Demande	
	Entrée	Sortie
Channel	M	M
State	M	M

4.1.2.3.2 Paramètres

4.1.2.3.2.1 Channel (voie)

Il spécifie l'une des voies disponibles pour lesquelles ce service est utilisé.

4.1.2.3.2.2 State (état)

Dans la primitive "request" (demande), il spécifie l'état dans lequel il convient que la voie se trouve, lorsque ce service est parachevé. Dans la primitive "confirm", il spécifie l'état actuel de la voie après que le service est parachevé. Sa valeur peut être soit Enable, soit Disable.

4.1.2.4 Service PH-CCA

4.1.2.4.1 Types de primitives et paramètres

Le Tableau 5 indique les types de primitives et les paramètres nécessaires au service PH-CCA. Ce service est utilisé pour vérifier l'état de la voie. Le service "request" n'est couronné de succès que si la voie est activée dans le mode réception.

Tableau 5 – Primitives et paramètres de PH-CCA

Nom de paramètre	Requête	
	Entrée	Sortie
State		M
NOTE La primitive "request" ne nécessite aucun paramètre.		

4.1.2.4.2 Paramètre - state

Ce paramètre montre le résultat de l'évaluation de voie libre. Les valeurs de ce paramètre sont:

- Channel Idle (voie au repos),
- Channel busy (voie occupée),
- Transceiver off and unable to perform assessment (c'est-à-dire émetteur-récepteur éteint et incapable d'accomplir l'évaluation).

4.1.2.5 Service PH-RECV-SD

La seule primitive de ce service est "indication" et elle n'a pas de paramètre. Cette primitive est utilisée par la PhE à la fin de la réception du délimiteur de début dans le mode réception.

4.1.2.6 Service PH-DATA

4.1.2.6.1 Types de primitives et paramètres

Le Tableau 6 indique les types de primitives et les paramètres nécessaires au service PH-DATA.

Tableau 6 – Primitives et paramètres de PH-DATA

Nom de paramètre	Requête	Indication	Confirmation
PhS-user-data	M	C(=)	
Receive signal level		C	
Status		M	M

4.1.2.6.2 Paramètres

4.1.2.6.2.1 PhS-user-data

Ce paramètre permet l'émission de données entre les utilisateurs du PhS sans altération par le fournisseur du PhS. S'il y a une erreur de réception, ce paramètre est absent de la primitive "indication". Autrement, sa valeur est identique à la valeur dans la primitive "request".

4.1.2.6.2.2 Niveau de signal reçu

Ce paramètre indique le niveau estimé du signal de réception pour cette PhPDU. Il n'est présent qu'en l'absence d'erreur de réception.

4.1.2.6.2.3 Statut

Dans la primitive "indication", ce paramètre indique soit le succès, soit l'une des causes d'erreur suivantes:

- Dépassement de capacité du tampon de réception,
- Paquet incomplet.

Dans la primitive "Confirm", ce paramètre indique soit le succès, soit l'une des causes d'erreur suivantes:

- Émetteur-récepteur éteint,
 - Récepteur allumé,
- NOTE Si le récepteur est allumé, la PhE ne peut pas émettre.
- Émetteur occupé.

4.1.2.7 Service PHM-SET

4.1.2.7.1 Types de primitives et paramètres

Le Tableau 7 indique la primitive et les paramètres de ce service.

Tableau 7 – Primitive et paramètres de PHM-SET

Nom de paramètre	Requête	
	Entrée	Sortie
PHM-object-identifiant	M	
Valeur désirée	M	
Statuts		M

4.1.2.7.2 Paramètres

4.1.2.7.2.1 PHM-object-identifiant

Ce paramètre spécifie l'objet au sein de la PhE dont la valeur est à modifier. Le domaine de désignation du PHM-object-identifiant est la vue locale.

4.1.2.7.2.2 Valeur désirée

Ce paramètre spécifie la valeur désirée pour l'objet spécifié par le paramètre "PHM-object-identifiant" associé. Son type est identique à celui de l'objet spécifié.

4.1.2.7.2.3 Statuts

Ce paramètre permet à l'utilisateur du PhS de déterminer si le service demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

4.1.2.8 Service PHM-GET

4.1.2.8.1 Types de primitives et paramètres

Le Tableau 8 indique la primitive et les paramètres de ce service.

Tableau 8 – Primitive et paramètres de PHM-GET

Nom de paramètre	Requête	
	Entrée	Sortie
PHM-object-identifiant	M	
Statuts		M
valeur courante		C

4.1.2.8.2 Paramètres

4.1.2.8.2.1 PHM-object-identifiant

Ce paramètre spécifie l'objet au sein de la PhE dont la valeur est demandée. Le domaine de désignation du PHM-object-identifiant est la vue locale.

4.1.2.8.2.2 Statuts

Ce paramètre permet à l'utilisateur du PhS de déterminer si le service demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

4.1.2.8.2.3 Valeur courante

Ce paramètre est présent lorsque le paramètre de statut indique que le service demandé a été accompli avec succès. Ce paramètre spécifie la valeur actuelle pour l'objet spécifié par le paramètre "PHM-object-identifier" associé. Son type est identique à celui de l'objet spécifié.

4.2 Spécification de protocole de couche physique

Le protocole de couche physique est spécifié dans l'IEEE 802.15.4-2006. Le profil pour la présente norme est spécifié en 9.3.3.

5 Définition des services de couche liaison de données – Éléments de Type 20

5.1 Services de la couche liaison de données

5.1.1 Moyens des services de la couche liaison de données

5.1.1.1 Service transfert de données

Les services Message prennent en charge le transfert de DL-service-data-unit (DLSDU) d'un utilisateur de DLS source vers un utilisateur de DLS de destination. La couche liaison de données prend en charge la réémission automatique afin d'assurer un transfert de données sans erreurs.

5.1.1.2 Service réception de données

Ce service prend en charge la réception seule d'une DL-service-data-unit (DLSDU) sans envoyer d'acquittement.

5.1.1.3 Service événement

Ces services indiquent les événements sur la liaison:

- réception d'une DLPDU issue d'une DLE indiquant qu'elle se déconnecte de la liaison et fourniture d'une indication de cet événement à l'utilisateur de DLS,
- détection de la défaillance du trajet vers une autre DLE et fourniture d'une indication de cet événement à l'utilisateur du DLS.
- réception d'une DLPDU qui invite une nouvelle DLE à se connecter à la liaison et fourniture d'une indication de cet événement à l'utilisateur de DLS, et
- réception d'une DLPDU issue d'une DLE absente du tableau des voisins de la DLE de réception et fourniture d'une indication de cet événement à l'utilisateur de DLS.

5.1.1.4 Service de gestion locale de DL

Ce service prend en charge la configuration de la DL et l'accès aux statistiques de la DLL. Les moyens de gestion de la DL fournissent des moyens pour

- écrire des paramètres de configuration de DLE;
- lire des paramètres de configuration, des paramètres opérationnels et des statistiques de DLE;
- commander des actions majeures de la DLE; et
- recevoir une notification des événements significatifs de DLE.

Ces moyens constituent ensemble le service de gestion de DL (DLMS).

5.1.1.5 Fonction de sécurité

La DL assure une fonction de sécurité pour les communications entre les paires de dispositifs communiquant sur une liaison. Un code est utilisé pour garantir que la DLPDU émane d'un dispositif authentifié et approuvé.

Il n'y a pas de service utilisateur explicite pour la sécurité. Les paramètres de sécurité sont écrits en utilisant le service de gestion de DL.

5.1.2 QoS pour le service messagerie

5.1.2.1 Priorité

Les primitives du service message spécifient la priorité de la DLSDU transférée. Les quatre priorités, de la plus haute à la plus basse, sont définies ci-dessous.

- Commande – il convient que l'utilisateur de DLS se serve de cette priorité pour envoyer une DLSDU d'informations de diagnostic, de configuration ou de commande relatives au réseau.
- Processus de données – il convient que l'utilisateur de DLS se serve de cette priorité pour envoyer une DLSDU contenant des mesures issues d'émetteurs de processus ou des valeurs de consigne à des dispositifs de commande ou n'importe quelles autres données relatives au processus ou données statistiques du réseau.
- Normal – il convient que l'utilisateur de DLS se serve de cette priorité pour envoyer une DLSDU contenant n'importe quelles autres données qui ne remplissent pas les critères relatifs à la priorité "Command", "Process-Data" ou "Alarm".
- Alarme – il convient que l'utilisateur de DLS se serve de cette priorité pour envoyer une DLSDU contenant seulement une alarme ou une DLSDU d'événement.

5.1.2.2 Expiration de délai

Chaque demande de message spécifie les bornes supérieures de la durée maximale permise pour l'achèvement de chaque instance correspondante d'une séquence de primitives de message.

5.1.3 Séquence de primitives

Les relations que les primitives de chaque type au niveau de l'interface d'utilisateur du DLS et d'une DLE ont avec les primitives au niveau de l'utilisateur du DLS et des autres DLE pour des cas exempts d'erreurs sont récapitulées dans les diagrammes de la Figure 6.

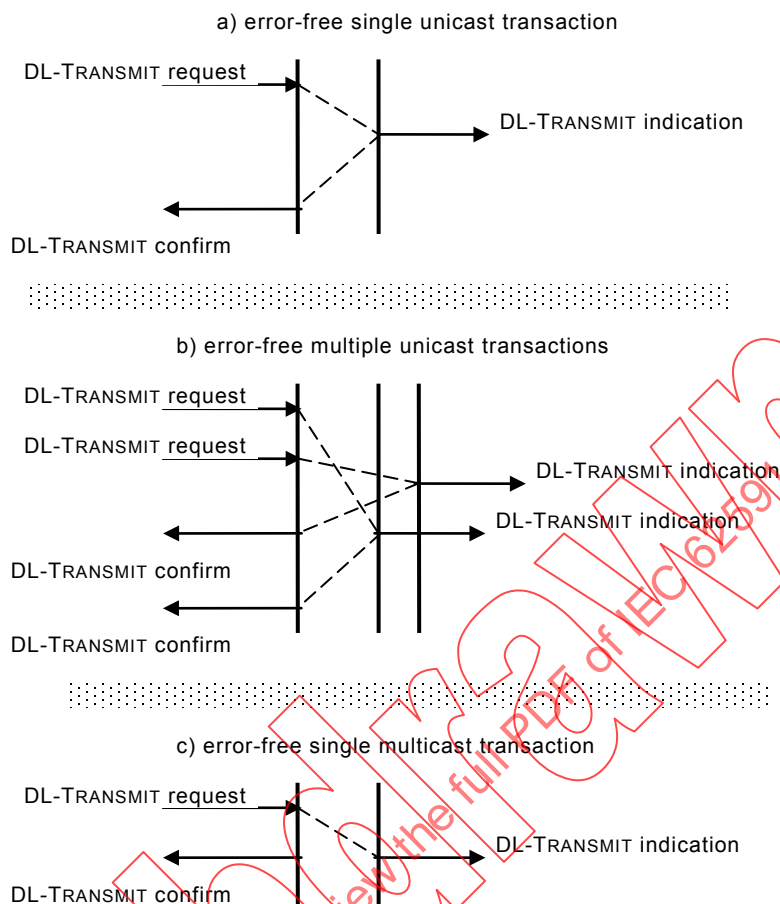
La séquence a) montre une simple transaction de monodiffusion sans erreurs. Dans cette séquence; la demande "DL-TRANSMIT request" au niveau de la DLE source insère le message dans la file d'attente d'émission de la DLE. Quelques temps plus tard, lorsque survient un intervalle de temps prenant en charge la communication programmée à la DLE de destination, la DLPDU transportant la DLSDU de message est émise. Lorsqu'une DLPDU de non-diffusion est reçue et validée, la DLE réceptrice émet un acquittement et génère une indication "DL-TRANSMIT indication" à l'attention de son utilisateur. À la réception de l'acquittement, la DLE source génère une confirmation "DL-TRANSMIT confirm" à l'attention de l'utilisateur demandeur.

La séquence b) montre plusieurs transactions de monodiffusion sans erreurs. Dans cette séquence; la demande "DL-TRANSMIT request" au niveau de la DLE source insère le message dans la file d'attente d'émission de la DLE. Avant que la DLPDU transportant cette DLSDU de message ne soit émise, la DLE reçoit une autre demande "DL-TRANSMIT request" à envoyer à une DLE différente de la demande précédente. Quelques temps plus tard, lorsque survient un intervalle de temps prenant en charge la communication programmée à la DLE de destination, la DLPDU transportant la seconde DLSDU de message est émise. Lorsqu'une DLPDU de non-diffusion est reçue et validée, la DLE réceptrice émet un acquittement et génère une

indication "DL-TRANSMIT indication" à l'attention de son utilisateur. Par conséquent, la DLSDU peut ne pas être émise dans le même ordre dans lequel elle a été émise au niveau de la DLE source.

La séquence c) montre une simple transaction de multidiffusion sans erreurs. Dans cette séquence; la demande "DL-TRANSMIT request" au niveau de la DLE source insère le message dans la file d'attente d'émission de la DLE. Quelques temps plus tard, lorsque survient un intervalle de temps prenant en charge la communication programmée à la DLE de destination, la DLPDU transportant la DLSDU de message est émise. Lorsqu'une DLPDU destinée à plusieurs DLE est reçue et validée, la DLE réceptrice génère une indication "DL-TRANSMIT indication" à l'attention de son utilisateur. À l'émission de la DLPDU, la DLE source génère une confirmation "DL-TRANSMIT confirm" à l'attention de l'utilisateur demandeur.

IECNORM.COM: Click to view the full PDF of IEC 62591:2010



Légende

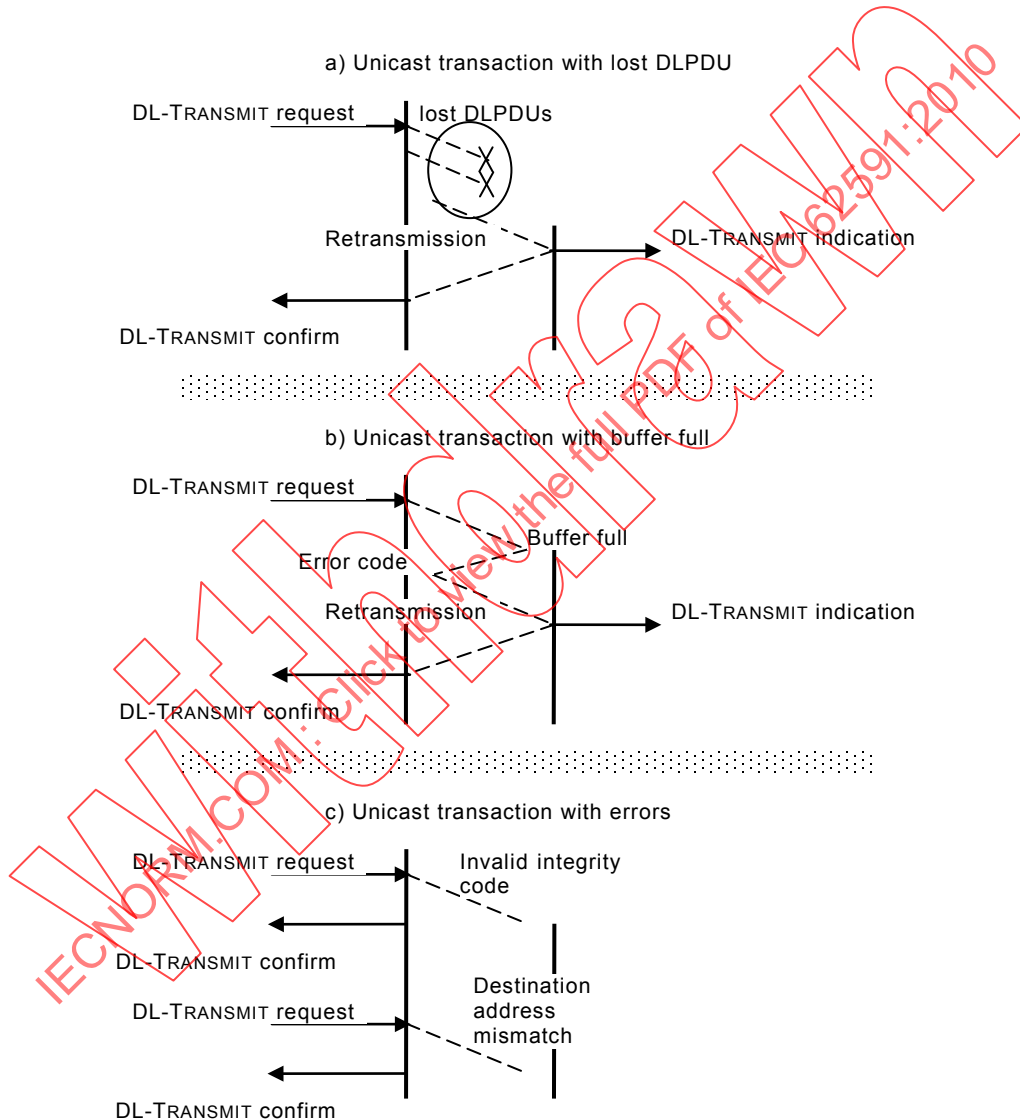
Anglais	Français
a) error-free single unicast transaction	a) simple transaction de monodiffusion sans erreurs
b) error-free multiple unicast transactions	b) multiples transactions de monodiffusion sans erreurs
c) error-free single multicast transaction	c) simple transaction de multidiffusion sans erreurs
DL-Transmit request	DL-Transmit request
DL-Transmit request	DL-Transmit request
DL-Transmit indication	DL-Transmit indication
DL-Transmit indication	DL-Transmit indication
DL-Transmit request	DL-Transmit request
DL-Transmit indication	DL-Transmit indication
DL-Transmit confirm	DL-Transmit confirm
DL-Transmit confirm	DL-Transmit confirm
DL-Transmit confirm	DL-Transmit confirm

Figure 6 – Séquences sans erreurs de services message

Les relations que les primitives de chaque type au niveau de l'interface d'utilisateur du DLS et d'une DLE ont avec des primitives au niveau de l'utilisateur du DLS et des autres DLE pour des cas d'erreurs sont récapitulées dans les diagrammes de la Figure 7.

La séquence a) montre la DLPDU perdue dans la transmission. La séquence b) montre que la DLE réceptrice a rejeté la DLPDU à cause d'un manque de ressources tampon et envoie un acquittement avec un code de réponse "Error". Comme montré, la DLE source réémet le message tant qu'un acquittement avec un code de réponse "Success" n'a pas été reçu ou bien tant qu'une limite sur le nombre minimum de nouvelles tentatives n'a pas été atteinte. Si le paquet séjourne dans la DLE source sans acquittement de la part de la DLE de destination pendant une durée supérieure à la valeur spécifiée pour l'expiration de délai, la DLE cesse ses tentatives.

La séquence c) montre que la DLE de destination DLE a reçu la DLPDU soit avec un code d'intégrité de message non valide, soit avec une adresse de destination qui ne concorde pas. La DLE réceptrice rejette la DLPDU et n'envoie pas d'acquiescement.



Légende

Anglais	Français
a) Unicast transaction with lost DLPDU	a) Transaction de monodiffusion avec perte de DLPDU
lost DLPDUs	DLPDU perdues
Retransmission	Retransmission
b) Unicast transaction with buffer full	b) Transaction de monodiffusion avec tampon plein

Anglais	Français
Buffer full	Tampon plein
Error code	Code d'erreur
Retransmission	Retransmission
c) Unicast transaction with errors	c) Transaction de monodiffusion avec erreurs
Invalid integrity code	Code d'intégrité non valide
Destination address mismatch	Discordance d'adresses de destination
DL-Transmit request	DL-Transmit request
DL-Transmit request	DL-Transmit request
DL-Transmit request	DL-Transmit request
DL-Transmit indication	DL-Transmit indication
DL-Transmit indication	DL-Transmit indication
DL-Transmit request	DL-Transmit request
DL-Transmit confirm	DL-Transmit confirm
DL-Transmit confirm	DL-Transmit confirm

Figure 7 – Séquences de service message avec erreurs

La Figure 8 montre la primitive "receive-only" (c'est-à-dire réception seule). Celle-ci est disponible dans une DLE qui prend en charge un mode de fonctionnement banalisé dans lequel des communications issues d'autres DLE sont captées.

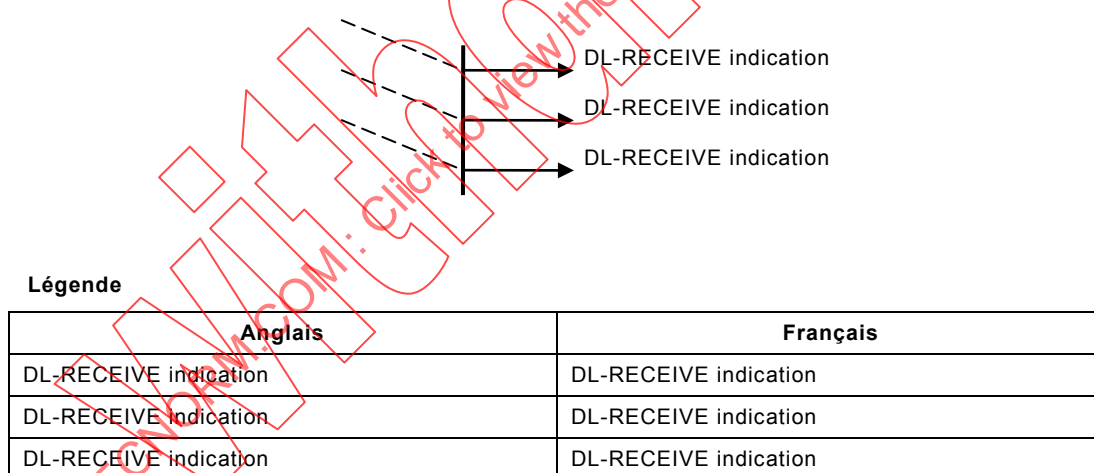


Figure 8 – Service message "réception seule"

5.1.4 Service DL-TRANSMIT

5.1.4.1 Fonction

Il peut être utilisé pour émettre une DLSDU indépendante complète d'une DLE vers une autre DLE dans un seul accès au service et pour retourner le statut de cette livraison à l'utilisateur de DLS émetteur.

5.1.4.2 Types de primitives et paramètres

Le Tableau 9 indique les types de primitives et les paramètres nécessaires au service DL-Transmit.

Tableau 9 – Primitives et paramètres de DL-Transmit

Nom de paramètre	Request	Indication	Confirm
Handle	M		M(=) ^a
Broadcast	M	M	
Source address		M	
Destination address			
Short address	S		
Long address	S		
Superframe_ID	C		
Graph	C		
Priority	M	M(=)	
Timeout	M		
DLS-user-data	M	M(=)	
Status		M	M
a La valeur de ce paramètre est égale à la valeur dans la primitive "request".			

5.1.4.3 Paramètres

5.1.4.3.1 Handle

Ce paramètre est pris en charge pour la commodité de l'utilisateur de DLS. Sa valeur dans la primitive "confirm" est identique à la valeur dans la primitive "request" correspondante.

5.1.4.3.2 Broadcast

Si la valeur de ce paramètre est TRUE, la DLSDU est diffusée dans la supertrame spécifiée dans la primitive "request".

5.1.4.3.3 Source address

Ce paramètre identifie la source de la DLSDU acheminée par la primitive "indication".

5.1.4.3.4 Destination address

5.1.4.3.4.1 Short address

Ce paramètre n'est présent que si Broadcast est FALSE. Il s'agit du surnom de la DLE de destination qui est utilisée pour le prochain saut.

5.1.4.3.4.2 Long address

Ce paramètre n'est présent que si Broadcast est FALSE. Il s'agit de l'ID propre de la DLE destination qui est utilisée pour le prochain saut.

5.1.4.3.5 Superframe_ID

Ce paramètre n'est présent que si Broadcast est TRUE. Il spécifie la supertrame dont les liaisons de diffusion doivent être utilisées pour émettre la DLSDU.

5.1.4.3.6 Graph

Ce paramètre n'est présent que si le routage graphique est à utiliser. Lorsqu'il est utilisé, le graphe indique les voisins qui peuvent servir de destination pour le prochain saut.

5.1.4.3.7 Priority

Ce paramètre spécifie la priorité de l'émission. Sa valeur est un élément quelconque de l'ensemble: {command, process-data, normal, alarm}.

5.1.4.3.8 Timeout

Il s'agit de la durée maximale pour l'achèvement de chaque instance correspondante d'une séquence de primitives de DL-TRANSMIT.

5.1.4.3.9 DLS-user-data

Ce paramètre permet l'émission de données entre les utilisateurs du DLS sans altération par le fournisseur du DLS.

5.1.4.3.10 Status

Dans la primitive "indication", ce paramètre indique la clé de liaison de données utilisée pour authentifier la DLPDU et si, oui ou non, la DLPDU était une diffusion.

Dans la primitive "confirm", ce paramètre indique que le service a été fourni avec succès ou n'a pas abouti pour la raison spécifiée.

5.1.5 Service DL-RECEIVE

5.1.5.1 Fonction

Il peut être utilisé par une DLE pour recevoir n'importe quelle DLPDU et fournir la DLSDU dans la DLPDU en question à l'utilisateur de DLS. Ce service est également appelé "mode de fonctionnement banalisé" et peut être utilisé pour le dépannage réseau.

5.1.5.2 Type de primitive et paramètres

Le Tableau 10 indique le type de primitive et les paramètres nécessaires au service Receive.

Tableau 10 – Primitive et paramètres de DL-RECEIVE

Nom de paramètre	Indication
Packet RSL	M
DLS-user-data	M

5.1.5.3 Paramètres

5.1.5.3.1 Packet RSL

Ce paramètre est le RSL (niveau de signal reçu) de la DLPDU juste reçue.

5.1.5.3.2 DLS-user-data

Ce paramètre achemine la donnée reçue par la DLE.

5.1.6 Service DL-DISCONNECT

5.1.6.1 Fonction

Ce service notifie qu'une autre DLE se déconnecte de la liaison. Une DLPDU a été reçue de cette DLE indiquant qu'elle se déconnecte de liaison.

NOTE Le service Disconnect DLPDU n'est attendu que d'un dispositif qui est un voisin. Il est possible que le dispositif de réception ne connaisse pas tous ses voisins. Par conséquent, il reçoit toujours Disconnect DLPDU et fournit une indication, même s'il provient d'un dispositif qui n'est pas connu comme voisin.

5.1.6.2 Type de primitive et paramètres

5.1.6.2.1 Généralités

Le Tableau 11 indique le type de primitive et les paramètres nécessaires au service Disconnect.

Tableau 11 – Primitive et paramètres de DL-DISCONNECT

Nom de paramètre	Indication
Source address	M

5.1.6.2.2 Source address

Ce paramètre identifie l'adresse de la DLE qui s'est juste déconnectée.

5.1.7 Service DL-PATH-FAILURE

5.1.7.1 Fonction

Ce service notifie que le trajet menant à une DLE avec laquelle cette DLE est connectée a une défaillance. Les communications avec la DLE indiquée ont cessé intempestivement suite à une expiration de délai et la DLE apparaît être indisponible.

5.1.7.2 Type de primitive et paramètres

5.1.7.2.1 Généralités

Le Tableau 12 indique le type de primitive et les paramètres nécessaires au service PATH-FAILURE.

Tableau 12 – Primitive et paramètres de DL-PATH-FAILURE

Nom de paramètre	Indication
Source address	M

5.1.7.2.2 Source address

Ce paramètre identifie l'adresse de la DLE à laquelle le trajet a juste failli.

5.1.8 Service DL-ADVERTISE

5.1.8.1 Fonction

Ce service notifie que la DLE a reçu une DLPDU Advertise. Il convient que l'utilisateur du DLS utilise les informations fournies dans ce service avant de rejoindre le réseau.

5.1.8.2 Type de primitive et paramètres

5.1.8.2.1 Généralités

Le Tableau 13 indique le type de primitive et les paramètres nécessaires au service Advertise.

Tableau 13 – Primitive et paramètres de DL-ADVERTISE

Nom de paramètre	Indication
DLS-user-data	M

5.1.8.2.2 DLS-user-data

Ce paramètre achemine les données reçues par la DLE dans la DLPDU Advertise.

5.1.9 Service DL-NEIGHBOR**5.1.9.1 Fonction**

Ce service notifie que la DLE a reçu une DLPDU d'une DLE qui n'est pas répertoriée dans son tableau de voisins (voir 6.3.3.4).

5.1.9.2 Type de primitive et paramètres

Le Tableau 14 indique le type de primitive et les paramètres nécessaires pour le service voisin.

Tableau 14 – Primitive et paramètres de DL-NEIGHBOR

Nom de paramètre	Indication
Source address	M
Packet RSL	M

5.1.9.3 Paramètres**5.1.9.3.1 Adresse Source**

Ce paramètre identifie l'adresse de la DLE qui n'est pas dans son tableau de voisins.

5.1.9.3.2 Paquet RSL

Cette primitive indique le RSL (niveau de signal reçu) de la DLPDU reçue.

5.1.10 DLM-SET**5.1.10.1 Fonction**

Cette primitive peut être utilisée pour établir (écrire) la valeur d'un paramètre de configuration de DLE.

5.1.10.2 Types de paramètres

Le Tableau 15 indique la primitive et les paramètres du service Set.

Tableau 15 – Primitive et paramètres de DLM-SET

Nom de paramètre	Requête	
	Entrée	Sortie
DLM-object-identifier	M	
Desired-value	M	
Status		M

5.1.10.3 Paramètres

5.1.10.3.1 DLM-object-identifiant

Ce paramètre spécifie la primitive ou l'objet composé au sein de la DLE dont la valeur est à modifier. Le domaine de désignation de l'identificateur d'objet de gestion de DL (DL-management (DLM) object-identifiant) est la vue locale de DLM.

5.1.10.3.2 Desired-value

Ce paramètre spécifie la valeur désirée pour l'objet DLM spécifié par le DLM object-identifiant associé. Son type est identique à celui de l'objet DLM spécifié.

5.1.10.3.3 Status

Ce paramètre permet à l'utilisateur du Service DLM (DLMS) de déterminer si le DLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.1.11 DLM-GET

5.1.11.1 Fonction

Cette primitive peut être utilisée pour récupérer (lire) la valeur d'un paramètre de configuration, d'un paramètre opérationnel ou les statistiques d'une DLE.

5.1.11.2 Types de paramètres

Le Tableau 16 indique la primitive et les paramètres du service GET.

Tableau 16 – Primitive et paramètres de DLM-GET

Nom de paramètre	Requête	
	Entrée	Sortie
DLM-object-identifiant	M	
Status		M
Current-value		C

5.1.11.3 Paramètres

5.1.11.3.1 DLM object-identifiant

Ce paramètre spécifie la primitive ou l'objet composé au sein de la DLE dont la valeur est demandée. Le domaine de désignation du DLM-object-identifiant est la vue locale DLM.

5.1.11.3.2 Status

Ce paramètre permet à l'utilisateur du DLMS de déterminer si le DLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.1.11.3.3 Valeur courante

Ce paramètre est présent lorsque le paramètre de statut indique que le service demandé a été accompli avec succès. Il spécifie la valeur actuelle pour l'objet DLM spécifié par le DLM-object-identifiant associé. Son type est identique à celui de l'objet DLM spécifié.

5.1.12 DLM-ACTION

5.1.12.1 Fonction

Cette primitive peut être utilisée pour commander une action spécifiée de la DLE.

5.1.12.2 Types de paramètres

Le Tableau 17 indique les primitives et les paramètres du service ACTION.

Tableau 17 – Primitive et paramètres de DLM-ACTION

	Requête	Confirmation
Nom de paramètre	Entrée	Sortie
Desired-action	M	
Action-qualifiers	C	
Status		M
Additional-information		C

5.1.12.3 Paramètres

5.1.12.3.1 Desired-action

Ce paramètre spécifie l'action souhaitée que la DLE a à entreprendre. Les valeurs possibles pour ce paramètre sont énumérées au Tableau 47.

5.1.12.3.2 Action-qualifiers

Ce paramètre spécifie des paramètres supplémentaires spécifiques à l'action qui servent à qualifier l'action commandée.

NOTE Certaines des actions exigent des paramètres complémentaires, et ainsi DELETE-SUPERFRAME exige Superframe_ID.

5.1.12.3.3 Status

Ce paramètre permet à l'utilisateur du DLMS de déterminer si le DLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.1.12.3.4 Additional-information

Ce paramètre facultatif fournit des informations complémentaires spécifiques à l'action qui augmentent le statut retourné.

5.1.13 DLM-EVENT

5.1.13.1 Fonction

Cette primitive est utilisée pour rapporter l'occurrence d'un événement DL susceptible d'être significatif pour la gestion de DL.

5.1.13.2 Types de paramètres

Pas de paramètres définis.

5.2 Services de couche réseau

5.2.1 Moyens des services de la couche réseau

5.2.1.1 Service transfert de données

La couche réseau fournit un service pour émettre une unité de donnée de service de couche réseau (NL) (une NSDU) d'une entité NL (NLE) source vers une NLE de destination. Cela peut nécessiter d'acheminer une unité de donnée de protocole de NL (une NPDU) en passant par des NLE intermédiaires. Le chemin pour une NPDU est établi à l'aide de services de gestion de NL (NLMS).

5.2.1.2 Service de gestion locale de NL

Ce service prend en charge la configuration de la NL et l'accès aux statistiques de la NL. Les moyens de gestion de la NL fournissent des moyens pour

- écrire des paramètres de configuration de NLE;
- lire des paramètres de configuration, des paramètres opérationnels et des statistiques de NLE;
- commander des actions majeures de la NLE; et
- recevoir une notification des événements significatifs de NLE.

Ces moyens constituent ensemble le NLMS.

5.2.1.3 Fonction de sécurité

La NL assure une fonction de sécurité pour les communications de bout en bout entre paires de dispositifs en communication. La NL est orientée session pour les besoins de la sécurité. Une session permet une communication privée et sécurisée entre une paire d'adresses réseau. Les types de clés de sécurité ci-après sont pris en charge.

- Join – (Rattachement) cette clé est utilisée lorsque le dispositif tente de rejoindre le réseau.
- Network manager – (Gestionnaire de réseau) cette clé sert à la communication entre le gestionnaire de réseau et un dispositif de réseau.
- Gateway – (Passerelle) cette clé sert à la communication entre la passerelle et un dispositif de réseau.

Il n'y a pas de service utilisateur explicite pour la sécurité. Les paramètres de sécurité sont écrits en utilisant le service de gestion de NL.

5.2.2 QoS pour le service messagerie

Les primitives du service transfert de données spécifient la priorité de la NSDU transférée. Les quatre priorités sont, de la plus haute à la plus basse:

Commande – il convient que l'utilisateur de NLS se serve de cette priorité pour envoyer des données d'informations de diagnostic, de configuration ou de commande relatives au réseau.

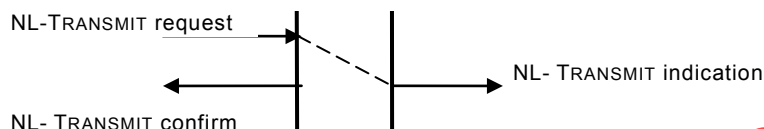
Processus de données – il convient que l'utilisateur de NLS se serve de cette priorité pour envoyer des données contenant des mesures issus d'émetteurs de processus ou des valeurs de consigne à des dispositifs de commande ou n'importe quelles autres données relatives au processus ou données statistiques du réseau.

Normal – il convient que l'utilisateur de NLS se serve de cette priorité pour envoyer des données contenant n'importe quelles autres données qui ne remplissent pas les critères relatifs à la priorité "Command", "Process-Data" ou "Alarm".

Alarme – il convient que l'utilisateur de NLS se serve de cette priorité pour envoyer des données contenant seulement une alarme ou des informations d'événement.

5.2.3 Séquence de primitives

Les relations que les primitives aux points d'extrémité du service message de chaque type au niveau de l'interface de l'utilisateur du NLS et d'une NLE ont avec les primitives au niveau de l'utilisateur de NLS et des autres NLE sont récapitulées dans les diagrammes de la Figure 9. Après que la DLE a fourni la confirmation d'une transmission, la NLE génère une confirmation "NL-Transmit confirm" à l'attention de son utilisateur demandeur. Il n'y a pas de confirmation de la livraison de la part de la NLE distante.



Légende

Anglais	Français
NL- Transmit confirm	NL- Transmit confirm
NL-Transmit request	NL-Transmit request
NL- Transmit indication	NL- Transmit indication

Figure 9 – Séquences de service de transfert de données NLL

5.2.4 Service NL-TRANSMIT

5.2.4.1 Fonction

Il peut être utilisé pour émettre une NSDU indépendante complète d'une NLE vers une autre NLE dans un seul accès au service et pour retourner le statut de cette livraison à l'utilisateur de NLS émetteur.

5.2.4.2 Types de primitives et paramètres

Le Tableau 18 indique les types de primitives et les paramètres nécessaires au service Transmit.

Tableau 18 – Primitives et paramètres de NL-TRANSMIT

Nom de paramètre	Requête	Indication	Confirmation
Handle	M		M(=) ^a
Source address		M	
Destination address			
Short address	S		
Long address	S		
Broadcast	S		
Priority	M	M(=)	
Timetable_ID	C		
NLS-user-data	M	M(=)	
Status			M
^a La valeur de ce paramètre est égale à la valeur dans la primitive "request".			

5.2.4.3 Paramètres

5.2.4.3.1 Handle

Dans la requête, ce paramètre est pris en charge pour la commodité de l'utilisateur de NLS. Sa valeur dans la primitive "confirm" est identique à la valeur dans la primitive "request" correspondante.

5.2.4.3.2 Source address

Ce paramètre identifie la source de la NSDU acheminée par la primitive "indication".

5.2.4.3.3 Destination address

5.2.4.3.3.1 Short address

Il s'agit du surnom de la NLE de destination.

5.2.4.3.3.2 Long address

Il s'agit de l'ID propre à la NLE de destination.

5.2.4.3.3.3 Broadcast

Toutes les NLE du réseau sont les destinations.

5.2.4.3.4 Priority

Ce paramètre spécifie la priorité de l'émission. Sa valeur est un élément quelconque de l'ensemble: {command, process-data, normal, alarm}.

5.2.4.3.5 Timetable_ID

Ce paramètre spécifie l'indicateur horaire (Timetable) qui a été utilisé pour allouer la largeur de bande de communication réseau pour cette primitive de service. Ce paramètre n'est présent que si l'adresse de destination n'est pas celle du gestionnaire de réseau et si cette demande est requise pour utiliser la largeur de bande allouée.

5.2.4.3.6 NLS-user-data

Ce paramètre permet l'émission de données entre les utilisateurs du NLS sans altération par le fournisseur du NLS.

5.2.4.3.7 Source address

Ce paramètre identifie la source de la NSDU acheminée par la primitive "indication".

5.2.4.3.8 Status

Dans la primitive "confirm", ce paramètre indique le succès, l'avertissement ou l'échec du service. Pour l'avertissement et l'échec, il indique aussi la cause de l'avertissement ou de l'erreur.

5.2.5 NLM-SET

5.2.5.1 Fonction

Cette primitive peut être utilisée pour établir (écrire) la valeur d'un paramètre de configuration de NLE.

5.2.5.2 Types de paramètres

Le Tableau 19 indique la primitive et les paramètres du service SET.

Tableau 19 – Primitive et paramètres de NLM-SET

Nom de paramètre	Requête	
	Entrée	Sortie
NLM-object-identifier	M	
Desired-value	M	
Status		M

5.2.5.3 Paramètres

5.2.5.3.1 NLM-object-identifier

Ce paramètre spécifie la primitive ou l'objet composé au sein de la NLE dont la valeur est à modifier. Le domaine de désignation du paramètre "NLM-object-identifier" est la vue locale NLM.

5.2.5.3.2 Desired-value

Ce paramètre spécifie la valeur désirée pour l'objet NLM spécifié par le paramètre "NLM object-identifier" associé. Son type est identique à celui de l'objet NLM spécifié.

5.2.5.3.3 Status

Ce paramètre permet à l'utilisateur du NLMS de déterminer si le NLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.2.6 NLM-GET

5.2.6.1 Fonction

Cette primitive peut être utilisée pour récupérer (lire) la valeur d'un paramètre de configuration, d'un paramètre opérationnel ou les statistiques d'une NLE.

5.2.6.2 Types de paramètres

Le Tableau 20 indique la primitive et les paramètres du service GET.

Tableau 20 – Primitive et paramètres de NLM-GET

Nom de paramètre	Requête	
	Entrée	Sortie
NLM-object-identifier	M	
Status		M
Current-value		C

5.2.6.3 Paramètres

5.2.6.3.1 NLM-object-identifiant

Ce paramètre spécifie la primitive ou l'objet composé au sein de la NLE dont la valeur est demandée. Le domaine de désignation du paramètre "NLM-object-identifiant" est la vue locale NLM.

5.2.6.3.2 Status

Ce paramètre permet à l'utilisateur du NLMS de déterminer si le NLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.2.6.3.3 Current-value

Ce paramètre est présent lorsque le paramètre de statut indique que le service demandé a été accompli avec succès. Ce paramètre spécifie la valeur actuelle pour l'objet NLM spécifié par le paramètre "NLM-object-identifiant" associé. Son type est identique à celui de l'objet NLM spécifié.

5.2.7 NLM-ACTION

5.2.7.1 Fonction

Cette primitive peut être utilisée pour commander une action spécifiée de la NLE.

5.2.7.2 Types de paramètres

Le Tableau 21 indique les primitives et les paramètres du service Action.

Tableau 21 – Primitive et paramètres de NLM-ACTION

	Requête	Confirmation
Nom de paramètre	Entrée	Sortie
Desired-action	M	
Action-qualifiers	C	
Status		M
Additional-information		C

5.2.7.3 Paramètres

5.2.7.3.1 Desired-action

Ce paramètre spécifie l'action souhaitée que la NLE a à entreprendre. Les valeurs possibles pour ce paramètre sont énumérées au Tableau 59.

5.2.7.3.2 Action-qualifiers

Ce paramètre spécifie des paramètres supplémentaires spécifiques à l'action qui servent à qualifier l'action commandée.

NOTE Certaines des actions exigent des paramètres complémentaires, et ainsi DELETE-SESSION exige Session ID.

5.2.7.3.3 Status

Ce paramètre permet à l'utilisateur du NLMS de déterminer si le NLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.2.7.3.4 Additional-information

Ce paramètre facultatif fournit des informations complémentaires spécifiques à l'action qui augmentent le statut retourné.

5.3 Services de la couche transport

5.3.1 Moyens des services de la couche transport

5.3.1.1 Service transfert de données

La couche transport (TL) prend en charge les services tant acquittés que non acquittés.

Le service Unacknowledged (non acquitté) permet à des dispositifs d'envoyer des paquets sans exiger d'acquiescement de la part de l'extrémité de réception. Les paquets peuvent être livrés au dispositif de destination dans un ordre différent de celui dans lequel ils ont été initialement envoyés du dispositif source. Ce service est utile pour éditer des données de processus. Sachant que les données de processus sont propagées périodiquement, l'acquiescement de bout en bout et les tentatives ont une utilité limitée étant donné qu'un nouveau point de données sera généré sur une base régulière.

Le service acknowledged (acquitté) est utilisé pour construire un tube de transport synchrone à travers le réseau reliant deux dispositifs. Ce tube permet aux dispositifs d'envoyer des paquets et de confirmer leur livraison. L'utilisateur du service de TL (TLS) peut envoyer un paquet qui sera livré à l'utilisateur de TLS de réception sans perte, duplication ou ordre incorrect. Ce service convient à un trafic de demandes et de réponses ainsi qu'à des notifications d'événements. Lorsque le service acquitté est utilisé, la communication est synchrone. Une seule transaction en cours à la fois est autorisée à travers le réseau pour le tube de transport en question.

L'utilisation du service acquitté permet à la couche client de garantir que les répétitions de tentatives sont effectuées automatiquement, si la latence de communication devient excessive. Du point de vue statistique, la fiabilité de la communication pour un réseau maillé de Type 20 bien formé est supérieure à 3σ (3 sigmas = 99,730 020 4 %) et normalement supérieure à 4σ (99,999 5 %). Le principal avantage fourni par un service acquitté est que la couche client recevra un acquiescement positif après l'arrivée à destination de l'unité de donnée de protocole de couche transport (TPDU). Cela permet un fonctionnement synchrone à travers le réseau et donne en même temps l'assurance à la couche client que la TPDU a été délivrée et acquittée.

5.3.1.2 Service de gestion locale de TL

Ce service prend en charge la configuration de la TL et l'accès aux statistiques de la TL. Les moyens de gestion de la TL fournissent des moyens pour

- écrire les paramètres de configuration de l'entité de TL (TLE); et
- lire des paramètres de configuration, des paramètres opérationnels et des statistiques de TLE;

Ces moyens constituent ensemble le service de gestion de NL (NLMS).

5.3.2 QoS pour le service messagerie

5.3.2.1 Priorité

Les primitives du service transfert de données spécifient la priorité de la donnée transférée. Les quatre priorités sont, de la plus haute à la plus basse:

Commande – il convient que l'utilisateur de TLS se serve de cette priorité pour envoyer des données d'informations de diagnostic, de configuration ou de commande relatives au réseau.

Processus de données – il convient que l'utilisateur de TLS se serve de cette priorité pour envoyer des données contenant des mesures issues d'émetteurs de processus ou des valeurs de consigne à des dispositifs de commande ou n'importe quelles autres données relatives au processus ou données statistiques du réseau.

Normal – il convient que l'utilisateur de TLS se serve de cette priorité pour envoyer des données contenant n'importe quelles autres données qui ne remplissent pas les critères relatifs à la priorité "Command", "Process-Data" ou "Alarm".

Alarme – il convient que l'utilisateur de TLS se serve de cette priorité pour envoyer des données contenant seulement une alarme et des informations d'événement.

5.3.2.2 Répétition de tentative

La gestion de TL spécifie les bornes supérieures de la durée maximale permise pour l'achèvement de chaque instance correspondante d'une séquence de primitives de transfert de données. Elle spécifie aussi le nombre maximal de tentatives pour envoyer une demande et recevoir une primitive "response".

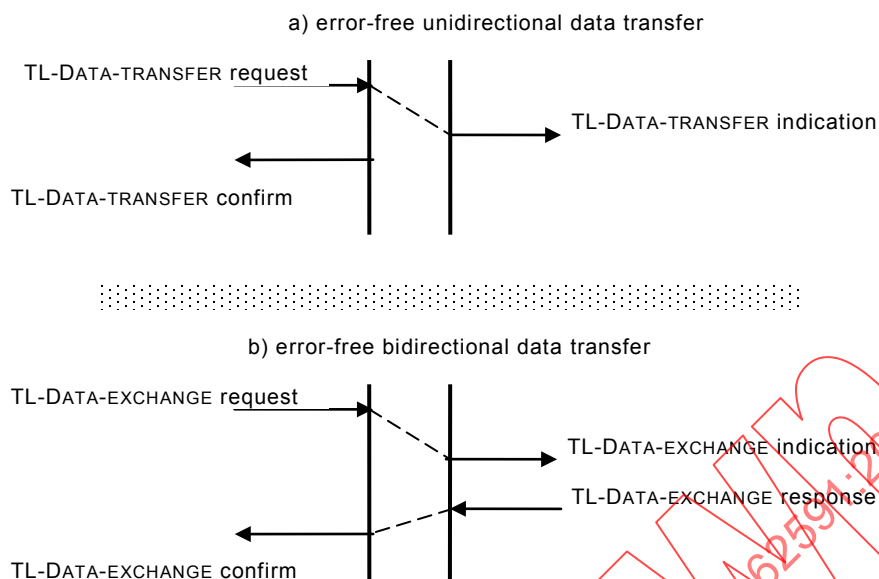
5.3.3 Séquence de primitives

Le présent paragraphe spécifie la relation des primitives au niveau des points d'extrémité du service transfert de données.

Les relations que les primitives de chaque type au niveau de l'interface d'utilisateur du TLS et d'une TLE ont avec les primitives au niveau de l'utilisateur du TLS et des autres TLE pour des cas exempts d'erreurs sont récapitulées dans les diagrammes de la Figure 10.

La séquence a) montre un transfert de données unidirectionnel exempt d'erreurs sans acquittement de la part de la TLE de destination. Après que la NLE a fourni la confirmation d'une transmission, la TLE génère une confirmation "TL-DATA-TRANSFER confirm" à l'attention de son utilisateur demandeur.

La séquence b) montre un échange de données acquitté bidirectionnel sans erreurs. Dans cette séquence, la demande "TL-DATA-EXCHANGE request" au niveau de la TLE source insère le message dans la file d'attente d'émission de la TLE. Lorsqu'une TPDU est reçue et validée, la TLE de destination génère une indication "TL-DATA-EXCHANGE indication". L'utilisateur de TLS à la destination en question est censé fournir des données de réponse. La TLE émet cette réponse vers la TLE source. À la réception de la réponse, la TLE source génère une confirmation "TL-DATA-EXCHANGE confirm" à l'attention de l'utilisateur.

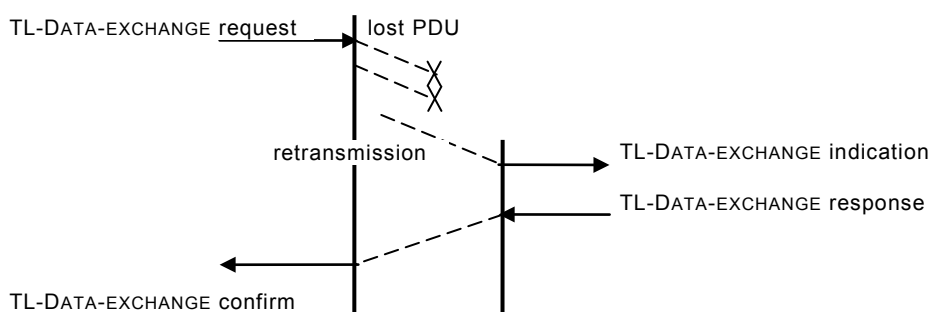


Légende

Anglais	Français
TL-Data-transfer confirm	TL-Data-transfer confirm
TL-Data-transfer request	TL-Data-transfer request
TL-Data-transfer indication	TL-Data-transfer indication
a) error-free unidirectional data transfer	a) transfert de données unidirectionnel sans erreurs
b) error-free bidirectional data transfer	b) transfert de données bidirectionnel sans erreurs
TL-Data-exchange response	TL-Data-exchange response
TL-Data-exchange request	TL-Data-exchange request
TL-Data-exchange confirm	TL-Data-exchange confirm
TL-Data-exchange indication	TL-Data-exchange indication

Figure 10 – Séquences sans erreurs du service transfert de données

Les relations que les primitives de chaque type au niveau de l'interface d'utilisateur du TLS et d'une TLE ont avec les primitives au niveau de l'utilisateur du TLS et des autres TLE pour des cas d'erreurs sont récapitulées dans le diagramme de la Figure 11. La séquence montre l'unité de données de protocole de couche transport (TPDU) perdue dans la transmission. Comme montré, la TLE source réémet la TPDU tant qu'une TPDU de réponse n'a pas été reçue ou tant qu'une limite sur le nombre maximal de tentatives n'a pas été atteinte. Si le paquet séjourne dans la TLE source sans réponse de la part de la TLE de destination pendant une durée supérieure à la valeur spécifiée pour l'expiration de délai, la TLE cesse ses tentatives.



Légende

Anglais	Français
TL-Data-exchange request	TL-Data-exchange request
lost PDU	PDU perdue
retransmission	retransmission
TL-Data-exchange indication	TL-Data-exchange indication
TL-Data-exchange response	TL-Data-exchange response
TL-Data-exchange confirm	TL-Data-exchange confirm

Figure 11 – Séquences avec erreurs du service transfert de données

5.3.4 Service TL-DATA-TRANSFER

5.3.4.1 Fonction

Il peut être utilisé pour émettre une unité de donnée de service de TL (TSDU) de demande indépendante complète d'une TLE vers une autre TLE dans un seul accès au service sans confirmation de livraison.

5.3.4.2 Types de primitives et paramètres

Le Tableau 22 indique les types de primitives et les paramètres nécessaires au service Data-transfer.

Tableau 22 – Primitives et paramètres de TL-DATA-TRANSFER

Nom de paramètre	Requête	Indication	Confirmation
Handle	M	M(=) ^a	M(=) ^b
Transport type	M	M	
Source address		M	
Destination address			
Short address	S		
Long address	S		
Broadcast	S		
Priority	M	M(=)	
Timetable_ID	C		
TLS-user-data	M	M(=)	
Status			M
^a Ce paramètre dans une indication est considéré comme le numéro de séquence.			
^b La valeur de ce paramètre est égale à la valeur dans la primitive "request".			

5.3.4.3 Paramètres

5.3.4.3.1 Handle

Dans la requête, ce paramètre est pris en charge pour la commodité de l'utilisateur de DLS. Sa valeur dans la primitive "confirm" est identique à la valeur dans la primitive "request" correspondante. Dans la primitive "indication", ce paramètre est utilisé pour fournir un numéro de séquence.

5.3.4.3.2 Transport type

Dans la primitive "request", ce paramètre indique le type de service demandé. Il a l'une des valeurs suivantes:

- Block-transfer request (demande de transfert par blocs),
- Block-transfer response (réponse de transfert par blocs),
- Search-broadcast (recherche en diffusion),
- Publish-broadcast (édition en diffusion),
- Response-broadcast (réponse en diffusion),
- Publish-unicast (édition en monodiffusion);

5.3.4.3.3 Source address

Ce paramètre identifie la source de la TSDU acheminée par la primitive "indication".

5.3.4.3.4 Destination address

5.3.4.3.4.1 Short address

Il s'agit du surnom de la TLE de destination.

5.3.4.3.4.2 Long address

Il s'agit de l'ID propre à la TLE de destination.

5.3.4.3.4.3 Broadcast

Toutes les TLE du réseau sont les destinations.

5.3.4.3.5 Priority

Ce paramètre spécifie la priorité de l'émission. Sa valeur est un élément quelconque de l'ensemble: {command, process-data, normal, alarm}.

5.3.4.3.6 Timetable_ID

Ce paramètre spécifie l'indicateur horaire (Timetable) qui a été utilisé pour allouer la largeur de bande de communication réseau pour cette primitive de service. Ce paramètre n'est présent que si l'adresse de destination n'est pas celle du gestionnaire de réseau.

5.3.4.3.7 TLS-user-data

Ce paramètre permet l'émission de données entre les utilisateurs du TLS sans altération par le fournisseur du TLS.

5.3.4.3.8 Source address

Ce paramètre identifie la source de la TSDU acheminée par la primitive "indication".

5.3.4.3.9 Status

Dans la primitive "confirm", ce paramètre indique le succès, l'avertissement ou l'échec du service. Pour l'avertissement et l'échec, il indique aussi la cause de l'avertissement ou de l'erreur.

5.3.5 Service TL-DATA-EXCHANGE

5.3.5.1 Fonction

Il peut être utilisé pour émettre une TSDU de demande indépendante complète d'une TLE vers une autre TLE dans un seul accès au service, pour recevoir une TSDU de réponse de la part de la TLE de destination et pour retourner le statut de cet échange à l'utilisateur de TLS émetteur.

5.3.5.2 Types de primitives et paramètres

Le Tableau 23 indique les types de primitives et les paramètres nécessaires au service TL-DATA-EXCHANGE.

Tableau 23 – Primitives et paramètres de TL-DATA-EXCHANGE

Nom de paramètre	Requête	Indication	Réponse	Confirmation
Handle	M	M(=) ^a	M(=)	M(=) ^b
Transport type	M	M(=)		
Source address		M		
Destination address				
Short address	S			
Long address	S			
Broadcast	S			
Priority	M	M(=)		
Timetable_ID	C			
TLS-user-data	M	M(=)	M	C(=)
Status				M
^a Ce paramètre dans une indication est considéré comme le numéro de séquence.				
^b Ce paramètre est égal à la valeur dans la primitive "request".				

5.3.5.3 Paramètres

5.3.5.3.1 Handle

Dans la requête, ce paramètre est pris en charge pour la commodité de l'utilisateur de DLS. Sa valeur dans la primitive "confirm" est identique à la valeur dans la primitive "request" correspondante. Dans la primitive "indication", ce paramètre est utilisé pour associer la demande à la réponse.

5.3.5.3.2 Transport type

Dans la primitive "request", ce paramètre indique le type de service demandé par le maître. Il a l'une des valeurs suivantes:

- Request-unicast (demande en monodiffusion)
- Request-broadcast (demande en diffusion).

Dans la primitive "response", ce paramètre indique le type de service demandé par l'esclave. Sa valeur est – Response-unicast.

5.3.5.3.3 Source address

Ce paramètre identifie la source de la TSDU acheminée par la primitive "indication".

5.3.5.3.4 Destination address

5.3.5.3.4.1 Short address

Il s'agit du surnom de la TLE de destination.

5.3.5.3.4.2 Long address

Il s'agit de l'ID propre à la TLE de destination.

5.3.5.3.4.3 Broadcast

Toutes les TLE du réseau sont les destinations.

5.3.5.3.5 Priority

Ce paramètre spécifie la priorité de l'émission. Sa valeur est un élément quelconque de l'ensemble: {command, process-data, normal, alarm}.

5.3.5.3.6 Timetable_ID

Ce paramètre spécifie l'indicateur horaire (Timetable) qui a été utilisé pour allouer la largeur de bande de communication réseau pour cette primitive de service. Ce paramètre n'est présent que si l'adresse de destination n'est pas celle du gestionnaire de réseau.

NOTE L'allocation de largeur de bande pour la communication est demandée par l'initiateur de la transaction et la réponse n'est incluse dans l'allocation pour aucun indicateur horaire dans le dispositif. Par conséquent, Timetable_ID n'est pas inclus dans la primitive "response".

5.3.5.3.7 TLS-user-data

Ce paramètre permet l'émission de données entre les utilisateurs du TLS sans altération par le fournisseur du TLS.

5.3.5.3.8 Source address

Ce paramètre identifie la source de la TSDU acheminée par la primitive "indication".

5.3.5.3.9 Status

Dans la primitive "confirm", ce paramètre indique le succès, l'avertissement ou l'échec du service. Pour l'avertissement et l'échec, il indique aussi la cause de l'avertissement ou de l'erreur.

5.3.6 TLM-SET

5.3.6.1 Fonction

Cette primitive peut être utilisée pour établir (écrire) la valeur d'un paramètre de configuration de TLE.

5.3.6.2 Types de paramètres

Le Tableau 24 indique la primitive et les paramètres du service SET.

Tableau 24 – Primitive et paramètres de TLM-Set

Nom de paramètre	Requête	
	Entrée	Sortie
TLM-object-identifier	M	
Desired-value	M	
Status		M

5.3.6.3 Paramètres

5.3.6.3.1 TLM-object-identifier

Ce paramètre spécifie la primitive ou l'objet composé au sein de la TLE dont la valeur est à modifier. Le domaine de désignation du paramètre " TLM-object-identifier" est la vue locale TLM.

5.3.6.3.2 Desired-value

Ce paramètre spécifie la valeur désirée pour l'objet TLM spécifié par le paramètre "TLM object-identifier" associé. Son type est identique à celui de l'objet TLM spécifié.

5.3.6.3.3 Status

Ce paramètre permet à l'utilisateur du Service de gestion de TL (TLMS) de déterminer si le TLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.3.7 TLM-GET

5.3.7.1 Fonction

Cette primitive peut être utilisée pour récupérer (lire) la valeur d'un paramètre de configuration, d'un paramètre opérationnel ou les statistiques d'une TLE.

5.3.7.2 Types de paramètres

Le Tableau 25 indique la primitive et les paramètres du service GET.

Tableau 25 – Primitive et paramètres de TLM-Get

Nom de paramètre	Requête	
	Entrée	Sortie
TLM-object-identifier	M	
Status		M
Current-value		C

5.3.7.3 Paramètres

5.3.7.3.1 TLM-object-identifier

Ce paramètre spécifie la primitive ou l'objet composé au sein de la TLE dont la valeur est demandée. Le domaine de désignation du paramètre "TLM-object-identifier" est la vue locale TLM.

5.3.7.3.2 Status

Ce paramètre permet à l'utilisateur du TLMS de déterminer si le TLMS demandé a été fourni avec succès ou a échoué.

NOTE La cause d'échec peut aussi être fournie.

5.3.7.3.3 Current-value

Ce paramètre est présent lorsque le paramètre de statut indique que le service demandé a été accompli avec succès. Ce paramètre spécifie la valeur actuelle pour l'objet TLM spécifié par le paramètre " TLM-object-identifier" associé. Son type est identique à celui de l'objet TLM spécifié.

6 Spécification de protocole de couche liaison de données – Éléments de Type 20

6.1 Vue d'ensemble

6.1.1 Fonctionnement de la couche liaison de données

6.1.1.1 Modèle de DLL à deux couches

La DLL est modélisée comme étant

- une sous-couche supérieure de commande de liaison logique (LLC), et
- une sous-couche inférieure de contrôle d'accès au support (MAC).

Les fonctions de gestion de DL inter fonctionnent avec toutes les sous-couches.

6.1.1.2 Commande de liaison logique

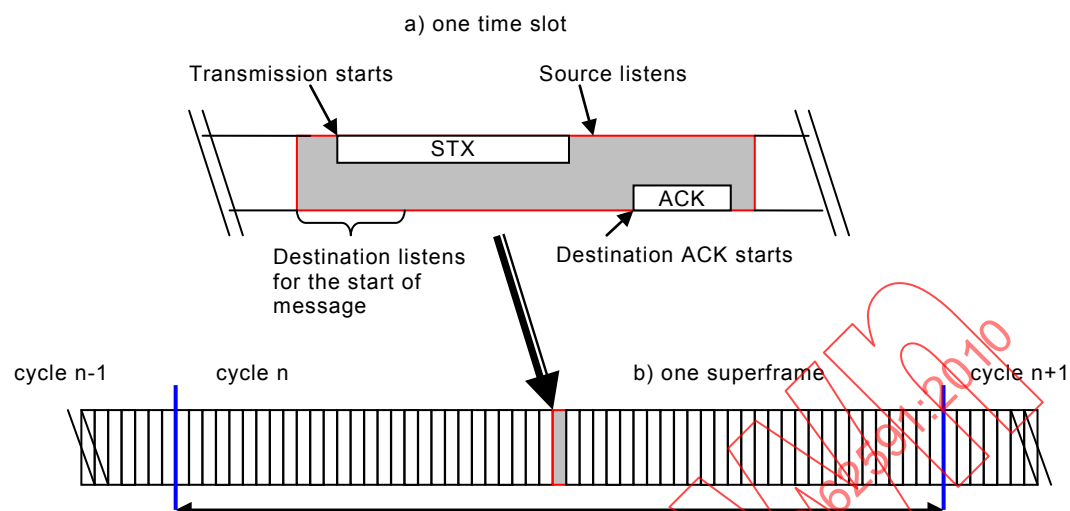
La sous-couche LLC fournit la fonctionnalité de plus haut niveau permettant de

- gérer toutes les interactions de DLE avec l'utilisateur de DLS, convertir toutes les primitives de demande et de réponse de l'utilisateur de DLS en la nécessaire séquence d'opérations de DLE; et générer des primitives d'indication et de confirmation de l'utilisateur de DLS s'il y a lieu;
- préparer la DLPDU pour l'émission;
- analyser la DLPDU reçue;
- réguler le débit;
- détecter les erreurs;
- maintenir le tableau de voisins; et
- garantir la sécurité de la couche liaison de données.

6.1.1.3 Contrôle d'accès au support

La couche liaison de données de Type 20 utilise l'Accès multiple par répartition dans le temps (TDMA) et le saut de voie pour commander l'accès à la liaison. Cette technique de Contrôle d'accès au support (MAC) assure une communication déterministe sans collision entre deux dispositifs. La communication entre deux dispositifs a lieu pendant un intervalle de temps. Une supertrame TDMA consiste en une série d'intervalles de temps. La structure de base d'intervalles de temps TDMA et la supertrame sont montrées à la Figure 12. Les tailles d'intervalles de temps et la longueur de supertrame en nombre d'intervalles de temps sont fixes et forment un cycle de liaison. Tous les dispositifs prennent en charge plusieurs supertrames, en commençant par la supertrame numéro zéro (0). Une supertrame au moins est toujours activée alors que des supertrames supplémentaires peuvent être activées ou désactivées. La longueur d'une supertrame ne peut être modifiée que pendant qu'elle est

active. Une supertrame peut avoir des longueurs différentes les unes des autres. Les supertrames se répètent en continu avec une fréquence de répétition fixe.



Légende

Anglais	Français
STX	STX
ACK	ACK
Transmission starts	L'émission commence.
Source listens	La source écoute.
Destination ACK starts	L'ACK de destination commence.
a) one time slot	a) un intervalle de temps
b) one superframe	b) une supertrame
cycle n	cycle n
cycle n-1	cycle n-1
cycle n+1	cycle n+1
Destination listens for the start of message	La destination se met à l'écoute pour détecter le début de message.

Figure 12 – Intervalle de temps TDMA et supertrame

Typiquement, deux dispositifs sont attribués à un même intervalle de temps. L'un est désigné comme étant la source et l'autre comme la destination. Une transaction de liaison dans un intervalle de temps prend en charge l'émission d'une DLPDU par un dispositif source et, le cas échéant, elle est immédiatement suivie de l'émission d'une DLPDU d'acquittement (ACK) par le dispositif de destination. La DLPDU ACK contient soit le code de réponse "Success" indiquant que la DLPDU initiale a été reçue et gérée avec succès, soit un code de réponse d'erreur. Un code de réponse d'erreur indique que la DLPDU initiale a été reçue avec succès, mais que le traitement ultérieur a échoué, par exemple il n'y avait pas de tampons disponibles dans le dispositif de réception.

Un message de diffusion utilisant l'adresse de destination de la liaison de données comme adresse de diffusion n'est jamais acquitté. Dans ce cas, plusieurs récepteurs sont attribués au même intervalle de temps.

Pour des communications TDMA réussies et efficaces, il est impératif que les dispositifs connaissent l'instant auquel un intervalle de temps commence. Cela exige la synchronisation des horloges parmi tous les dispositifs du réseau. Pour assurer la synchronisation des

horloges des dispositifs à l'échelle du réseau, les tolérances sur les mécanismes de maintien de l'heure et de synchronisation du temps sont spécifiées dans ce protocole.

Dans un intervalle de temps, l'émission de la DLPDU source débute à un décalage spécifié par rapport au début de l'intervalle. Ce décalage permet à la source et à la destination de régler leur voie de fréquence et permet au récepteur de commencer l'écoute sur la voie spécifiée. Sachant qu'il existe une tolérance sur les horloges, le récepteur commence à écouter avant l'instant idéal de début d'émission et continue l'écoute après l'instant idéal en question. Une fois l'émission terminée, le sens de la communication est inversé et le dispositif de destination indique, en émettant un ACK, s'il a reçu la DLPDU du dispositif source avec succès ou avec une classe spécifique d'erreurs détectées. Si la destination n'a pas entendu d'émission ou s'il y a eu une erreur de réception qui n'est pas incluse dans cette classe d'erreurs, le dispositif de destination n'émet pas de DLPDU ACK.

6.1.1.4 Saut de voie

Pour renforcer la fiabilité, le saut de voie est combiné au TDMA comme montré à la Figure 13. Le saut de voie fournit la diversité de fréquence, ce qui peut éviter les systèmes perturbateurs et réduire les effets d'évanouissement par trajets multiples. Le TDMA permet une communication à saut de voie efficace, de faible puissance et fiable car la synchronisation de l'intervalle de temps et de la voie utilisée par les dispositifs en communication leur permet d'effectuer un rendez-vous en temps et en fréquence, ce qui favorise le succès des communications.

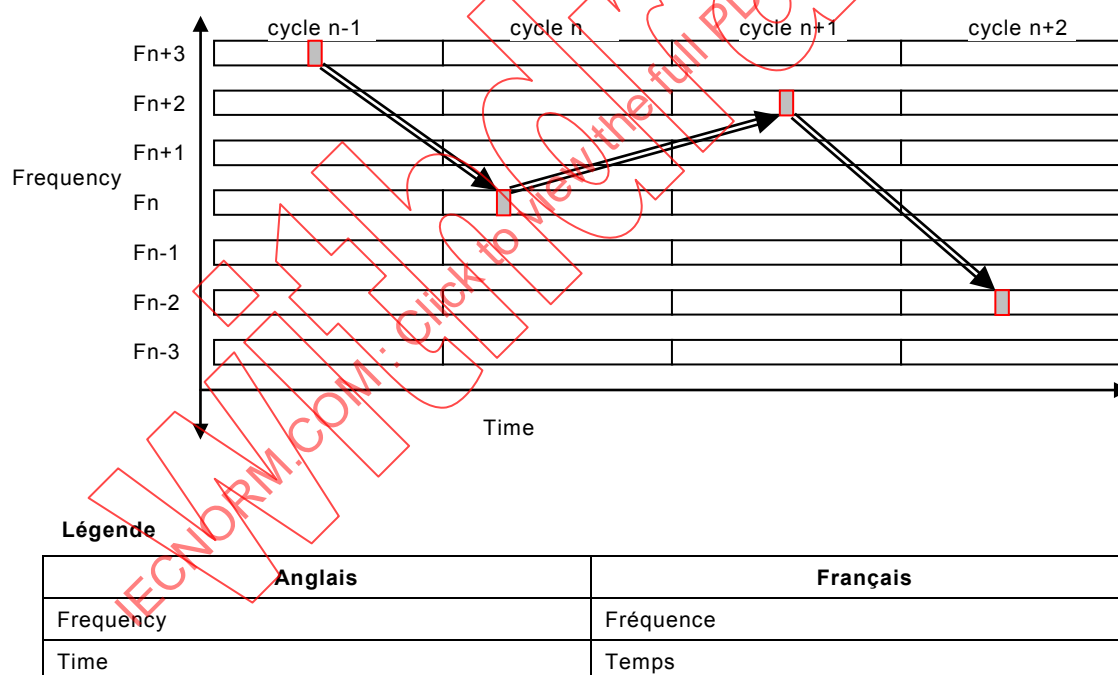


Figure 13 – Saut de voie

Des dispositifs en communication sont attribués à une supertrame, à un intervalle de temps dans la supertrame en question et à un décalage de voie. Ce trio forme une liaison de communications entre deux dispositifs en communication. Tous les dispositifs prennent en charge les liaisons multiples. Le nombre de liaisons possibles est, typiquement, égal au nombre de voies utilisées par un réseau multiplié par le nombre d'intervalles de temps de la supertrame. Par exemple, avec 15 voies et 9 000 intervalles de temps par supertrame, il y a 135 000 liaisons possibles..

Le saut de voie fournit la diversité de voies et, de ce fait, chaque intervalle de temps est utilisé sur plusieurs voies en même temps par des nœuds différents. Cela peut s'obtenir par la création de liaisons sur le même intervalle de temps, mais avec des décalages de voie différents. Chaque dispositif maintient une liste de voie en utilisation et la spécification (la

fréquence par exemple) pour la voie en question. Tous les dispositifs dans un réseau ont des listes de voies identiques qui sont déterminées par le gestionnaire de réseau. L'affectation des liaisons et des dispositifs dans une liaison est de la responsabilité du gestionnaire de réseau.

La mise en liste noire d'une voie est une caractéristique de ce protocole qui permet à l'administrateur réseau de limiter le saut de voie par les dispositifs à des voies sélectionnées dans la bande RF disponible. Par exemple, des administrateurs réseau peuvent placer en liste noire certaines voies afin de protéger un service sans fil qui utilise une partie fixe de la bande RF qui aurait pu être partagée par les dispositifs dans ce réseau. Dans la pratique, la communication dans la largeur de bande disponible par les dispositifs utilisant d'autres protocoles est très aléatoire et utilise une quantité infime de la largeur de bande disponible totale. Par conséquent, la liste noire procure rarement des avantages tangibles.

6.1.1.5 Maintenance de réseau

Le temps et la voie sont les deux premières dimensions d'un réseau de Type 20. La troisième dimension est l'espace (la distance). Les dispositifs de réseau sont installés en divers emplacements dans une installation et un dispositif donné a un ensemble d'autres dispositifs avec lesquels il peut communiquer directement. Cet ensemble de dispositifs est défini comme étant le voisinage. Sachant que l'environnement RF est susceptible de changer, le dispositif maintient courante sa liste de voisins. Les activités de maintenance incluent de découvrir des voisins potentiels, de rassembler des statistiques relatives à la voie de communication menant à chaque voisin et de maintenir la synchronisation du temps avec les voisins.

Deux DLPDU spéciales aident à dresser et maintenir une liste de voisins d'un dispositif. Il s'agit de la DLPDU Advertise (c'est-à-dire d'annonce) et de la DLPDU Keep-alive (c'est-à-dire d'entretien). Ces DLPDU contiennent des informations suffisantes pour permettre de découvrir de nouveaux voisins ou pour permettre à un dispositif nouvellement installé de demander son admission au réseau. S'il est accepté, le nouveau dispositif peut devenir un voisin du dispositif annonceur.

NOTE Le gestionnaire de réseau peut programmer l'émission de ces DLPDU.

Chaque communication réussie avec un voisin confirme la présence du voisin et permet d'évaluer la de la liaison de communication. Les communications n'ayant lieu que lorsque le dispositif a une DLPDU pour le voisin, la liaison peut rester non utilisée pendant de longs intervalles. Les DLPDU Keep-alive sont utilisées pour sonder les liaisons au repos et pour maintenir la synchronisation du temps.

6.1.1.6 Conservation du temps

La synchronisation du temps à travers le réseau est essentielle pour les communications TDMA. Quel que soit le choix de la source matérielle de temps du dispositif (par exemple quartz, résonateurs céramiques, etc.), un certain décalage entre dispositifs (en raison de variations de température ou de tension ou du vieillissement par exemple) est inévitable. En conséquence, la couche liaison de données de Type 20 a plusieurs mécanismes pour synchroniser le temps à l'échelle du réseau.

Lorsqu'un dispositif reçoit une DLPDU qui lui est destinée, son heure d'arrivée est notée. En utilisant cette information, la destination calcule la différence entre le temps réel d'arrivée et le temps prévu d'arrivée pour la DLPDU en question. Ce delta-t (Δt) est communiqué dans chaque DLPDU de réponse d'ACK envoyée au dispositif source. Ainsi, chaque transaction acquittée mesure l'alignement du temps réseau entre les dispositifs.

La liste de voisins contient des voisins sélectionnés, spécifiés par le gestionnaire de réseau, qui sont utilisés comme sources de synchronisation du temps. Lorsqu'une DLPDU issue d'un voisin de synchronisation du temps est reçue, l'heure réseau du dispositif récepteur est réajustée. La synchronisation du temps repose soit sur l'heure d'arrivée de la DLPDU, soit sur le delta-t dans la DLPDU ACK, cela dépendant du dispositif qui a lancé la transaction.

En outre, il convient que les concepteurs de dispositifs comprennent les caractéristiques de dérive du temps de leurs produits. Lorsque la source de temps du dispositif dérive, le dispositif émet une DLPDU Keep-alive, au besoin, vers ses voisins de synchronisation du temps afin de maintenir la synchronisation du temps. Il convient que, par construction, un dispositif ne nécessite pas de DLPDU Keep-alive à une fréquence supérieure à une fois toutes les 30 s alors que la température varie de 2° C/min ou moins. De surcroît, il convient que les conceptions de dispositifs tolèrent une répétition de tentative en cas de perte de paquet (à savoir une marge de sécurité de 10 s). Cela correspond à approximativement une précision d'horloge compensée de 10 parties par million ou meilleure.

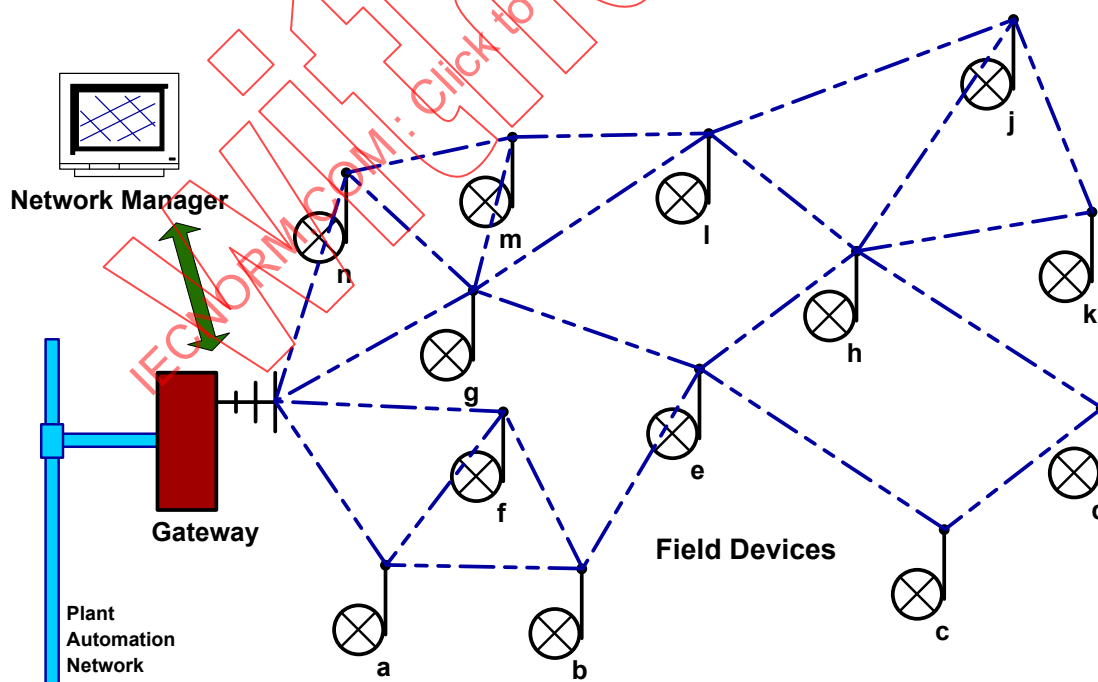
6.1.2 Fonctionnement de la couche réseau

6.1.2.1 Généralités

Le réseau Type 20 est un système de communication maillé de capteurs et d'actionneurs qui simplifie l'installation du réseau et des dispositifs et permet à l'utilisateur final de personnaliser l'installation et sa topologie pour satisfaire à des exigences d'application spécifiques. Les éléments de base de ce réseau comprennent les dispositifs de terrain, la passerelle et le gestionnaire de réseau. Chaque réseau comporte au moins une passerelle. Alors que des gestionnaires de réseau redondants sont pris en charge, il n'y a qu'un seul gestionnaire de réseau actif par réseau. Les passerelles peuvent inclure un ou plusieurs points d'accès.

6.1.2.2 Réseau maillé

La Figure 14 montre un exemple de réseau avec les dispositifs de terrain déployés en une topologie maillée. Cet exemple de réseau comporte une passerelle, un gestionnaire de réseau et treize dispositifs de terrain (étiquetés a à n). Toute la communication a lieu en déplaçant des paquets de la passerelle, en passant par les dispositifs intermédiaires, jusqu'au dispositif de terrain de destination du paquet. Chaque déplacement d'un paquet d'un dispositif vers un autre le long du chemin menant à la destination finale du paquet est appelé "saut".



Légende

Anglais	Français
Plant Automation Network	Réseau d'automation d'installation
Network Manager	Gestionnaire de réseau

Anglais	Français
Gateway	Passerelle
Field Devices	Dispositifs de terrain

Figure 14 – Réseau sans fil

Tous les dispositifs sont capables d'être des sources et des puits pour les paquets et sont capables d'acheminer les paquets reçus en provenance d'autres dispositifs dans le réseau. L'acheminement des paquets de leur source initiale jusqu'à leur destination finale peut nécessiter plusieurs sauts. L'acheminement des paquets est de la responsabilité de la couche réseau.

Au sein de ce réseau, les nœuds 'a', 'f', 'g' et 'n' forment un saut à partir de la passerelle. Étant donné que ces dispositifs peuvent transmettre des paquets directement à la passerelle, la communication avec ces dispositifs a la latence la plus faible. Cependant, étant donné que ce réseau utilise une topologie maillée, les liaisons redondantes sont incluses afin d'améliorer la fiabilité du système en permettant l'acheminement des paquets en contournant les perturbations. Dans la Figure 14, le nœud 'a' peut communiquer directement avec la passerelle mais peut aussi communiquer via le nœud f avec la passerelle, si le chemin direct devient bloqué.

Les nœuds 'c', 'd', 'j' et 'k' se situent à plusieurs sauts de la passerelle. Tous les dispositifs intermédiaires (les dispositifs de terrain 'g' et 'e' par exemple) sont capables de recevoir et transmettre les paquets en provenance et à destination de ces dispositifs. Étant donné que les paquets doivent faire plusieurs sauts depuis les nœuds 'c', 'd', 'j' et 'k' et vers ceux-ci, la latence de communication par rapport à ces dispositifs est supérieure à celle relative aux dispositifs qui se trouvent à une distance d'un saut. Cependant, la topologie maillée permet d'agrandir la taille physique du réseau et permet la communication autour d'obstacles. De surcroît, le nombre de dispositifs pouvant être pris en charge dans un même réseau maillé peut être plus grand que celui de ceux pouvant être pris en charge par un seul réseau en étoile.

Le protocole pour la topologie maillée peut aussi être utilisé dans une topologie en étoile, c'est-à-dire que tous les dispositifs sont à un saut seulement de la passerelle pour prendre en charge une application haute performance. Il peut aussi être utilisé dans une topologie maillée exagérément connectée à sauts multiples pour une application moins exigeante (par exemple surveillance) ou n'importe quelle topologie entre les deux. Ce protocole est suffisamment flexible pour qu'une diversité d'applications (tant haute que basse performance) puisse fonctionner dans le même réseau.

Le réseau sans fil maintient une très haute fiabilité en utilisant plusieurs mécanismes comportant des trajets multiples menant aux dispositifs de réseau, des fréquences multiples et des répétitions de tentatives. Si une fiabilité améliorée est exigée, d'autres trajets peuvent être insérés en ajoutant des dispositifs de terrain supplémentaires. La passerelle utilise un point d'accès pour se connecter au réseau et peut avoir plus d'un point d'accès (voir Annexe D). Des points d'accès réseau supplémentaires peuvent être utilisés pour augmenter le débit et réduire la latence.

6.1.3 Procédures relatives aux dispositifs

Plusieurs procédures au niveau dispositif sont spécifiées dans l'Annexe B.

6.2 Commande de liaison logique

6.2.1 Structure générale de DLPDU

6.2.1.1 Vue d'ensemble

La Figure 15 montre la structure commune aux DLPDU. La DLE doit fournir les octets à la Couche physique dans la même séquence comme montré dans cette figure. L'octet de poids fort d'un champ multi-octet doit être émis le premier, sauf spécification contraire.

Chaque DLPDU doit être constituée de champs dans une séquence décrite ci-dessous:

- Un seul octet mis à 0x41 (qui signifie mode de donnée activé conformément à l'IEEE 802.15.4-2006, Tableau 79 et sécurité activée conformément à l'IEEE 802.15.4-2006, 7.2.1.1.2),
- Un spécificateur d'adresse à 1 octet,
- Un numéro de séquence (Sequence Number) de 1 octet,
- Le network_ID (identificateur de réseau) de 2 octets,
- Les adresses de destination et de source, chacune pouvant avoir une longueur de 2 octets ou de 8 octets,
- Un spécificateur de DLPDU de 1 octet,
- La charge utile de la DLL,
- Un code d'intégrité de message (MIC) chiffré de 4 octets, et
- Un CRC16 de l'UIT-T de 2 octets.

0x41	Address specifier	Sequence number	Network_ID	Destination address	Source address	DLPDU specifier	DLL payload	MIC	CRC

Légende

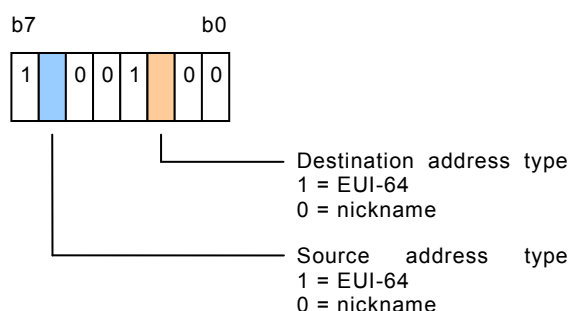
Anglais	Français
Address specifier	Spécificateur d'adresse
Sequence number	Numéro de séquence
Network_ID	Network_ID
Destination address	Adresse de destination
Source address	Adresse source
DLPDU specifier	Spécificateur de DLPDU
DLL payload	Charge utile de DLL
MIC	MIC
CRC	CRC

Figure 15 – Structure de DLPDU

6.2.1.2 Champs de la structure de DLPDU

6.2.1.2.1 Spécificateur d'adresse (Address specifier)

Ce champ spécifie la longueur et le type des adresses de destination et de source comme montré à la Figure 16. Le bit 6 spécifie le type d'adresse source; Le bit -2 spécifie le type d'adresse de destination. Si la valeur de l'un de ces deux bits est '1', l'adresse dans la DLPDU correspondant au bit en question est EUI-64, autrement il s'agit d'un surnom (voir 6.2.1.2.4). N'importe quelle combinaison de longueurs d'adresse peut être utilisée dans une DLPDU.



Légende

Anglais	Français
Destination address type 1 = EUI-64 0 = nickname	Type de l'adresse de destination 1 = EUI-64 0 = surnom
Source address type 1 = EUI-64 0 = nickname	Type de l'adresse source 1 = EUI-64 0 = surnom

Figure 16 – Spécificateur d'adresse

Les autres bits de ce champ sont mis comme indiqué à la Figure 16.

6.2.1.2.2 Numéro de séquence (Sequence number)

Le numéro de séquence doit être égal à l'octet de poids faible du Nombre d'intervalles de temps absolus (ASN).

6.2.1.2.3 Network_ID (identificateur de réseau)

Tous les réseaux sont identifiés à l'aide d'un nombre de deux octets. Un dispositif connecté à un réseau donné ne doit pas envoyer une PDU directement à un autre dispositif connecté à un réseau différent.

La valeur à 2 octets est émise, l'octet de poids faible (LSB) le premier, dans toutes les DLPDU. Si le network_ID ne concorde pas avec celui du réseau dont le dispositif est membre, le paquet doit être rejeté. Les plages des valeurs de network_ID et leur application sont montrées au Tableau 26.

Sauf note spécifique contraire, l'octet de poids fort de la DLPDU doit être émis le premier. Le network_ID doit être émis avec l'octet de poids faible en premier.

Tableau 26 – Allocation de network_ID

Plage des network_ID	Application des network_ID
00 000-32 767 (0x0000-0x7FFF)	Réseaux permanents définis par l'utilisateur (réseaux critiques)
32 768-36 863 (0x8000-0x8FFF)	Réseaux temporaires définis par l'utilisateur (démonstrations, salons commerciaux, essais de terrain, etc.)
36 864-57 343 (0x9000-0xDFFF)	Réservé
57 344-61 439 (0xE000-0xEFFF)	Réseaux de fabrication (non publics utilisés par les fabricants de dispositifs)
61 440-65 535 (0xF000-0xFFFF)	Réservé

6.2.1.2.4 Adresse de destination et adresse source (Destination and source addresses)

La DLL prend en charge deux types d'adresses: une adresse surnom de 2 octets et une adresse IEEE EUI-64 de 8 octets.

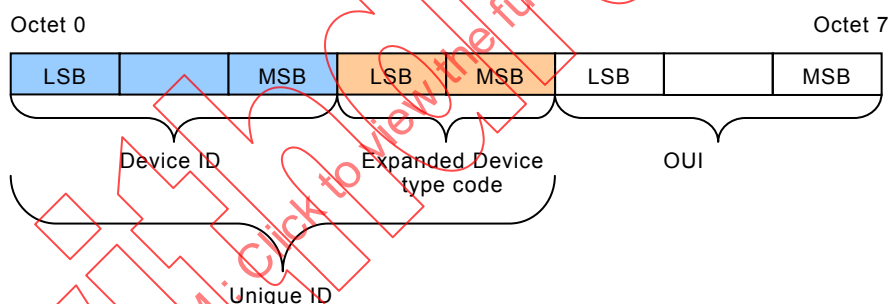
Le surnom de 2 octets est une adresse attribuée à un dispositif de réseau spécifique (voir 3.3.67). Il est nécessaire qu'il soit unique seulement au sein du réseau auquel le dispositif est connecté. La valeur 0xFFFF est réservée comme adresse de diffusion.

L'EUI-64 doit être une concaténation de la valeur à 3 octets de l'identificateur propre à une organisation (l'OUI) telle que définie par l'IEEE 802 et d'un ID propre de 5 octets (voir 3.3.91) comme montré à la Figure 17. L'ID propre doit être une concaténation du code de type de Dispositif étendu (Expanded Device) de 2 octets et de l'ID de dispositif (Device ID) de 3 octets.

NOTE 1 Deux dispositifs quelconques conformes à la présente norme n'ont pas d'adresse EUI-64 identique.

NOTE 2 La valeur de l'OUI est donnée par le Registration Authority Committee de l'IEEE – voir <http://standards.ieee.org/regauth/tutorials.html>.

Le code Expanded Device Type (type de dispositif étendu) doit être attribué de manière univoque pour des dispositifs au sein d'un réseau et des dispositifs dans la portée d'autres réseaux. Chaque dispositif fabriqué avec le code de type de dispositif étendu identique doit recevoir un ID de dispositif qui est unique parmi toutes les instances du type de dispositif en question.



Légende

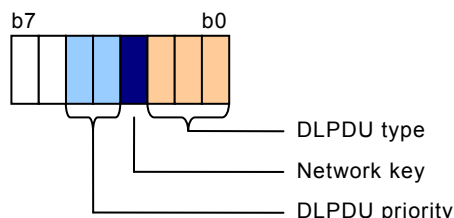
Anglais	Français
LSB	LSB
MSB	MSB
Device ID	ID de dispositif
Expanded Device type code	Code de type pour dispositif étendu
OUI	OUI
Octet 0	Octet 0
Unique ID	ID unique (ID propre)
Octet 7	Octet 7

Figure 17 – Construction de l'adresse EUI-64 de 8 octets

L'IEEE STD 802.15.4-2006 exige que les champs à octets multiples soient émis le LSB en premier. Par conséquent, l'adresse EUI-64 dans la DLPDU doit être émise dans la séquence montrée à la Figure 17 c'est-à-dire commençant par le LSB de l'ID de dispositif et se terminant par le MSB de l'OUI. Le surnom doit aussi être émis en commençant par le LSB.

6.2.1.2.5 Spécificateur de DLPDU (DLPDU specifier)

Ce champ est montré à la Figure 18. Les deux bits de poids fort sont réservés pour usage futur. La DLE doit mettre la valeur de ces bits à '00' avant l'émission. La DLE doit ignorer toute valeur de ces deux bits.



Légende

Anglais	Français
DLPDU type	Type de la DLPDU
Network key	Clé de réseau
DLPDU priority	Priorité de la DLPDU

Figure 18 – Spécificateur de DLPDU

Les bits 5 et 4 indiquent la priorité de la DLSDU contenue dans la DLPDU. Son codage est:

- 00: Alarm (priorité la plus basse) ;
- 01: Normal (priorité normale) ;
- 10: Process data (données de processus);
- 11: Command (priorité la plus haute).

Le bit 3 Indique le type de la clé de réseau utilisée dans la DLPDU. Son codage est:

- 0: la clé bien connue de cette spécification, utilisée seulement par le dispositif non authentifié;
- 1: La clé-réseau confidentielle, utilisée par le dispositif authentifié.

Elle indique la clé utilisée pour authentifier la DLPDU. Pour toutes les communications au sein du réseau entre dispositifs authentifiés, la DLE doit mettre ce bit à '1' et utiliser la clé-réseau confidentielle pour générer la DLPDU MIC (voir 6.2.4.2). Ce bit doit être remis à '0' au cours du processus de rattachement alors que le dispositif est non authentifié. Les DLPDU transmises entre le dispositif non authentifié et ses voisins doivent utiliser la clé bien connue (voir 6.2.4.2.3).

Les bits 2, 1 et 0 du spécificateur de DLPDU indiquent le type de la DLPDU. Son codage est:

- 000: ACK;
- 001: Advertise (annonce);
- 010: Keep-alive (entretien);
- 011: Disconnect (déconnexion);
- 111: Data (données).

Les DLPDU ACK, Advertise, Keep-alive and Disconnect doivent utiliser la priorité (la plus élevée) Command.

Les autres codages de ce champ sont réservés pour usage futur.

6.2.1.2.6 Charge utile de la DLL (DLL payload)

La charge utile de la DLL dépend du type de DLPDU. Les DLPDU Data contiennent une DLSDU. Les autres types de DLPDU ont des données de maintenance de DLE qui dépendent du type de DLPDU. Par exemple, une DLPDU ACK a une charge utile qui est constituée d'un ajustement du temps tel que mesuré par le dispositif de destination spécifié dans la DLPDU immédiatement antérieure.

6.2.1.2.7 Code d'intégrité de message chiffré

Un code d'intégrité de message (MIC) chiffré est utilisé pour l'authentification de liaison de la DLPDU reçue. Les dispositifs récepteurs doivent répondre uniquement à une DLPDU de non-acquittement de monodiffusion qui a été authentifiée avec succès.

6.2.1.2.8 Contrôle de redondance cyclique

Le champ Contrôle de redondance cyclique (CRC) est basé sur un polynôme CRC de l'UIT-T à 16 bits, également appelé CRC16. Le CRC est calculé sur toute la trame en utilisant le polynôme générateur de degré 16 suivant:

$$G_{16}(x) = x^{16} + x^{12} + x^5 + 1.$$

Pour plus d'informations, voir l'IEEE 802.15.4-2006, 7.2.1.9. Il convient de calculer le CRC dans le matériel du dispositif.

NOTE Au niveau des récepteurs, le train d'octets reçus est soumis à un calcul similaire. Lorsqu'un récepteur peut prédire les octets qui contiendront un CRC, il peut omettre ces octets du calcul et vérifier que le CRC reçu concorde avec ces octets. En variante, le récepteur peut calculer un CRC sur tous les octets du message et vérifier que le résultat concorde avec une valeur résiduelle prévue de zéro.

Le CRC est utilisé pour détecter des erreurs sur les bits et les dispositifs doivent répondre uniquement à une DLPDU de non-acquittement avec un CRC concordant avec celui calculé par le dispositif récepteur ou dont la valeur résiduelle après traitement de la DLPDU complète, y compris les octets de CRC reçus, concorde avec la valeur résiduelle prévue de zéro.

6.2.2 Structure, codage et éléments de procédure spécifiques à la DLPDU

6.2.2.1 Généralités

Il existe cinq types de DLPDU.

DLPDU de type Data – contient la DLSDU en transit vers leur dispositif de destination finale. La couche réseau est la source et le puits pour les DLPDU de type Data.

DLPDU de type ACK – est la réponse de niveau de liaison apportée immédiatement à la réception de la DLPDU d'émission du dispositif source.

DLPDU de type Keep-alive – facilite la maintenance de la connexion entre dispositifs voisins.

DLPDU de type Advertise – fournit les informations aux dispositifs voisins qui essaient de rejoindre le réseau.

DLPDU de type Disconnect – elles sont utilisées pour signaler aux dispositifs voisins que le dispositif quitte le réseau.

Les dispositifs recevant une DLPDU d'un type autre que ceux énumérés ci-dessus doivent la rejeter et ne doivent pas acquitter la DLPDU reçue.

Les DLPDU ACK, Advertise, Keep-Alive et Disconnect sont générées et consommées par des DLE homologues. Ces DLPDU ne sont pas propagées vers la couche réseau ou à travers le réseau.

6.2.2.2 DLPDU Data

6.2.2.2.1 But d'une DLPDU Data

La DLPDU Data est utilisée pour envoyer des données d'utilisateur de DLS d'une DLE source vers une DLE de destination. Elle est également utilisée par la couche réseau pour transmettre ses données d'utilisateur d'une DLE intermédiaire vers une autre DLE.

6.2.2.2.2 Contenu de la charge utile d'une DLPDU Data

La charge utile de DLL dans cette DLPDU provient de la couche réseau du dispositif source de ce saut et elle est transmise à la couche réseau du dispositif de destination. Elle est ensuite transmise par la couche réseau homologue vers sa destination finale.

6.2.2.3 DLPDU ACK

6.2.2.3.1 But d'une DLPDU ACK

Elle est utilisée pour envoyer une réponse immédiate à la DLPDU non-ACK qui vient juste d'être reçue.

6.2.2.3.2 Contenu de la charge utile d'une DLPDU ACK

La structure de la charge utile de DLL pour une DLPDU ACK est montrée au Tableau 27.

Tableau 27 – Contenu de la charge utile d'une DLPDU ACK

Décalage par rapport au début de la charge utile	Longueur en octets	Description
0	1	Response-code
1	2	Ajustement de temps en μ s. La valeur de l'ajustement du temps est positive (négative) si la DLPDU a été reçue plus tôt (plus tard) que prévu.

Le code de réponse doit spécifier soit la réception réussie, soit l'erreur de réception de la DLPDU immédiatement antérieure comme montré au Tableau 28.

Tableau 28 – Code de réponse ACK

Code	Classe	Description
0	Success (succès)	La DLPDU a été reçue sans erreur et a été acceptée par le dispositif de destination.
61	Error (erreur)	Aucun tampon n'était disponible pour stocker la DLPDU ayant une priorité autre que 'Alarm'; La DLPDU a été rejetée.
62	Error	Aucun tampon n'était disponible pour stocker la DLPDU ayant une priorité 'Alarm'; La DLPDU a été rejetée.
63	Error	La priorité de la DLPDU reçue était inférieure à Priority_threshold, c'est-à-dire au seuil de priorité (voir 6.4.2); La DLPDU a été rejetée.

L'ajustement du temps est la différence de l'heure prévue de réception du délimiteur de début complet qui encadre une DLPDU non-ACK et l'heure de réception réelle du délimiteur de début complet, mesuré en μ s. L'ajustement du temps est un entier de 2 octets en complément en deux.

6.2.2.3.3 Contenu du MIC de l'ACK

Le champ MIC de la DLPDU ACK doit être calculé en utilisant la même clé que la clé dans la DLPDU juste reçue. Voir 6.2.4.2 pour la génération du MIC.

6.2.2.3.4 Envoi de la DLPDU ACK

La DLPDU ACK doit être envoyée sur la liaison lorsque la DLE d'envoi a juste reçu une DLPDU quelconque excepté une DLPDU ACK et il n'y a pas d'erreur de réception et l'Adresse de destination dans la DLPDU juste reçue n'est pas une adresse de diffusion. La DLE doit vérifier la priorité de la DLPDU reçue et la disponibilité du tampon ("buffer") de stockage local et soit accepter, soit rejeter la DLPDU reçue comme spécifié en 6.2.3.

6.2.2.3.5 Réception de la DLPDU ACK

La DLPDU reçue doit être traitée comme spécifié en 6.3.5.2.4.

6.2.2.4 DLPDU Keep-alive

6.2.2.4.1 But de la DLPDU Keep-alive

Les DLPDU Keep-alive sont utilisées:

- Pour la synchronisation du temps réseau. La synchronisation du temps est mise en jour en fonction de la valeur d'ajustement du temps retournée dans la DLPDU ACK;
- Pour confirmer la connectivité avec un voisin;
- Pour permettre au dispositif d'être détecté par d'autres.

6.2.2.4.2 Contenu de la charge utile d'une DLPDU Keep-alive

Ce champ doit être nul.

6.2.2.4.3 Envoi de la DLPDU Keep-alive

Cette DLPDU est envoyée comme spécifié en 6.3.3.4.

6.2.2.4.4 Réception de la DLPDU Keep-alive

La DLPDU reçue doit être utilisée pour mettre à jour le tableau de voisins comme spécifié en 6.3.3.4.

6.2.2.5 DLPDU Advertise

6.2.2.5.1 But d'une DLPDU Advertise

La DLPDU Advertise est utilisée pour inviter de nouveaux dispositifs dans le réseau.

6.2.2.5.2 Contenu des adresses de destination

L'adresse de destination doit être l'adresse de diffusion. L'adresse de diffusion doit avoir une longueur de cinq octets et doit avoir des zéros partout comme valeur.

6.2.2.5.3 Contenu de la charge utile d'une DLPDU Advertise

La charge utile de la DLL inclut des informations de réseau. La structure de la charge utile de DLL pour une DLPDU Advertise est montrée au Tableau 29.

Tableau 29 – Contenu de la charge utile d'une DLPDU Advertise

Décalage par rapport au début de la charge utile	Longueur en octets	Description
0	5	Absolute slot number – le nombre d'intervalles de temps depuis le début du réseau jusqu'à l'intervalle de temps utilisé pour l'émission de cette DLPDU.
5	1	Join-control
6	1	La taille d'une matrice de carte de voies en nombre de bits. La taille maximale de la matrice de carte de voies est de 256 bits
7	variable	Channel-map-array – il s'agit d'une matrice de bits commençant avec le bit de poids faible (bit 0 de l'octet 0) et ajoutant des octets autant que nécessaire jusqu'à ce que tous les bits soient comptabilisés. Chaque bit correspond à une voie. Si le bit est mis à '1', la voie correspondante est en utilisation.
n+1	2	Graph_ID
n+3	1	Norme de superframes

Le champ Join-control spécifie le niveau de sécurité et la join_priority (priorité de rattachement) comme montré ci-dessous:

Bits 7 à 4: Niveau de sécurité pris en charge par le réseau;

Bits 3 à 0: Join_priority – entier non signé indiquant la capacité du dispositif annonceur à prendre en charge un autre dispositif enfant. Plus la valeur est faible, le mieux ce dispositif annonceur se prête au rattachement.

Le contenu du Tableau 30, Tableau 31 et Tableau 32 est répété pour chaque superframe.

Tableau 30 – Début de la superframe

Décalage par rapport au début de la charge utile	Longueur en octets	Description
Variable	1	Superframe_ID
Variable	2	Taille de superframe, à savoir le nombre d'intervalles de temps dans cette superframe.

Tableau 31 – Le prochain champ après la taille de la dernière superframe

Décalage par rapport au début de la charge utile	Longueur en octets	Description
Variable	1	Nombre de liaisons

Tableau 32 – Répété pour chaque liaison

Décalage par rapport au début de la charge utile	Longueur en octets	Description
Variable	2	Join-slot – (c'est-à-dire intervalle de temps de rattachement) l'intervalle de temps spécifique au sein de la superframe pour cette liaison
Variable	1	Join-slot-channel-offset

Le champ Join-slot-channel-offset (décalage de voies d'intervalles de rattachement) spécifie:

- Bit 7: Réservé pour usage futur; la DLE d'envoi doit le mettre à '0';
- Bit 6: Si la liaison sert à l'émission de DLPDU par le dispositif de rattachement, la DLE d'envoi doit le mettre à '1';
- Bits 5 à 0: Channel offset – (décalage de voie) le décalage de voie de fréquence pour cet intervalle de temps. Cette valeur est utilisée pour calculer la fréquence de la liaison/le numéro de voie.

La taille de Channel-map-array dépend de la couche physique sous-jacente. La couche physique 2 450 MHz de l'IEEE 802.15.4-2006 fournit 16 voies. Le champ channel-map-array a une longueur de deux octets. L'octet de poids faible contient les voies 11 à 18 et l'octet de poids fort contient les voies 19 à 25 comme montré au Tableau 33. La voie 26 n'est pas utilisée et donc le bit 15 est toujours '0'.

Tableau 33 – Affectations des voies et de la fréquence

Indice	Numéro de voie
0	11
1	12
2	13
3	14
4	15
5	16
6	17
7	18
8	19
9	20
10	21
11	22
12	23
13	24
14	25
15	26

Le nombre d'intervalles de temps absolus et la carte des voies permettent d'identifier le décalage de voie actuel lors de l'émission d'un paquet pour demander au gestionnaire de réseau l'admission au réseau.

6.2.2.5.4 Contenu du MIC de l'Advertise

Le champ MIC de la DLPDU Advertise doit être calculé en utilisant la clé bien connue. Voir 6.2.4.2 pour la génération du MIC.

6.2.2.5.5 Envoi de la DLPDU Advertise

Un paquet Advertise peut être envoyé sur n'importe quelle liaison d'émission qui est disponible pour l'émission par le dispositif annonceur. Sur le dispositif de rattachement, lorsque la gestion de DL demande que la DLE rejoigne un réseau, la DLE est à l'écoute pour détecter les DLPDU Advertise envoyées par d'autres DLE. Elle utilise ensuite les informations contenues dans les DLPDU reçues pour se synchroniser au réseau et lancer le processus de rattachement en envoyant sa DLPDU Join Request (Demande de rattachement). Le dispositif de rattachement est limité à ces liaisons jusqu'à ce qu'il soit authentifié par le gestionnaire de réseau et ait reçu sa clé de réseau. En limitant la communication aux seules liaisons de

rattachement, le gestionnaire de réseau peut mettre le dispositif en quarantaine jusqu'à ce que le dispositif soit prêt à participer totalement au réseau.

Toutes les autres DLPDU ont une plus haute priorité que la DLPDU Advertise.

6.2.2.5.6 Réception de la DLPDU Advertise

La DLE qui n'a pas rejoint le réseau doit recevoir toutes les DLPDU Advertise. Elle doit apprendre les paramètres réseau inclus comme charge utile de DLL de la DLPDU Advertise. La réception de cette DLPDU doit générer la primitive "indication" d'Advertise de DL à l'attention de l'utilisateur de DLM.

6.2.2.6 DLPDU Disconnect

6.2.2.6.1 But d'une DLPDU Disconnect

La DLPDU Disconnect est utilisée par un dispositif qui quitte le réseau. Elle est envoyée à chaque voisin connu du dispositif.

6.2.2.6.2 Contenu des adresses de destination

L'adresse de destination doit être l'adresse du voisin.

6.2.2.6.3 Contenu de la charge utile d'une DLPDU Disconnect

Ce champ doit être vide, ce qui signifie qu'aucun octet ne doit être utilisé dans la charge utile.

6.2.2.6.4 Contenu du MIC de Disconnect

Le champ MIC de la DLPDU ACK doit être calculé en utilisant la clé de réseau. Voir 6.2.4.2 pour la génération du MIC.

6.2.2.6.5 Envoi de la DLPDU Disconnect

Lorsque la gestion de DL demande que la DLE quitte le réseau, elle doit envoyer la DLPDU Disconnect DLPDU à la première occasion.

6.2.2.6.6 Réception de la DLPDU Disconnect

La DLE de réception doit retirer la DLE d'envoi de sa liste de voisins et doit supprimer toutes les liaisons assurant la connexion à la DLE d'envoi. La réception de cette DLPDU doit générer la primitive "indication" de DISCONNECT DE DL à l'attention de l'utilisateur de DLM.

6.2.3 Priorité de DLPDU et régulation de débit

La priorité de la DLPDU est utilisée à des fins de régulation de débit, pour limiter l'encombrement du réseau et assurer que le gestionnaire de réseau garde le contrôle du réseau pendant un dysfonctionnement de processus ou lorsqu'un événement RF défavorable se produit. Par exemple, le gestionnaire de réseau peut élever le Priority_threshold pour diminuer le débit de paquets à travers le dispositif en changeant la valeur de Priority_threshold (voir 6.4.2).

La DLE est tenue d'avoir au moins un tampon de stockage réservé pour recevoir une DLPDU de priorité "Alarm" et au moins un tampon de stockage réservé pour recevoir une DLPDU de priorité "Command". En outre, ces tampons réservés ne doivent pas être utilisés pour stocker la DLPDU reçue ayant une autre priorité. La DLE de réception doit vérifier la priorité de la DLPDU reçue, le Priority_threshold courant et la disponibilité du tampon de stockage local pour déterminer si la DLPDU est acceptée ou rejetée comme suit.

- a) Si la destination finale de la DLPDU juste reçue n'est pas la DLE elle-même et le niveau de priorité de la DLPDU reçue est inférieur au `Priority_threshold`, la DLE doit rejeter la DLPDU reçue et le code de réponse dans la DLPDU ACK doit être mis à 63.

NOTE 1 Pour gérer le flux de DLPDU à travers le réseau, le gestionnaire de réseau peut élever ou baisser le niveau de priorité des DLPDU qu'une DLE est autorisée à accepter.

Sinon, si le niveau de priorité est "Alarm", alors

- i) si un et un seul tampon dédié au stockage de ce niveau de priorité n'est pas disponible, la DLE doit rejeter la DLPDU reçue et le code de réponse dans la DLPDU ACK doit être mis à 62;

NOTE 2 Un seul tampon pour la priorité "Alarm" est autorisé à réduire le flot d'alarmes. Les alarmes ne sont perdues, car elles sont datées et la PDU contenant les informations relatives à l'alarme est envoyée de nouveau après une expiration de délai.

- ii) Sinon, la DLE doit accepter la DLPDU reçue et le code de réponse dans la DLPDU ACK doit être mis à "Success".

Sinon, si le niveau de priorité n'est pas "Alarm" et

- i) si un tampon dédié au stockage du niveau de priorité de la DLPDU reçue n'est pas disponible, la DLE doit rejeter la DLPDU reçue et le code de réponse dans la DLPDU ACK doit être mis à 61;
- ii) Sinon, la DLE doit accepter la DLPDU reçue et le code de réponse dans la DLPDU ACK doit être mis à "Success".
- iii) Si la DLPDU reçue est du type Keep-alive ou Advertise ou Disconnect, la DLE doit toujours l'accepter et le code de réponse dans la DLPDU ACK doit être mis à "Success".

NOTE 3 si un tampon de stockage est disponible, la DLE accepte toujours la DLPDU de priorité "Command" car elle a la priorité la plus élevée. Il convient que la DLE alloue suffisamment de tampons de manière à pouvoir recevoir pratiquement toutes les DLPDU de priorité "Command" reçues.

En résumé, les DLPDU de gestion de réseau se propagent toujours à travers le réseau, permettant au gestionnaire de réseau de maintenir le réseau opérationnel. Le flux d'alarmes à travers le réseau est limité, ce qui assure que des flots d'alarmes ne perturbent pas le fonctionnement du réseau. Les alarmes étant toujours datées, aucune information relative aux séquences de défaillances n'est perdue.

Tous les autres trafics du réseau s'écoulent à travers le réseau dans la mesure permise par l'espace tampon et la capacité du réseau. Au sein du trafic réseau, les données de processus ont la priorité la plus élevée. Le fonctionnement et le contrôle du processus ont une priorité plus faible pour prévenir l'interruption des communications du réseau.

6.2.4 Détection d'erreurs et sécurité

6.2.4.1 Généralités

Pour réaliser la détection d'erreurs, la présente norme inclut une séquence de contrôle des trames à la fin de chaque DLPDU. Le contrôle (CRC) est utilisé pour détecter les erreurs des communications. Le CRC est calculé sur toute la DLPDU en utilisant l'algorithme de l'UIT-T de 16 bits.

Pour la sécurité, chaque DLPDU inclut un MIC dans chaque DLPDU.

6.2.4.2 Calcul de MIC

6.2.4.2.1 Généralités

Un MIC chiffré est utilisé pour garantir que la DLPDU émane d'un dispositif authentifié et approuvé. La DLPDU elle-même n'est pas chiffrée; en revanche, son contenu est authentifié à l'aide du MIC de quatre octets. Le MIC est généré et confirmé en utilisant le mode CCM* (compteur combiné avec CBC-MAC, cipher block chaining-message authentication code (corrigé)) en association avec le chiffre à bloc de 128 bits de l'AES (advanced encryption

standard) pour fournir l'authentification comme spécifié dans la Publication 800-38C du NIST (National Institute of Standards and Technology). Ce chiffre exige quatre chaînes d'octets comme paramètres:

- 'a', les données complémentaires devant être authentifiées mais pas chiffrées;
- 'm', le message devant être chiffré;
- 'N', le nonce de 13 octets; et
- 'K', la clé AES de 128 bits.

Les parties d'une DLPDU, de l'octet 0x41 jusqu'à la fin de la charge utile, correspondent à la chaîne d'octets 'a'. La DLPDU n'étant pas chiffrée, la chaîne d'octets 'm' est vide.

6.2.4.2.2 Nonce de la DLL

Le 'N' doit avoir une longueur de 13 octets exactement et il est la concaténation de l'ASN et de l'adresse source dans la DLPDU.

L'ASN contient toujours le numéro de l'intervalle de temps courant. L'ASN, d'une longueur de 5 octets, est copié dans les octets N[0] à N[4], où N[0] est l'octet de poids fort MSB de l'ASN.

Si la DLPDU a l'adresse EUI-64, elle est copiée (MSB à LSB) dans les octets N[5] à N[12]. En d'autres termes, l'OUI est copié dans les octets N[5] à N[7]. Le code de type de dispositif étendu et l'ID de dispositif sont respectivement copiés (MSB à LSB) dans les octets N[8] à N[9] et N[10] à N[12].

Si la DLPDU a le surnom de 2 octets, elle est copiée (MSB à LSB) dans les octets N[11] à N[12]. Les octets N[5] à N[10] sont mis à 0x00.

6.2.4.2.3 Clé de DLL

La clé doit avoir une longueur de 128 bits (16 octets) comme exigé par le Mode CCM. Elle est copiée dans la chaîne d'octets 'K' dans l'octet de poids fort en premier. Les octets K[0] et K[15] sont respectivement l'octet de poids fort et l'octet de poids faible de la clé.

Il y a deux clés de DLL:

- la clé bien connue, qui est utilisée dans la DLPDU Advertise, et
- la clé de réseau, qui est utilisée pour toutes les autres DLPDU.

La clé bien connue doit être identique pour tous les dispositifs et a une valeur hexadécimale de 7777 772E 6861 7274 636F 6D6D 2E6F 7267. Elle est utilisée pour les DLPDU transmises entre le dispositif de rattachement et les dispositifs faisant déjà partie du réseau.

NOTE 1 La clé bien connue est choisie au hasard.

Cette clé de réseau est fournie par le gestionnaire de réseau à dispositif, lorsque celui-ci rejoint le réseau. Le dispositif ne doit pas changer la valeur de cette clé.

NOTE 2 Le gestionnaire de réseau peut changer la clé de réseau à tout moment.

6.2.4.3 Détection d'erreurs

Lorsque la DLE reçoit une DLPDU, elle vérifie le CRC dans la DLPDU reçue. Si le CRC dans la DLPDU ne concorde pas avec le CRC attendu, la DLE doit rejeter la DLPDU reçue et ne doit pas envoyer de DLPDU ACK en réponse.

Si le CRC dans la DLPDU reçue est correcte, la DLE doit calculer le MIC attendu. Elle doit ensuite le comparer au MIC dans la DLPDU reçue. Si ces deux ne sont pas identiques, la DLE doit rejeter la DLPDU reçue et ne doit pas envoyer de DLPDU ACK en réponse.

6.3 Contrôle d'accès au support

6.3.1 Fonction

La sous-couche Contrôle d'accès au support (MAC) fournit la communication de base d'une DLE à une autre. Le moment d'une telle communication est régulé par la synchronisation des intervalles de temps à travers le réseau. Pour ce faire, la DLE comprend:

- Le tableau de voisins, des supertrames, des liaisons et des graphes qui configurent la communication entre le dispositif et ses voisins (voir 6.3.3.). Ces tableaux sont préparés par le gestionnaire de réseau. De plus, le tableau de voisins est mis à jour par la DLE, lorsqu'il découvre l'ajout d'un nouveau voisin au réseau ou la déconnexion d'un voisin existant du réseau.
- Un programmeur de liaison (voir 6.3.4) qui évalue les tableaux de dispositifs et choisit le prochain intervalle de temps devant être pris en charge en étant à l'écoute pour détecter une DLPDU ou en envoyant une DLPDU. En général, le programmeur de liaison parcourt les tableaux pour identifier le prochain intervalle de temps dans lequel envoyer une DLPDU et le prochain intervalle de temps dans lequel être à l'écoute. L'intervalle de temps programmé est le suivant de ces deux intervalles de temps qui se produit.
- Des diagrammes d'états qui commandent la propagation des DLPDU à travers la sous-couche MAC. Le fonctionnement de MAC (voir 6.3.5) consiste en des intervalles de temps d'entretien et de maintenance programmée. Le fonctionnement de MAC est guidé par des événements et répond à des invocations de primitive de prise en charge et le début d'un intervalle de temps ayant besoin de prise en charge.

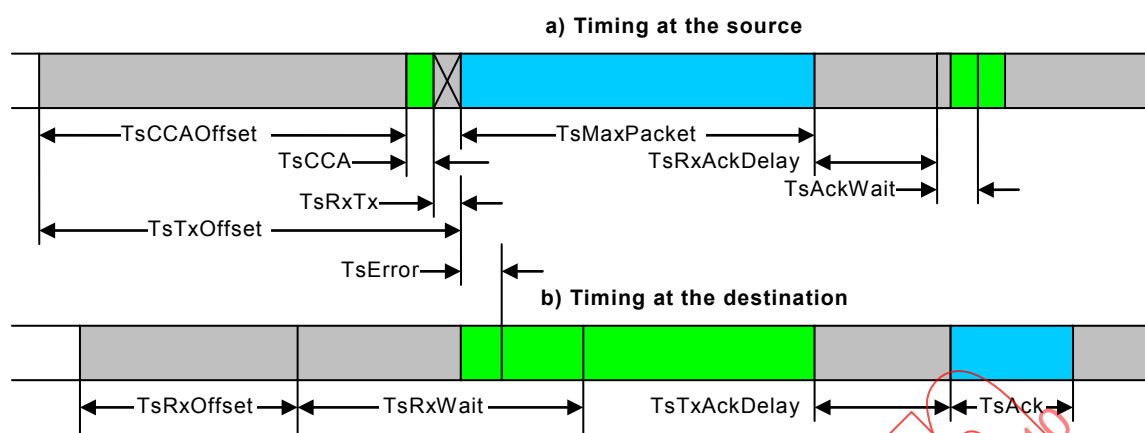
La sous-couche MAC envoie la DLPDU placée en file d'attente dans les tampons du dispositif et elle reçoit la DLPDU envoyée par une DLE voisine. Ces deux opérations sont exécutées un intervalle de temps à la fois. La réussite de la communication dépend de la synchronisation de l'heure des intervalles de temps entre voisins et du minutage conforme de la transaction dans l'intervalle de temps.

6.3.2 Temporisations d'intervalle de temps

6.3.2.1 Généralités

La Figure 19 montre un intervalle de temps et fournit une vue générale des minutages d'une transaction simple. Le diagramme (a) montre le fonctionnement au niveau du voisin source et le diagramme (b) montre le fonctionnement au niveau du voisin de destination. Les détails des paramètres d'heure d'un intervalle de temps sont définis au Tableau 34; Y sont également montrées les valeurs requises pour la couche physique prises en charge par la présente norme et spécifiées dans la couche physique 2 450 MHz de l'IEEE 802.15.4-2006. Toutes les transactions sont tenues de se conformer à ces paramètres temporels.

L'heure de début d'intervalle au niveau de la source et celle au niveau de la destination peuvent différer en raison de la grandeur différente de la dérive d'horloge. Ce diagramme montre que l'intervalle de temps débute à un instant plus tardif au niveau du dispositif de destination.



Légende

Anglais	Français
a) Timing at the source	a) Temporisation à la source
b) Timing at the destination	b) Temporisation à la destination

Figure 19 – Détails de temporisation d'un intervalle de temps

Tableau 34 – Définitions et valeurs de temporisation d'un intervalle de temps

Symbole	Définition	Valeur requise
TsTxOffset	Du début de l'intervalle jusqu'au début de l'émission de préambule	2 120 $\mu\text{s} \pm 100 \mu\text{s}$
TsRxOffset	Du début de l'intervalle de temps jusqu'à l'instant où l'émetteur-récepteur doit être à l'écoute	1 120 $\mu\text{s} \pm 100 \mu\text{s}$
TsRxWait	Temps minimum à attendre le début du message; il est corrélé à la grandeur de la dérive entre les voisins qui peut être tolérée pour maintenir les communications	2 200 $\mu\text{s} \pm 100 \mu\text{s}$
TsMaxPacket	Durée que peut prendre l'émission du paquet le plus long possible qui inclut le préambule, le délimiteur, la longueur et la DLPDU de PhL	4 256 μs
TsTxAckDelay	De la fin du message jusqu'au début de l'ACK; le dispositif de destination doit valider le STX et générer un ACK, si requis, au cours de cet intervalle	1 000 $\mu\text{s} \pm 100 \mu\text{s}$
TsRxAckDelay	De la fin de PhPDU d'émission jusqu'à l'instant où l'émetteur-récepteur doit être à l'écoute pour détecter l'ACK	800 $\mu\text{s} \pm 100 \mu\text{s}$
TsAckWait	Temps minimum à attendre le début d'un ACK;	400 $\mu\text{s} \pm 100 \mu\text{s}$
TsAck	Temps pour émettre un ACK (26 octets)	832 μs
TsCCAOOffset	Du début d'intervalle de temps jusqu'au début de CCA	1 800 $\mu\text{s} \pm 100 \mu\text{s}$
TsCCA	Temps pour effectuer la CCA (8 symboles)	128 μs
TsRxTx	Durée la plus longue pour commuter de la réception à l'émission ou vice versa (12 symboles)	192 μs
TsError	Il s'agit de la différence entre l'heure de début réel du message et l'heure de début idéal du message telle que perçue par le dispositif de réception. En d'autres termes, il s'agit de quantité dont le dispositif de réception perçoit que le dispositif émetteur est désynchronisé	

Si un dispositif a un paquet prêt pour l'émission et il est programmé pour l'émission dans un intervalle de temps, l'intervalle de temps en question commence en accordant une durée pour préparer la PhPDU en vue de l'émission. La source doit effectuer l'évaluation de voie libre (CCA) et émettre le paquet. Si la DLPDU reçue exige un acquittement, la DLE de destination doit émettre la PhPDU contenant la DLPDU ACK.

Si un dispositif est programmé pour la réception dans un intervalle de temps, le dispositif en question doit entrer dans le mode réception. Le dispositif doit commencer à écouter le support après un retard de $TsRxOffset$ par rapport au début de l'intervalle de temps. La fenêtre de réception spécifiée par $TsRxWait$ permet la dérive de l'heure du dispositif, tout en permettant encore aux dispositifs de communiquer et de resynchroniser leurs temporisateurs d'intervalles de temps.

Si le dispositif de destination détecte une PhpDU, il doit saisir la fin de la réception de l'heure de début du délimiteur de couche physique. Il doit calculer $TsError$ comme étant la différence entre l'heure prévue de début et l'heure réelle de début du délimiteur de couche physique.

Du temps est alloué vers la fin de l'intervalle de temps pour traiter le paquet reçu et se préparer pour le prochain intervalle de temps, (par exemple évaluer et classer par ordre de priorité les paquets mis maintenant en file d'attente dans chaque dispositif). Si l'un des voisins était la source de temps pour l'autre, l'heure de fin de l'intervalle de temps doit être alignée après une communication réussie.

6.3.2.2 Émission d'un paquet

Le dispositif source doit attendre pendant une durée $TsCCAOffset$ après le début d'un intervalle de temps et ensuite il doit effectuer une CCA pendant la durée $TsCCA$. Si la voie est occupée, la tentative de transaction est reprogrammée pour un intervalle de temps ultérieur. Sinon, si la voie est libre, le dispositif doit faire commuter l'émetteur-récepteur de la réception à l'émission. Ensuite, le dispositif doit commencer son émission de telle manière que le début de message (SOM) ait lieu exactement à une durée $TsTxOffset$ par rapport au début de l'intervalle.

6.3.2.3 Réception d'un paquet

Le dispositif de destination doit entrer dans le mode réception au début de l'intervalle de temps. Il doit commencer à écouter le support après un retard de $TsRxOffset$ par rapport au début de l'intervalle de temps. Le dispositif de destination doit continuer à rechercher le SOM. Si le SOM n'est pas détecté au bout de la durée ($TsRxOffset + TsRxWait$) à partir du début de l'intervalle de temps, il doit cesser d'être à l'écoute. Sinon, si le SOM est détecté, il doit recevoir et valider le paquet reçu.

6.3.2.4 Transaction acquittée

La plupart des communications consistent en ce que le dispositif source propage un message en émettant un paquet et le dispositif de destination acquitte la réception du paquet en question. Pour une communication acquittée, les adresses de la source et de la destination dans la DLPDU doivent contenir l'adresse de dispositif individuel.

Si le paquet reçu est valide, le dispositif de destination doit examiner l'adresse de destination dans la DLPDU. Si l'adresse de destination n'est pas une adresse de diffusion et il n'y a pas d'erreur de réception, le dispositif doit commuter de la réception à l'émission et doit émettre la PhpDU contenant la DLPDU ACK et, de ce fait, le SOM de cette PhpDU se produit exactement $TsTxAckDelay$ après la fin du paquet émis du dispositif source.

Entre-temps, le dispositif source doit commuter de l'émission vers la réception. Après une durée $TsRxAckDelay$ à partir de la fin de l'émission, le dispositif source doit écouter le support. Il doit rechercher le SOM pendant la durée $TsAckWait$.

6.3.2.5 Transaction de diffusion

Les émissions diffusées sont également prises en charge. L'adresse source de la DLPDU est une adresse de dispositif individuelle et l'adresse de destination est une adresse de diffusion. Les messages de diffusion ne sont pas acquittés par le dispositif de réception. L'adresse de diffusion doit avoir une longueur de cinq octets et doit avoir des zéros partout comme valeur.

Si le paquet reçu est valide, le dispositif de destination doit examiner l'adresse de destination dans la DLPDU. Si l'adresse de destination est une adresse de diffusion, le dispositif de réception ne doit pas émettre de réponse.

6.3.3 Tableaux et tampons de communication

6.3.3.1 Généralités

Tous les dispositifs doivent maintenir une série de tableaux qui commandent les communications réalisées par le dispositif et recueillir les statistiques relatives à ces communications. De plus, le dispositif doit placer en tampon les paquets reçus, les traiter et les transmettre, le cas échéant.

Les tableaux commandant les activités de communication sont:

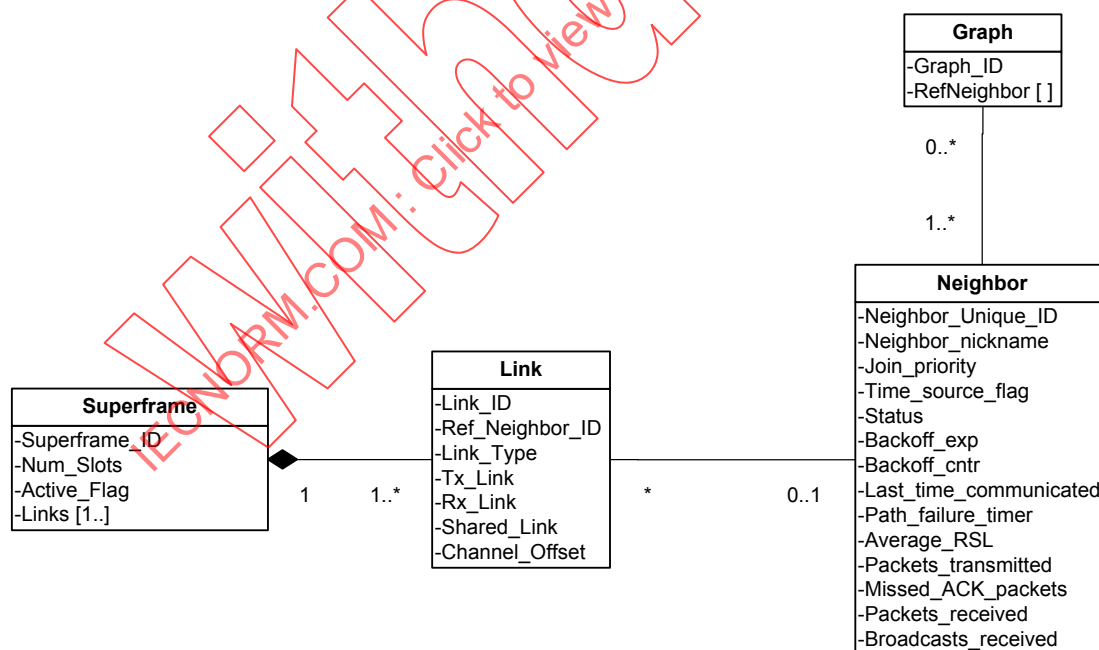
Tableau Superframe (Tableau de superframe) – il stocke les attributs d'une superframe;

Tableau Link (Tableau de liaison) – ce type de tableaux est utilisé pour stocker les attributs d'une liaison;

Tableau Neighbor (Tableau de voisins) – il stocke la liste de tous les dispositifs avec lesquels le dispositif peut communiquer directement;

Tableau Graph (Tableau de graphe) – il stocke les destinations des prochains sauts légitimes pour propager le paquet vers sa destination.

Certaines des informations dans ces tableaux sont partagées entre la NLE et la DLE. La relation entre ces tableaux est montrée à la Figure 20. Au sein du dispositif, le tableau de voisins est central. Ce tableau contient une liste de tous les dispositifs qui ont été identifiés par le dispositif.



Légende

Anglais	Français
Superframe	Superframe
Link	Liaison
Graph	Graphe
Neighbor	Voisin

Figure 20 – Tableaux de DLE et relation

Une supertrame contient une ou plusieurs liaisons. Une liaison est associée à 0 (liaison de diffusion) ou 1 voisin (dispositif de réseau spécifique). Un voisin peut être associé à plusieurs liaisons. Les voisins peuvent aussi être associés avec 0 ou plusieurs graphes et les graphes peuvent être associés à 1 ou plusieurs voisins.

En plus de ces tableaux, il y a des tampons pour stocker les paquets. Le Tableau 35 montre le nombre minimum d'entrées de tableau et l'espace tampon minimum que chaque dispositif de réseau doit prendre en charge.

Tableau 35 – Exigence en minimum de Tableaux de DLE et d'espace tampon

Nom du tableau	Nombre minimum d'entrées requis
Superframe (supertrame)	16
Link (liaison)	64
Neighbor (voisin)	32
Graph (graphe)	32
Nombre total de connexions Graph-Neighbor	128
Tampons de paquets	16

NOTE Certains des attributs de ces tableaux décrits dans la présente norme peuvent être calculés ou dérivés d'autres d'informations et il n'est pas nécessaire de les stocker séparément. Bien que la mise en œuvre spécifique du stockage de données et de configuration soit laissée aux concepteurs du dispositif, une description générale des champs contenus dans les structures de données est importante pour le fonctionnement global.

Chaque supertrame a une ou plusieurs liaisons. Les liaisons spécifient l'intervalle de temps et les informations associées indispensables pour transmettre ou accepter un paquet.

Chaque liaison définit une occasion de communications. Une liaison peut appartenir à une et une seule supertrame. Les liaisons spécifient un voisin qui est le partenaire dans toute communication utilisant la liaison. Une liaison est orientée vers un voisin ou bien elle est une liaison de diffusion. Pour une liaison, les paquets sont propagés soit vers le voisin, soit à partir de celui-ci. Les liaisons de diffusion propagent toujours les paquets à partir du dispositif.

Un graphe peut spécifier plus d'un voisin, dont n'importe lequel peut être utilisé pour le prochain saut pour les paquets empruntant le chemin désigné par le Graph_ID. Pour transmettre un paquet en utilisant l'acheminement de graphes, le dispositif peut le propager vers n'importe lesquels des voisins associés au Graph_ID du paquet en question. Pour plus d'informations, voir 6.3.3.5

6.3.3.2 Tableau Superframe

Les attributs de supertrame sont montrés au Tableau 36. Une supertrame consiste en un nombre fixe d'intervalles de temps, ne contient initialement pas de liaisons et commence comme étant inactive. Les supertrames sont créées et maintenues par le gestionnaire de réseau. Une fois qu'une supertrame est créée, le gestionnaire de réseau peut ajouter, supprimer ou modifier des liaisons au sein de la supertrame, identifiant ainsi les occasions pour les communications entre dispositifs. Une fois configurée avec des liaisons, la supertrame peut être activée pour permettre au programmeur de liaison de commencer à identifier les intervalles de temps d'émission et de réception.

Tableau 36 – Attributs de supertrame

Nom d'attribut	Plage de valeurs	Définition
Superframe_ID	0 à 255	Identificateur propre à la supertrame, fourni par le gestionnaire de réseau.
Num_Slots	0 à 65 535	Nombre d'intervalles de temps dans cette supertrame, c'est-à-dire taille de la supertrame.
Active_Flag	TRUE, FALSE	Fanion indiquant si la supertrame est actuellement activée.
Links [1..]	Link_ID	Liste des liaisons dans cette supertrame.

6.3.3.3 Tableau Link

6.3.3.3.1 Généralités

Les attributs d'une liaison sont montrés au Tableau 37. La liaison inclut une référence à un voisin qui est autorisé à communiquer avec le dispositif. Cette référence peut être faite à un seul voisin ou il peut s'agir d'un groupe non spécifié de voisins (pour une liaison de diffusion). Le nombre d'intervalles de temps dans la supertrame, le sens de la communication, les caractéristiques de la liaison (partagée ou dédiée) et la voie de communication initiale sont spécifiés. La liaison est identifiée de façon univoque par la combinaison des paramètres-clés suivants:

Superframe_ID, nombre d'intervalles de temps et adresse de voisin.

Tableau 37 – Attributs d'une liaison

Nom d'attribut	Plage de valeurs	Définition
Link_ID	Voir Note	Identificateur de la liaison
Ref_Neighbor_ID	Voir Note	Référence à une entrée du tableau de voisins
Link_type	NORMAL, BROADCAST, JOIN, DISCOVERY	Indique le type de liaison.
Tx_Link	TRUE, FALSE	Si TRUE, indique que la liaison doit être utilisée pour l'émission.
Rx_Link	TRUE, FALSE	Si TRUE, indique que la liaison doit être utilisée pour la réception.
Shared_Link	TRUE, FALSE	Si TRUE, indique que la liaison est partagée par plusieurs dispositifs
Slot	0 à 65 535	Nombre d'intervalles de temps dans la supertrame attribué à cette liaison
Channel_Offset	11 – 25	Décalage initial de la voie de saut de fréquence (voir Tableau 33)
NOTE La plage de valeurs n'est pas spécifiée par la présente norme; Elle dépend de la mise en œuvre.		

NOTE Le gestionnaire de réseau ne supprime ni ne suspend de liaison du type JOIN, pendant qu'il y a une demande "join request" (demande de rattachement) en cours.

Les liaisons sont allouées à une supertrame par le gestionnaire de réseau. Lorsque le gestionnaire de réseau désigne une liaison comme étant partagée, l'accès à multiples expéditeurs à base de conflit est accompli dans la liaison correspondante. Pour ces liaisons, les messages qui ne sont pas acquittés conduisent à l'application d'un algorithme de back-off (algorithme de repli) aléatoire (voir 6.3.4.4).

6.3.3.3.2 Calcul de voie de liaison

La liaison spécifie également le Channel_Offset et, donc, implicitement l'ordre dans lequel les voies sont sautées. Le reste de cette opération indexe le tableau de voies pour obtenir la voie réelle utilisée pour la communication dans l'intervalle de temps actif. Le contenu et la longueur du tableau de voies dépendent de la couche physique sous-jacente. Dans la couche physique préférentielle spécifiée dans l'IEEE 802.15.4-2006 (2 450 MHz), le nombre de voies disponibles est de 16.

NOTE La présente norme n'utilise que les 15 premières sur les 16 voies disponibles.

La DLE doit contenir une variable ChannelMap de 64 bits qui suit les voies actives du dispositif. Chaque bit (bit 0 à bit 63) qui est TRUE identifie une voie active. Le ChannelMap est initialisé à TRUE partout (à savoir, toutes les voies sont actives par défaut). Seuls les bits correspondant à des voies disponibles pour la couche physique sont significatifs et considérés dans les calculs éventuels.

La DLE doit avoir une variable ActiveChannelArray (Matrice des voies actives) pour stocker le jeu ordonné de numéros de bits actifs dans ChannelMap. Le numéro de bit de la voie active la moins significative doit être stocké à l'indice zéro de la matrice. Le numéro de bit correspond à l'indice dans le tableau de voies physiques pour la couche physique utilisée.

La DLE doit calculer la valeur de la variable ActiveChannel en divisant la somme du Channel_Offset et du nombre d'intervalles de temps absolus par le nombre de voies actuellement actives.

$$\text{ActiveChannel} = (\text{Channel\_Offset} + \text{nombre d'intervalles de temps absolus}) \% \text{nombre de voies actives}.$$

Elle doit utiliser la valeur ActiveChannel comme indice dans l'ActiveChannelArray. La valeur stockée à cet indice doit être utilisée comme étant la voie à utiliser pour la liaison dans le nombre d'intervalles de temps absolus en question.

$$\text{Voie} = \text{ActiveChannelArray}[\text{ActiveChannel}].$$

6.3.3.4 Tableau Neighbor

Le dispositif doit maintenir une liste des voisins dont il a connaissance. Ce sont les dispositifs connexes qui partagent une liaison avec ce dispositif ou des voisins dont les communications ont fait l'objet d'écoute indiscrete.

Comme montré à la Figure 20, le tableau de voisins est central au pilotage des communications du dispositif:

- Chaque liaison a une référence à un voisin ou bien elle est une liaison de diffusion.
- Les graphes peuvent avoir des références à plusieurs voisins. Lorsqu'un graphe est utilisé pour l'acheminement, les voisins pour le graphe en question sont tous des récipiendaires valides du paquet propagé.

Le tableau de voisins recueille une diversité d'attributs et de statistiques concernant le voisin comme montré au Tableau 38, y compris:

- les informations de base sur l'identité du voisin;
- la performance et les statistiques antérieures; et
- les attributs d'intervalles de temps partagés.

Les informations de base concernant l'identité comprennent l'ID propre au voisin, l'adresse de surnom de 2 octets et si le dispositif est une source de temps ou non. Pour prendre en charge

l'accès à base de conflit, le tableau de voisins contient également les paramètres nécessaires pour prendre en charge l'algorithme de repli (voir 6.3.4.4).

Tableau 38 – Attributs de voisin et variables

Nom d'attribut	Plage de valeurs	Définition
Neighbor_Unique_ID	voir 6.2.1.2.4	ID propre au dispositif voisin (à savoir, l'adresse longue)
Neighbor_nickname	voir 6.2.1.2.4	L'adresse courte du voisin est celle définie en 3.3.67
Join_priority	0 à 15	Join_priority (Priorité de rattachement) à laquelle le voisin s'annonce
Time_source_flag	TRUE, FALSE	Flag indiquant s'il convient que le dispositif prenne la synchronisation du temps de son voisin
Status	Bit Field (Champ de bits)	Informations sur le statut relatives à ce voisin (par exemple, une défaillance de trajet) comme spécifié au Tableau F.34
Backoff_exp	0 à 7	Exposant de repli dans l'algorithme d'évitement de collision dans le cas des liaisons partagées ^b
Backoff_cntr	0 à 127	Compteur de repli dans l'algorithme d'évitement de collision dans le cas des liaisons partagées ^b
Last_time_communicated	0 à $2^{32}-1$ en unités de ms	Temps de la dernière communication avec ce voisin
Path_failure_timer	0 à $2^{32}-1$	Temporisateur cyclique en ms pour détecter la défaillance du trajet menant au voisin
Average_RSL	-128 à 127	Niveau moyen de signal reçu en dBm pour les paquets reçus provenant de ce voisin
Packets_transmitted	0 à 65 535 ^a	Nombre de paquets de non-diffusion émis vers ce voisin
Missed_ACK_packets	0 à 65 535 ^a	Nombre de paquets pour lesquels un ACK escompté n'a pas été reçu de ce voisin
Packets_received	0 à 65 535 ^a	Nombre de paquets de non-diffusion corrects reçus provenant de ce voisin
Broadcasts_received	0 à 65 535 ^a	Nombre de paquets de diffusion corrects reçus provenant de ce voisin
^a La valeur des compteurs de paquets repasse de 65 535 à 0.		
^b Voir 6.3.4.4 pour l'algorithme.		

La capacité du dispositif à communiquer avec un voisin est une métrique clé pour former et nettoyer le réseau maillé. Les statistiques sont maintenues pour chaque voisin. Elles incluent Average_RSL; les statistiques sur les paquets émis et reçus ainsi que le marqueur temporel de la dernière communication avec le voisin. Pour les voisins liés, Average_RSL se calcule en utilisant un filtre avec l'équation suivant:

$$\text{Average\_RSL} = \text{Average\_RSL} - (\text{Average\_RSL} / \text{RSL\_damp}) + (\text{MeasuredRSL} / \text{RSL\_damp})$$

où

MeasuredRSL est le niveau RSL pour le paquet courant;
RSL_damp est le facteur d'amortissement.

RSL_damp doit être une puissance de 2 avec une valeur minimale de 2 et une valeur maximale de 128. Sa valeur par défaut doit être 64. Si le dispositif ne reçoit aucune DLPDU Data d'un voisin, ce filtre ne doit pas être utilisé pour le voisin en question. Au contraire, la valeur de RSL la plus élevée parmi toutes les DLPDU reçues de la part de ce voisin au cours de sa découverte doit être utilisée.

Si une liaison à un voisin existe, le Last_time_communicated est utilisé pour déclencher l'émission d'une DLPDU Keep-alive. Une DLPDU Keep-alive doit être émise vers ce voisin

chaque fois que la valeur du temps `Last_time_communicated` plus le `Keep_alive_time` (voir Tableau 46) est supérieure au temps courant. La DLPDU Keep-alive doit être répétée jusqu'à réception d'une nouvelle DLPDU provenant du voisin en question.

Le `Path_failure_timer` doit aussi être maintenu pour chaque voisin. Chaque fois qu'une DLPDU est reçue du voisin, le temporisateur doit être initialisé à `Path_fail_time` (voir Tableau 46). Lorsque ce temporisateur atteint zéro, l'indication d'événement DLM avec l'événement `PATH_FAILURE` doit être invoquée. Lorsque cela se produit, le temporisateur doit être réinitialisé à `Path_fail_time` et être redémarré. Le dispositif doit continuer de tenter d'utiliser le trajet défaillant jusqu'à ce que le voisin ou ses liaisons soi(en)t supprimé(es) du tableau.

Chaque paquet reçu est utilisé pour mettre à jour les attributs de voisin correspondants ou il crée un nouveau tableau de voisins. Si un dispositif partage des liaisons avec un voisin, le voisin en question doit être maintenu dans le tableau. Les voisins sans liaisons à ce dispositif doivent être supprimés. Lorsque le tableau de voisins est plein et un nouveau voisin est détecté, le voisin ayant la valeur de `Last_time_communicated` la plus ancienne doit être supprimé et le nouveau voisin doit être ajouté au tableau.

6.3.3.5 Tableau Graph

Le graphe fournit les informations relatives à l'acheminement pour guider la livraison d'un paquet jusqu'à sa destination finale. Un graphe est une liste orientée de trajets qui relient deux dispositifs dans le réseau. Les graphes utilisés sont tant amont (à destination de la passerelle) qu'aval. Le gestionnaire de réseau est chargé de configurer correctement chaque graphe. Le Tableau 39 montre les attributs d'un graphe. Les graphes ont un identificateur; une liste de voisins ainsi qu'une adresse longue et une adresse courte. Lorsque la valeur de `Graph_ID` est inférieure à 0x0100, cela indique un `Superframe_ID`, et lorsqu'elle est égale à 0xFFFF, cela indique que le graphe n'est pas valide.

Tableau 39 – Attributs d'un graphe

Nom d'attribut	Plage de valeurs	Définition
<code>Graph_ID</code>	256 à 65 534	Identificateur propre au graphe tel que défini en 3.3.44
<code>RefNeighbor []</code>	<code>Neighbor_Unique_ID</code> ou <code>Neighbor_nickname</code>	Liste des références aux voisins qui se situent au prochain en direction de la destination.

Au niveau du dispositif source d'origine pour un paquet, les couches supérieures identifient la destination finale du paquet et le graphe devant être utilisé pour acheminer le paquet.

NOTE Le dispositif utilise les informations relatives aux chemins pour choisir le `Graph_ID`.

`RefNeighbor` identifie les dispositifs qui sont des destinations de liaisons de données admissibles pour le prochain saut du paquet vers sa destination finale. Sachant qu'un graphe a la possibilité de spécifier plusieurs prochains sauts, le routage du graphe intègre la redondance et la fiabilité. Les références individuelle de voisins sont également appelées "connexion".

Les adresses de destination sont exigées par tous les dispositifs alimentant des données à une destination finale spécifique. Cependant, les adresses sont facultatives car des dispositifs intermédiaires peuvent tout simplement transmettre le paquet sur son chemin et ne pas alimenter de données à la même adresse finale.

6.3.3.6 Liste de tampons de paquets

6.3.3.6.1 Généralités

Tous les dispositifs doivent maintenir une liste de tampons de paquets. Ces tampons sont utilisés pour recevoir, traiter et émettre des paquets. L'enregistrement associé à un paquet est montré au Tableau 40. Le Packet_ID est utilisé pour référencer le paquet et il est créé lorsque le paquet est ajouté via la primitive du service "request" DL-TRANSMIT (demande d'émission de DL).

Tableau 40 – Enregistrement de paquets

Élément d'enregistrement	Définition
Packet_ID	Identificateur propre au paquet
Payload	La charge utile de la DLL, c'est-à-dire la NPDU
Priority	Priorité d'émission du paquet
Destination	Graphe, informations de routage de source ou de proxy ou destination de diffusion
Packet_time_stamp	L'heure ASN à laquelle un paquet a été ajouté à la liste d'émission

Le Packet_time_stamp doit être fixé lorsque le paquet est ajouté à la liste d'émission. Ce marqueur temporel doit être utilisé pour sélectionner le paquet à émettre lorsque l'un ou l'autre de deux paquets d'égale priorité peut être propagé dans le même intervalle de temps absolu. Dans ce cas, le paquet le plus ancien doit être envoyé le premier.

L'enregistrement contient la priorité du paquet et la spécification de la destination du paquet.

6.3.3.6.2 Priorités des paquets

Sachant que le dispositif doit pouvoir stocker plusieurs paquets, en attendant leur propagation, il est possible que plusieurs paquets soient candidats à l'émission dans le même intervalle de temps. Lorsque cela se produit, la priorité et l'âge du paquet doivent être utilisés pour choisir le paquet à émettre dans l'intervalle de temps en question. Lorsqu'il y a plusieurs paquets pouvant être émis dans l'intervalle de temps, la priorité la plus élevée doit être choisie pour l'émission (voir 6.2.3). Si plusieurs paquets sont attachés à la priorité la plus élevée, le paquet le plus ancien doit être émis le premier.

6.3.3.6.3 Destinations

La destination d'une DLPDU peut être spécifiée comme étant de l'un de types suivants:

- Graph route (chemin de graphe) – la couche réseau spécifie un Graph_ID comme destination. La DLE peut envoyer le paquet à n'importe lesquels des dispositifs associés au graphe en question.
- Source route (chemin de source) – la couche réseau spécifie une adresse de dispositif spécifique comme destination. La DLE doit émettre le paquet sur une liaison vers le voisin en question. Si le dispositif n'a pas de liaison vers le voisin en question, le gestionnaire de réseau doit être avisé qu'une erreur de chemin de source s'est produite.
- Broadcast – l'adresse de destination est diffusée. La DLE doit transmettre en utilisant une liaison de diffusion à partir de la supertrame désignée.
- Proxy Route (chemin de proxy) – la couche réseau indique que l'adresse de destination est celle d'un dispositif candidat au rattachement. La DLE doit émettre le paquet en utilisant une liaison de rattachement.

Le type de destination détermine le Link_type qui peut être utilisé et *in fine* l'intervalle de temps dans lequel la DLPDU peut être émise.

6.3.4 Programmation de liaison

6.3.4.1 Généralités

Tous les dispositifs doivent maintenir un programme de liaisons qui identifie le prochain intervalle de temps qui doit être pris en charge. Toutes les communications sont programmées pour avoir lieu au sein d'une supertrame au cours d'intervalles de temps spécifiques dans la supertrame en question. La prise en charge d'un intervalle de temps consiste soit à être à l'écoute pour détecter un nouveau paquet, soit à émettre un paquet. Lorsqu'un intervalle de temps a un paquet attendant d'être émis et il a aussi une liaison de réception, l'émission du paquet doit avoir la priorité sur la tentative d'écoute pour détecter un nouveau paquet.

La préparation du programme d'une liaison consiste à :

- évaluer les paquets attendant une émission et déterminer le premier ASN qui peut être utilisé pour émettre un paquet; et
- déterminer le premier ASN qui peut être utilisé dans la tentative de réception d'un nouveau paquet.

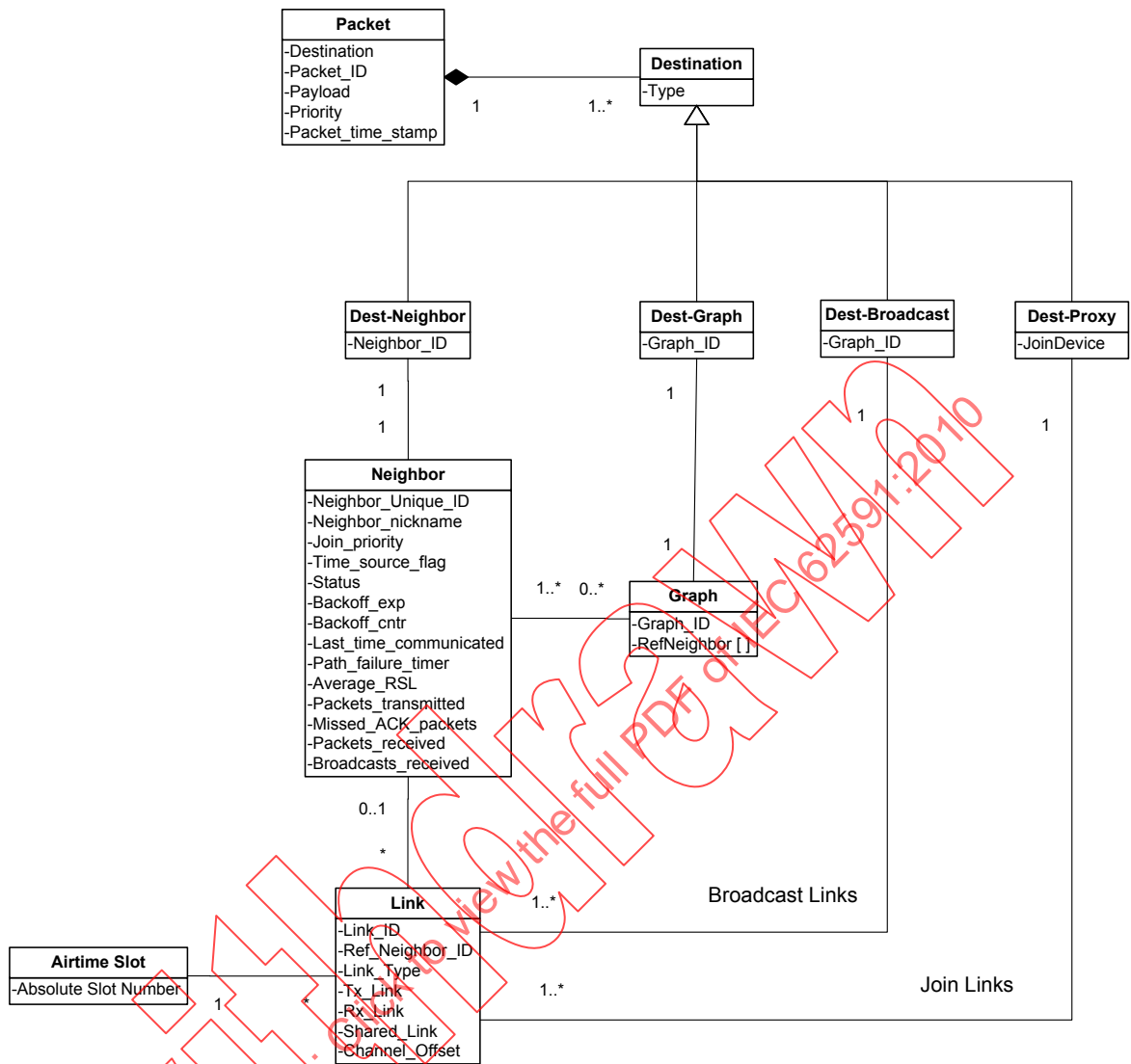
Les événements externes à la DLE et les événements internes peuvent avoir une incidence sur le programme de la liaison. Par conséquent, la DLE doit réévaluer le programme chaque fois qu'un tel événement se produit. Un exemple d'événement interne est la défaillance d'une transaction d'émission de haute priorité. Un exemple d'événement externe est la désactivation d'une supertrame par le gestionnaire de réseau.

6.3.4.2 Prise en charge d'une liaison d'émission

La Figure 21 résume les relations qui ont une incidence sur le calcul du prochain intervalle de temps d'émission devant être pris en charge. Les paquets reçus en provenance de la NLE sont programmés pour un intervalle de temps basé sur le graphe, l'adresse de destination et le fait que le dispositif agisse, oui ou non, comme proxy pour un dispositif rejoignant le réseau. Chaque paquet a une destination et une priorité. Comme spécifié en 6.3.3.6.3, il existe plusieurs types de destinations, chacune de celles-ci a une incidence sur la sélection de l'intervalle de temps. Pour chaque type de destination, le jeu de liaisons d'émission doit être déterminé.

- Pour une destination avec chemins dans le graphe, le jeu de liaisons doit être constitué de toutes les liaisons d'émission pour tous les voisins figurant dans le graphe.
- Pour une adresse de destination de diffusion, le jeu de liaisons doit être constitué de toutes les liaisons de diffusion pour la supertrame désignée.
- Pour l'adresse de destination d'un voisin pris séparément, le jeu de liaisons doit être constitué de toutes les liaisons d'émission menant au voisin en question.
- Pour une adresse de destination de proxy, le jeu de liaisons doit être constitué de toutes les liaisons d'émission du type Join.

Une fois que le jeu de liaisons appropriées pour émettre le paquet est déterminé, le jeu d'intervalles de temps à venir doit être déterminé. L'intersection des intervalles de temps à venir et des liaisons avec un paquet en attente détermine le prochain intervalle de temps devant être programmé pour propager le paquet en question.



Légende

Anglais	Français
Packet	Packet (paquet)
Destination	Destination
Dest-Neighbor	Dest-Neighbor (destination de type voisin)
Dest-Graph	Dest-Graph (destination de type graphe)
Dest-Broadcast	Dest-Broadcast (destination de type diffusion)
Dest-Proxy	Dest-Proxy (destination de type proxy)
Neighbor	Neighbor (voisin)
Graph	Graph (graphe)
Link	Link (liaison)
Broadcast Links	Liaisons de diffusion
Airtime Slot	Airtime Slot (intervalle de temps relatif au temps d'utilisation)
Join Links	Liaisons de rattachement

Figure 21 – Relations utilisées pour la programmation de liaisons

La figure illustre les règles statiques utilisées au niveau de la DLL pour déterminer la liaison qui sera utilisée pour l'émission. Un nouvel objet 'Airtime Slot' (intervalle de temps relatif au temps d'utilisation) est introduit pour modéliser le fait qu'une liaison complètement spécifiée sera associée à un intervalle de temps d'utilisation.

Un paquet contient une destination. La destination peut être spécialisée encore plus par le type de destination – un voisin spécifique, un graphe, une diffusion ou un proxy. Si l'adresse proxy est utilisée, la liaison doit être une liaison de rattachement. Si l'adresse est une adresse de diffusion, une liaison de diffusion doit être utilisée. Si l'adresse est un voisin spécifique, une liaison spécifique associée à un voisin doit être utilisée. Si la destination est un graphe, n'importe quel voisin dans le graphe peut être utilisé.

L'ASN doit être utilisé pour suivre à la trace l'occurrence de chaque intervalle de temps. Une liste ordonnée de liaisons qui vont apparaître bientôt au sein d'une supertrame doit être identifiée en utilisant l'équation suivante:

$$\text{Intervalle de temps de supertrame} = \text{ASN} \% (\text{Num\_Slots de la supertrame}).$$

NOTE Les liaisons peuvent être partagées (voir 6.3.4.4), être utilisées pour annoncer (voir 6.3.4.5) la présence du dispositif à de nouveaux dispositifs tentant de rejoindre le réseau ou être utilisées pour découvrir des voisins (voir 6.3.4.6)

L'algorithme de repli dans des intervalles de temps partagés doit être utilisé pour déterminer le moment où une émission dans un intervalle de temps partagé est autorisée. Les paquets d'annonce (voir 6.3.4.5) sont émis périodiquement comme spécifié par le gestionnaire de réseau.

Les liaisons qui ne peuvent pas être utilisées pour propager un paquet en attente sont ignorées. à partir du jeu de liaisons résultant, la première liaison potentielle et son intervalle de temps associé peuvent être identifiés. Lorsqu'il y a plusieurs paquets qui peuvent être propagés dans l'intervalle de temps en question, les règles spécifiées ci-dessous doivent être utilisées.

- Trouver les paquets ayant la priorité la plus haute;
- S'il n'y a qu'un seul paquet de ce type, sélectionner le paquet en question pour l'émission;
- Sinon, s'il y a plus d'un paquet de ce type, trouver le paquet destiné aux voisins avec lesquels l'absence de toute communication a duré le plus longtemps;
- S'il n'y a qu'un seul paquet de ce type, sélectionner le paquet en question pour l'émission;
- Sinon s'il y a plus d'un paquet de ce type, alors si le Keep_alive_time a expiré avec un voisin, alors préparer une DLPDU Keep-alive pour l'émission vers les voisins avec lesquels l'absence de toute communication a duré le plus longtemps;
- Sinon, si une durée Advertise s'est écoulée, préparer la DLPDU Advertise pour l'émission.

6.3.4.3 Prise en charge d'une liaison de réception

Pour chaque supertrame active, toutes les liaisons de réception doivent être programmées. Le jeu de liaisons de réception à venir peut être calculé de la même manière que les liaisons d'émission (voir 6.3.4.2). S'il n'est pas nécessaire de propager un paquet, toutes les liaisons de réception doivent être prises en charge. Un fois que la liste ordonnée des liaisons est créée, l'intervalle de temps le plus précoce est sélectionné et la liaison au sein de l'intervalle de temps en question ayant le nouveau d'identificateur de supertrame le plus faible doit être le candidat à la prise en charge. S'il n'a pas d'autres liaisons de réception à prendre en charge, le dispositif doit prendre en charge la liaison de réception de découverte. Dans tous les cas, cette liaison de réception sera prise en charge s'il n'y a pas de paquets en attente d'être émis sur cet intervalle de temps ou avant celui-ci.

6.3.4.4 Intervalles de temps partagés

Des intervalles de temps partagés sont attribués à plus d'un dispositif source. Un ou plusieurs de ces dispositifs peuvent tenter d'acheminer un paquet dans l'intervalle de temps et la voie en question. Par conséquent, des collisions peuvent se produire au sein d'un intervalle de temps partagé. Si une collision se produit, le dispositif de destination ne pourra pas recevoir avec succès d'émissions d'une source quelconque et ne produira un acquittement pour aucune d'entre elles. Afin de réduire la probabilité de collisions répétées, les dispositifs source doit utiliser un retard de repli aléatoire lorsque leur émission dans un intervalle de temps partagé n'est pas acquittée (à savoir, aucun ACK n'est reçu par le dispositif source).

Un dispositif doit maintenir deux variables pour chaque voisin: l'exposant de repli (Backoff_exp) et le compteur de repli (Backoff_cntr). Chacune de ces deux variables doit être initialisée à 0. Lorsqu'une transaction dans un intervalle de temps partagé échoue, la période de repli aléatoire doit être calculée en fonction du Backoff_exp. Pour chaque tentative non réussie effectuée par le dispositif source dans un intervalle de temps partagé, le Backoff_exp doit être incrémenté et un ensemble séquentiel de nombres calculé. La valeur de Backoff_exp ne doit pas dépasser celle de Max_backoff_exp. L'ensemble de nombres est constitué des nombres entiers $\{0, 1, \dots, L\}$ où

$$L = ((2^{\text{puissance Backoff\_exp}}) - 1).$$

Le Tableau 41 montre les ensembles de repli pour les valeurs de Backoff_exp allant de un (1) à sept (7).

Tableau 41 – Exemple d'ensembles de sélection de Backoff_cntr

Backoff_exp	Ensemble des valeurs possibles pour Backoff_cntr
1	{0, 1}
2	{0, 1, 2, 3}
3	{0, 1, 2, 3, 4, 5, 6, 7}
4	{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}
5	{0, 1, 2, 3, 4, 5, ..., 28, 29, 30, 31}
6	{0, 1, 2, 3, 4, 5, ..., 60, 61, 62, 63}
7	{0, 1, 2, 3, 4, 5, ..., 124, 125, 126, 127}

Une valeur aléatoire pour Backoff_cntr doit être sélectionnée dans cet ensemble calculé en fonction du Backoff_exp. Pour chaque liaison partagée ultérieure menant au voisin en question, le Backoff_cntr doit être décrémenté. Lorsque le Backoff_cntr correspondant est zéro, seul le dispositif source doit tenter une émission dans l'intervalle de temps partagé en question.

Comme il est également possible qu'une interférence soit la cause d'une perte de paquet, l'exposant et le compteur de repli doivent être maintenus séparément pour chaque voisin. Si la communication avec le voisin en question s'inscrit sur une liaison dédiée, le dispositif doit supposer une dégradation de voie, au lieu d'une collision comme cause de la défaillance dans l'intervalle de temps partagé. De plus, Backoff_exp et Backoff_cntr doivent être remis tous les deux à zéro.

Les messages de diffusion ne doivent pas être émis dans des intervalles de temps partagés.

6.3.4.5 Annonce

Les nœuds qui font déjà partie du réseau peuvent être configurés par le gestionnaire de réseau pour l'annonce au réseau et la facilitation du rattachement de nouveaux dispositifs. L'attribut Advertise_time (voir 6.4.2) fixe l'intervalle auquel les paquets Advertise sont

générés. Chaque fois que l'intervalle `Advertise_time` s'écoule, un paquet `Advertise` doit être émis sur la première liaison d'émission non partagée disponible. Lorsque l'`Advertise_time` est mis à zéro, un paquet `Advertise` doit être généré chaque fois qu'une liaison d'émission non partagée est disponible.

Un paquet `Advertise` peut être envoyé sur une ligne d'émission non partagée qui n'est pas en cours d'utilisation. Tous les autres trafics ont une plus haute priorité que l'annonce.

6.3.4.6 Découverte de voisins

Les dispositifs doivent être en permanence à l'écoute pour des communications provenant du voisin existant et pour des communications provenant de nouveaux voisins. Une surveillance continue des voisins et la découverte de nouveaux voisins sont critiques pour la maintenance de la maille et l'amélioration de la fiabilité des communications.

À la réception d'une DLPDU quelconque, provenant d'un nouveau voisin, le dispositif doit invoquer l'indication d'événement de DLM avec l'adresse du voisin en question et le niveau RSL pour la DLPDU en question. Périodiquement, la NLE communique la liste de nouveaux voisins au gestionnaire de réseau. Le dispositif peut recevoir une DLPDU, adressée au dispositif, provenant d'un nouveau dispositif tentant de rejoindre le réseau. Lorsque cela se produit, le dispositif doit ajouter le dispositif au tableau de voisins. Le dispositif doit alors dupliquer les liaisons de rattachement et insérer le dispositif candidat au rattachement comme le voisin de la liaison. Ces voisins sont périssables et peuvent être détruits en fonction du `Last_time_communicated` pour dégager de la place pour les nouveaux voisins (voir 6.3.3.4). Lorsqu'un voisin est supprimé, les liaisons de rattachement créées automatiquement reliant le dispositif au dispositif de candidat au rattachement doivent être supprimées.

Pour aider à la découverte de nouveaux voisins, le dispositif se met à l'écoute chaque fois que possible sur des liaisons de découverte. Les liaisons de découverte sont partagées par tous les dispositifs dans le réseau. Outre l'écoute, un dispositif doit aussi émettre de façon aléatoire une DLPDU `Keep-alive` dans une liaison de découverte d'émission. Cela accorde à d'autres dispositifs à l'écoute sur la liaison de découverte l'occasion de découvrir le dispositif.

La fréquence la plus lente à laquelle le dispositif émet sur la liaison de découverte est limitée par le `Discovery_time` (voir 6.4.2). Pour programmer l'émission de découverte, le dispositif doit sélectionner une durée aléatoire entre 0 et `Discovery_time` et l'utiliser pour initialiser le temporisateur de découverte. Lorsque le temporisateur en question expire, le dispositif doit mettre le fanion `TimeToDiscover` à `TRUE` et programmer la prochaine émission de découverte chronométrée de façon aléatoire. Si la valeur de `Discovery_time` est '-1' (c'est-à-dire 0xFFFFFFF), la DLE ne doit pas desservir la liaison de découverte.

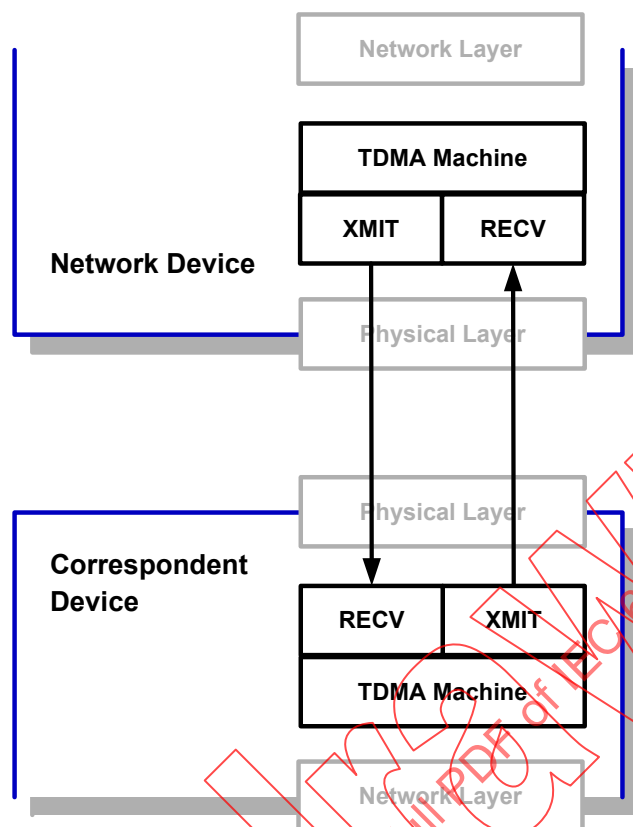
Une fois que le fanion `TimeToDiscover` est mis, la programmation de liaisons émettra un paquet `Keep-alive` à la première liaison d'émission de découverte disponible. Le paquet `Keep-alive` doit être adressé au voisin relié ayant la valeur la plus ancienne de `Last_time_communicated`.

6.3.5 Fonctionnement de MAC

6.3.5.1 Généralités

La présente spécification décompose la sous-couche MAC en trois composants primaires comme montré à la Figure 22. Les composants sont:

- La machine TDMA qui spécifie le fonctionnement global de la sous-couche MAC;
- L'émetteur qui envoie une DLPDU – à savoir la machine XMIT; et
- Le récepteur qui est à l'écoute d'une liaison et reçoit une DLPDU – à savoir la machine RECV.



Légende

Anglais	Français
Network Device	Dispositif de réseau
TDMA Machine	Machine TDMA
XMIT	XMIT
Physical Layer	Couche physique
Correspondent Device	Dispositif de correspondant
RECV	RECV
XMIT	XMIT
Network Layer	Couche réseau

Figure 22 – Composants MAC

6.3.5.2 Machine TDMA

6.3.5.2.1 Généralités

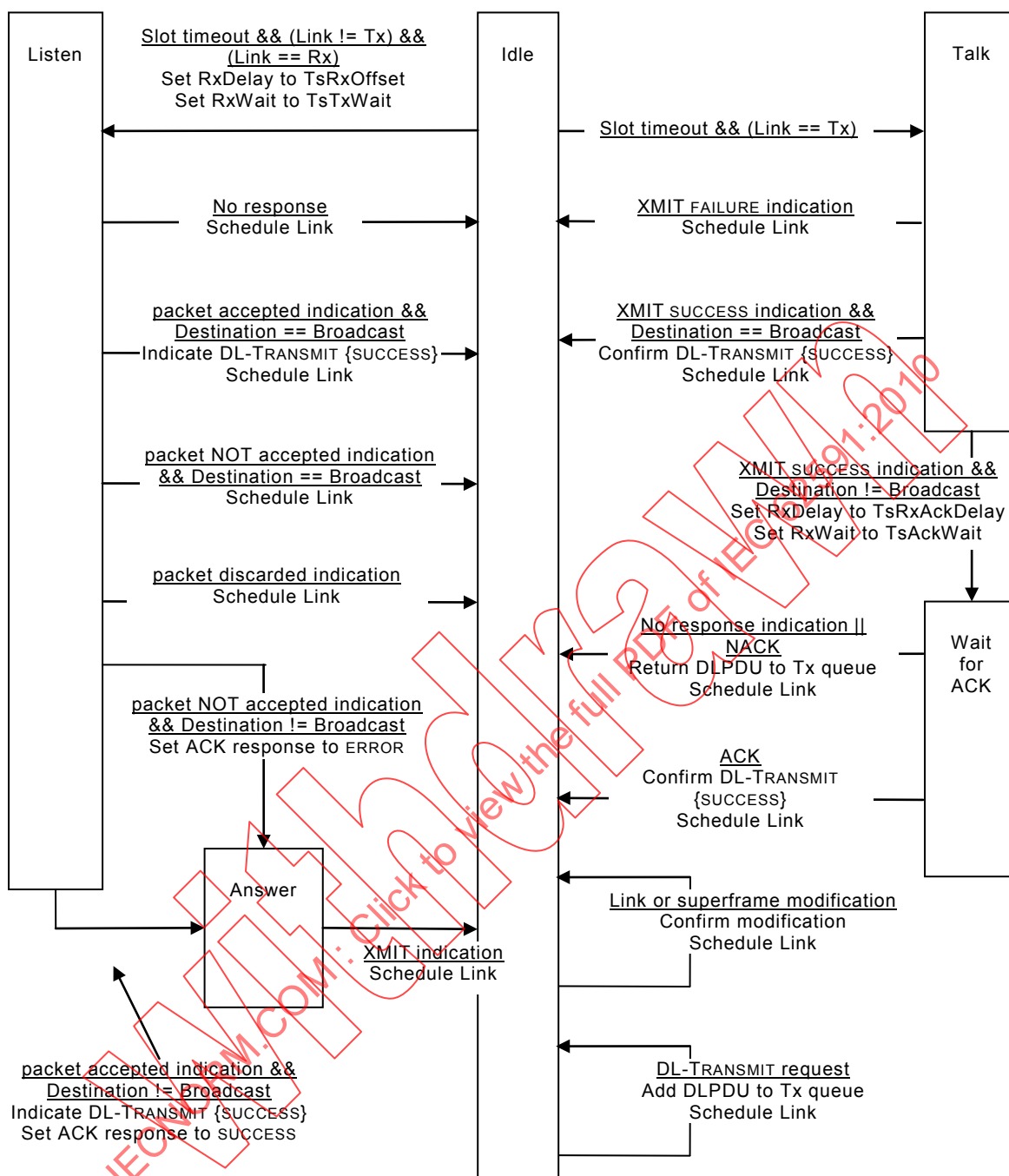
Le fonctionnement de la machine TDMA est montré à la Figure 23. Les transitions d'état sont montrées au Tableau 42. Le fonctionnement de la machine TDMA commence dans l'état "Idle" (repos), lorsque le dispositif rejoint un réseau; est configuré avec une liste de superframes, de graphes de liaisons; et commence à recevoir des DLPDU provenant d'autres dispositifs ou une demande d'émission provenant de la NLE. Le fonctionnement normal peut être divisé en quatre responsabilités:

- gérer le programme,
- émettre des DLPDU vers d'autres dispositifs,
- recevoir des DLPDU provenant d'autres dispositifs, et
- maintenir la synchronisation du temps.

La gestion de programme inclut la création et la maintenance de supertrames, de liaisons et statistiques relatives aux voisins. À mesure que les DLPDU sont acquises et propagées, les programmes doivent être mis à jour. La programmation de liaisons consiste à évaluer les supertrames actives, les liaisons, les DLPDU en attente d'émission et à identifier le prochain intervalle de temps qui a besoin de prise en charge.

Le programme détermine la répartition des DLPDU. Toutes les liaisons de réception doivent être prises en charge en tentant une réception de DLPDU. Sachant que les liaisons doivent être allouées pour prendre en charge des répétitions de tentatives possibles, il y a souvent plus de liaisons de réception que de liaisons d'émission. Sachant qu'une émission est souvent réussie, plusieurs des liaisons de réception seront inutilisées, ne contenant pas d'émission correspondante. La plupart des DLPDU reçues contiennent des DLPDU destinées à la NLE.

IECNORM.COM: Click to view the full PDF of IEC 62591:2010



Légende

Anglais	Français
Listen	Écouter
Slot timeout && (Link != Tx) && (Link == Rx) Set RxDelay to TsRxOffset Set RxWait to TsTxWait	Slot timeout && (Link != Tx) && (Link == Rx) Mettre RxDelay à TsRxOffset Mettre RxWait à TsTxWait
No response Schedule Link	Aucune réponse Programmer la liaison
packet accepted indication && Destination == Broadcast Indicate DL-TRANSMIT {SUCCESS}	packet accepted indication && Destination == Broadcast Indiquer DL-TRANSMIT {SUCCESS}

Anglais	Français
Schedule Link	Programmer la liaison
packet NOT accepted indication && Destination == Broadcast Schedule Link	packet NOT accepted indication && Destination == Broadcast Programmer la liaison.
packet discarded indication Schedule Link	Indication de paquet rejeté Programmer la liaison.
packet NOT accepted indication && Destination != Broadcast Set ACK response to ERROR	packet NOT accepted indication && Destination != Broadcast Mettre la réponse ACK à ERROR
Answer	Répondre
XMIT indication Schedule Link	XMIT indication Programmer la liaison.
packet accepted indication && Destination != Broadcast Indicate DL-TRANSMIT {SUCCESS} Set ACK response to SUCCESS	packet accepted indication && Destination != Broadcast Indiquer DL-TRANSMIT {SUCCESS} Mettre la réponse ACK à SUCCESS
Idle	Repos
Slot timeout && (Link == Tx)	Slot timeout && (Link == Tx)
XMIT FAILURE indication Schedule Link	XMIT FAILURE indication Programmer la liaison
XMIT SUCCESS indication && Destination == Broadcast Confirm DL-TRANSMIT {SUCCESS} Schedule Link	XMIT SUCCESS indication && Destination == Broadcast Confirm DL-TRANSMIT {SUCCESS} Programmer la liaison.
XMIT SUCCESS indication && Destination != Broadcast Set RxDelay to TsRxAckDelay Set RxWait to TsAckWait	XMIT SUCCESS indication && Destination != Broadcast Mettre RxDelay à TsRxAckDelay Mettre RxWait à TsAckWait
No response indication NACK Return DLPDU to Tx queue Schedule Link	No response indication NACK Retourner DLPDU à la file d'attente Tx Programmer la liaison.
ACK Confirm DL-TRANSMIT {SUCCESS} Schedule Link	ACK Confirm DL-TRANSMIT {SUCCESS} Programmer la liaison.
Link or superframe modification Confirm modification Schedule Link	Modification de liaison ou supertrame Confirmer modification Programmer la liaison.
DL-TRANSMIT request Add DLPDU to Tx queue Schedule Link	DL-TRANSMIT request Ajouter DLPDU à la file d'attente d'émission Programmer la liaison.
Talk	Parler
Wait for ACK	Attendre l'ACK.

Figure 23 – Diagramme d'état TDMA

Tableau 42 – Diagramme d'état TDMA

État courant	Événement ou condition => action	Prochain état
Idle (repos)	Slot timeout && (Link == Tx)	Talk (parler)
Idle	Slot timeout && (Link != Tx) && (Link == Rx) => Set RxDelay to TsRxOffset; Set RxWait to TsTxWait	Listen (écouter)
Idle	Link or superframe modification => Confirm modification; Schedule Link	Idle
Idle	DL-TRANSMIT request => Add DLPDU to Tx queue; Schedule Link	Idle
Talk	XMIT FAILURE indication => Schedule Link	Idle
Talk	XMIT SUCCESS indication && Destination == Broadcast => Confirm DL-TRANSMIT {SUCCESS}; Schedule Link	Idle
Talk	XMIT SUCCESS indication && Destination != Broadcast => Confirm DL-TRANSMIT {SUCCESS}; Schedule Link	Wait for ACK (attendre l'acquittement)
Wait for ACK	No response indication NACK => Return DLPDU to Tx queue; Schedule Link	Idle
Wait for ACK	ACK => Confirm DL-TRANSMIT {SUCCESS}; Schedule Link	Idle
Listen	No response => Schedule Link	Idle
Listen	packet accepted indication && Destination == Broadcast => Indicate DL-TRANSMIT {SUCCESS}; Schedule Link	Idle
Listen	packet NOT accepted indication && Destination == Broadcast => Schedule Link	Idle
Listen	packet discarded indication => Schedule Link	Idle
Listen	packet NOT accepted indication && Destination != Broadcast => Set ACK response to ERROR	Answer
Listen	packet accepted indication && Destination != Broadcast => Indicate DL-TRANSMIT {SUCCESS}; Set ACK response to SUCCESS	Answer (répondre)
Answer	XMIT indication => Schedule Link	Idle

6.3.5.2.2 État de repos « Idle »

Les événements suivants peuvent survenir pendant l'état "Idle"

- a) Slot timeout (expiration de délai de l'intervalle de temps): l'activité la plus fréquente accomplie par la machine TDMA est la prise en charge d'un événement expiration de délai d'intervalle de temps. Cet événement indique qu'un intervalle de temps a besoin d'une prise en charge.
 - 1) Si l'intervalle de temps a expiré pour une liaison d'émission et il y a une DLPDU en attente d'émission, le TDMA doit passer à l'état "Talk" (état "parler").
 - 2) Sinon s'il s'agit d'une liaison de réception, le TDMA doit passer à l'état "Listen" (l'état "écouter").

- b) Modification apportée à la liste de supertrames ou de liaisons du dispositif: elle a une incidence sur la programmation des liaisons. Ces modifications résultent typiquement dans une révision des définitions des liaisons et peuvent entraîner un nombre différent de tentatives d'émissions et de réceptions par seconde.
- c) DL-Transmit request (demande d'émission de DL): cela doit ajouter une DLPDU à la file d'attente d'émission (Tx) du dispositif et cela peut avoir une incidence sur la programmation des liaisons.

6.3.5.2.3 État de parole « Talk »

Dans cet état, la DLE doit tenter d'émettre la DLPDU vers son/ses voisin(s). Le fonctionnement de cet état est spécifié en 6.3.5.3. La tentative débouchera soit sur un succès, soit sur l'un de plusieurs résultats négatifs.

- La propagation réussie d'une DLPDU avec une adresse de destination qui est l'adresse de diffusion a lieu dès que la DLPDU est émise. Le tampon des DLPDU doit être libéré à l'achèvement de l'émission de la DLPDU et le programme de liaisons doit être évalué.
- La propagation réussie d'une DLPDU avec une adresse de destination qui n'est pas l'adresse de diffusion a lieu lorsqu'un ACK réussi validé est reçu. Cela indique que la propagation de DLPDU a été achevée avec succès et le tampon des DLPDU doit être libéré.
- Si l'ACK contient un code de réponse d'erreur, le voisin de destination a refusé la DLPDU. Lorsque cela se produit, la DLPDU doit être retenue et sa propagation de nouveau tentée.
- Si aucun ACK n'est reçu, la DLPDU doit être reprogrammée et l'émission de nouveau tentée. Si le délai de la DLPDU expire, la DLPDU en file d'attente est retournée à la NLE pour prise de décision et un possible réacheminement.
- Si l'émission échoue dans la machine XMIT, la transaction doit être abandonnée et le programme de liaisons doit être évalué.

À la suite d'une émission réussie et si la destination de la DLPDU n'est pas de diffusion, la machine TDMA doit passer à l'état "Wait for ACK" (attendre un ACK), qui est décrit en 6.3.5.4. À l'entrée dans cet état, le temporisateur RxDelay doit être mis à TsRxAckDelay et le temporisateur RxWait (fenêtre de réception) doit être mis à TsAckWait.

6.3.5.2.4 État attente confirmation « Wait for ACK »

La machine TDMA reste dans cet état jusqu'à ce que la machine RECV soit terminée. Si la machine RECV indique qu'il y avait "No Response" (aucune réponse), la transaction échoue. S'il s'agissait d'une liaison partagée, le Backoff_exp et le Backoff_cnr doivent être réévalués pour produire une période de repli aléatoire avant la prochaine tentative d'émission dans un intervalle de temps partagé (voir 6.3.4.4). S'il ne s'agissait pas d'une liaison partagée, le Backoff_exp et le Backoff_cnr doivent être remis à zéro. Alors, après l'évaluation du programme de liaisons, la machine TDMA doit passer à l'état "Idle".

Si un ACK contenant un code de réponse (Response Code) "Success" est reçu, la communication avait réussi. Cependant, si un code de réponse d'erreur a été reçu, le voisin n'a pas accepté la DLPDU. Dans un cas comme dans l'autre, la synchronisation du temps réseau peut être évaluée en utilisant le champ "Time-adjustment" (réglage de l'heure) dans la réponse du voisin. Si le voisin est une source de temps pour le dispositif, alors le dispositif doit resynchroniser son temps réseau en utilisant le champ "Time-adjustment". Si le voisin n'est pas une source de temps, aucune correction de temps ne doit être effectuée.

La réception d'un ACK indique que le dispositif voisin a reçu le message avec succès et l'a accepté. Le tampon des DLPDU contenant la DLPDU émise doit être libéré. La machine TDMA doit passer à "Idle" et évaluer le programme de liaisons.

6.3.5.2.5 État écoute « Listen »

Dans cet état, la DLE doit tenter de recevoir la DLPDU. Le fonctionnement de cet état est spécifié en 6.3.5.4. La réception d'une DLPDU a trois résultats possibles:

- la destination finale de la DLPDU est la DLE elle-même et alors la DLPDU doit être retenue pour traitement par la DLE;
- la destination finale de la DLPDU n'est pas la DLE elle-même et alors la DLPDU doit être transmise par la DLE vers la destination finale du message;
- la DLPDU n'est pas adressée au dispositif et alors la DLPDU doit être rejetée.

Dans tous les cas, lorsqu'une DLPDU est reçue, l'entrée correspondante du tableau de voisins doit être mise à jour ou, s'il y a lieu, l'entrée du tableau doit être créée.

Le cycle de réception des DLPDU consiste en une tentative de réception d'une DLPDU, la validation de toute DPDU reçue et, si l'adresse de destination de la DLPDU n'est pas de diffusion, l'émission d'une réponse à une DLPDU reçue valide.

Lors d'une tentative de réception d'une DLPDU, le temporisateur RxDelay doit être mis à TsRxOffset et le temporisateur RxWait (fenêtre de réception) doit être mis à TsTxWait. Le temps TsTxWait fixe la durée de la fenêtre de réception et représente la plus grande dérive temporelle entre dispositifs voisins autorisée par la présente norme.

Si aucune DLPDU n'est reçue dans l'état "Listen", la machine TDMA doit réévaluer le programme de liaisons et retourner à l'état "Idle". Si une DLPDU est reçue, la synchronisation du temps réseau doit être évaluée. TsError doit être calculé en prenant la différence entre l'heure réelle de début et l'heure de début prévue de la DLPDU reçue. Si le voisin envoyant la DLPDU est l'une des sources de temps de la DLE, le dispositif doit utiliser le TsError mesuré pour resynchroniser son temps réseau. Si le voisin n'est pas une source de temps, aucune correction de temps ne doit être effectuée.

Alors qu'il est à l'état "Listen" et à la réception réussie d'une DLPDU, le dispositif doit déterminer s'il accepte la DLPDU ou la rejette. Le dispositif accepte ou rejette la DLPDU en fonction de la priorité de la DLPDU, du Priority_threshold courant (voir 6.2.3) et du nombre de tampons de DLPDU actuellement occupés dans le dispositif. Si la DLPDU est acceptée par le dispositif, elle est soit consommée par la DLE elle-même, soit transmise à la NLE en utilisant d'indication de DL-TRANSMIT et la DLE doit mettre à jour le programme de liaisons.

Une fois que la couche liaison de données a déterminé la décision relative à la DLPDU (accepter ou rejeter la DLPDU), la machine TDMA doit quitter l'état "Listen" comme suit:

- Si l'adresse de destination de la DLPDU est l'adresse de diffusion, la machine TDMA doit passer à l'état "Idle".
- Si l'adresse de destination n'est pas l'adresse de diffusion et la DLPDU a été acceptée, la DLE doit émettre une DLPDU ACK, avec une valeur de code de réponse égale à "Success", vers l'expéditeur de la DLPDU juste reçue. La machine TDMA doit passer à l'état "Answer".
- Si l'adresse de destination n'est pas l'adresse de diffusion et la DLPDU n'a pas été acceptée, la DLE doit émettre une DLPDU ACK, avec une valeur de code de réponse égale à un code d'erreur, vers l'expéditeur de la DLPDU juste reçue. La machine TDMA doit passer à l'état "Answer".

6.3.5.2.6 État réponse "Answer"

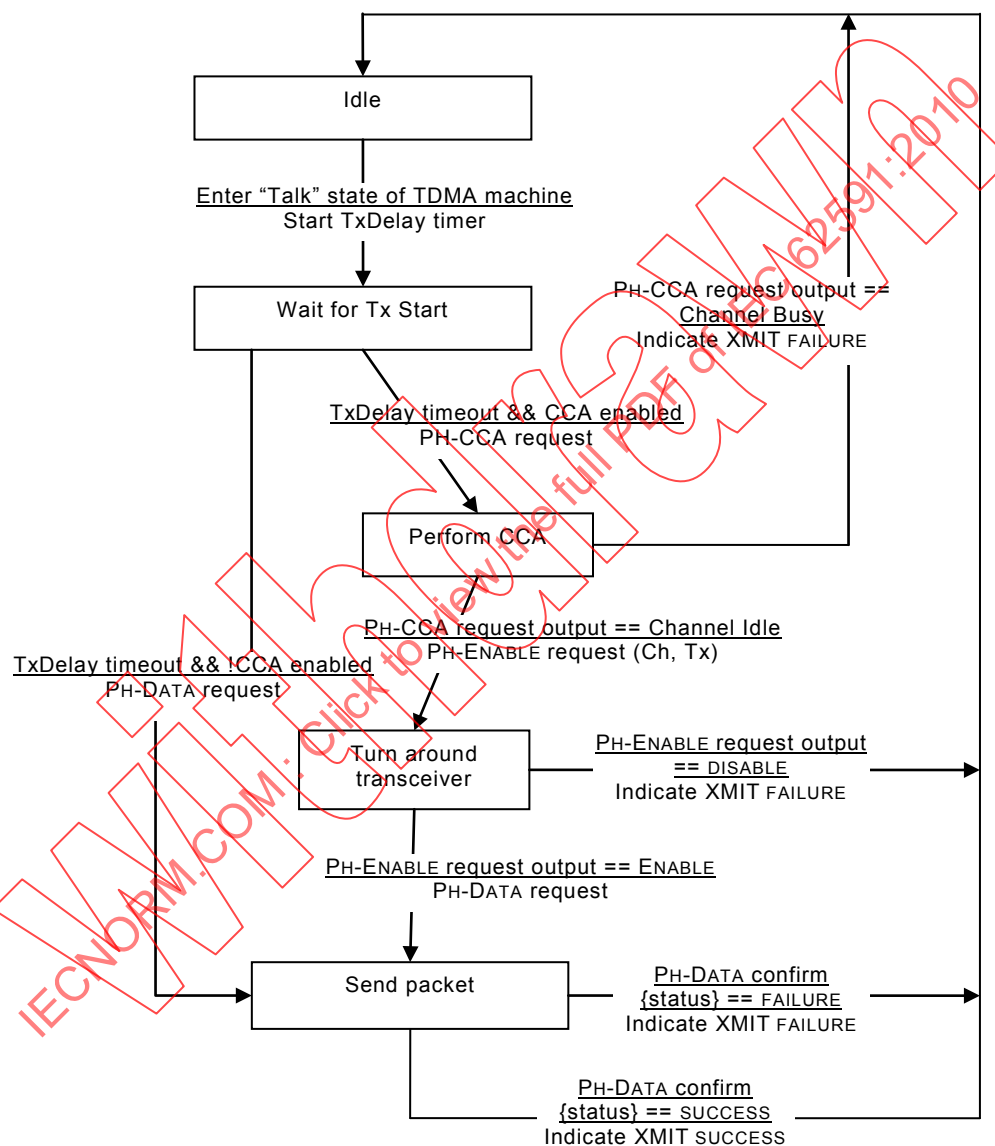
La DLE doit copier le TsError mesuré dans le champ "Time-adjustment" de la DLPDU ACK. La machine TDMA fonctionne comme une machine d'émission d'ACK décrite en 6.3.5.5.

Après l'achèvement de l'émission de l'ACK, la machine TDMA doit évaluer le programme de liaisons et passer à l'état "Idle".

6.3.5.3 Machine XMIT

6.3.5.3.1 Généralités

La Figure 24 montre le diagramme d'état utilisé pour émettre une DLPDU par l'intermédiaire d'un intervalle de temps soit dédié, soit partagé. Les transitions d'état sont montrées au Tableau 43. Cette machine est démarrée pour émettre une DLPDU.



Légende

Anglais	Français
Idle	Repos
Enter "Talk" state of TDMA machine Start TxDelay timer	Entrer dans l'état "Talk" de la machine TDMA Démarrer le temporisateur TxDelay
Wait for Tx Start	Attendre le début de l'émission Tx
TxDelay timeout && CCA enabled PH-CCA request	TxDelay timeout && CCA enabled Demande de CCA de PH

Anglais	Français
Perform CCA	Exécuter CCA
TxDelay timeout && !CCA enabled PH-DATA request	TxDelay timeout && !CCA enabled Demande de données PH
PH-CCA request output == Channel Idle PH-ENABLE request (Ch, Tx)	PH-CCA request output == Channel Idle PH-ENABLE request (Ch, Tx)
Turn around transceiver	«Tourner autour» de l'émetteur-récepteur transceiver
PH-ENABLE request output == DISABLE Indicate XMIT FAILURE	PH-ENABLE request output == DISABLE Indiquer défaillance de XMIT
PH-ENABLE request output == ENABLE PH-DATA request	PH-ENABLE request output == ENABLE Demande de données PH
Send packet	Envoyer un paquet
PH-DATA confirm {status} == FAILURE Indicate XMIT FAILURE	PH-DATA confirm {status} == FAILURE Indiquer défaillance de XMIT
PH-DATA confirm {status} == SUCCESS Indicate XMIT SUCCESS	PH-DATA confirm {status} == SUCCESS Indiquer succès de XMIT
PH-CCA request output == Channel Busy Indicate XMIT FAILURE	PH-CCA request output == Channel Busy Indiquer défaillance de XMIT

Figure 24 – Diagramme d'état XMIT

Tableau 43 – Transitions d'états de XMIT

État courant	Événement ou condition => action	Prochain état
Idle	Enter "Talk" state of TDMA machine => Start TxDelay timer	Wait for Tx Start (Attendre le début d'émission)
Wait for Tx Start	TxDelay timeout && !CCA enabled => PH-DATA request {DLPDU}	Send packet (Envoyer le paquet)
Wait for Tx Start	TxDelay timeout && CCA enabled => PH-CCA request	Perform CCA (Effectuer une CCA)
Perform CCA	PH-CCA request output == Channel Idle => PH-ENABLE request {channel number, transmit}	Turn around transceiver (Tourner autour de l'émetteur-récepteur)
Perform CCA	PH-CCA request output == Channel Busy => XMIT FAILURE indication	Idle
Turn around transceiver	PH-ENABLE request output == ENABLE PH-DATA request {DLPDU}	Send packet
Turn around transceiver	PH-ENABLE request output == DISABLE XMIT FAILURE indication	Idle
Send Packet	PH-DATA confirm {status} == success => XMIT SUCCESS indication	Idle
Send Packet	PH-DATA confirm {status} == FAILURE => XMIT FAILURE indication	Idle

6.3.5.3.2 État de repos "Idle"

Lorsque la machine XMIT a démarré, la DLE doit démarrer le temporisateur TxDelay. Le temps TxDelay doit être fixé de manière que l'émission débute au centre de la fenêtre de réception du voisin. Le dispositif peut utiliser le temps TxDelay pour construire la DLPDU, faire fonctionner son moteur de chiffrement AES 128 bits et générer le MIC pour la DLPDU.

La machine doit passer à l'état "Wait for Tx start".

6.3.5.3.3 État attente pour démarrer l'émission "Wait for Tx start"

Dans l'état "Wait for Tx Start", la DLE doit sélectionner la voie pour l'émission (voir 6.3.3.3.2). Lorsque le temporisateur TxDelay expire, la machine XMIT change l'état. Deux séquences d'émission sont possibles.

- a) Si CCAEnabled est FALSE, la DLE doit émettre une demande PH-DATA request d'émettre la DLPDU et faire passer l'état à "Send packet".
- b) Si CCAEnabled est TRUE, la DLE doit émettre la demande PH-CCA request et faire passer l'état à "Perform CCA".

6.3.5.3.4 État lancer CCA "Perform CCA"

La DLE doit attendre que la PhE ait achevé l'évaluation de voie libre.

- a) Si la DLE reçoit une demande de CCA produite avec le statut "Channel idle" provenant de la PhE, alors
 - 1) la machine XMIT doit passer à l'état "Turn around transceiver";
 - 2) la PhE doit être initialisée en utilisant la demande PH-ENABLE request pour mettre l'émetteur-récepteur dans le mode émission.
- b) Sinon, si la DLE reçoit une demande de CCA produite avec le statut "Channel busy", le moteur XMIT doit s'arrêter, indiquer que la tentative d'émission a échoué et ne doit pas émettre la DLPDU.

6.3.5.3.5 État retournement du transmetteur "Turn around transceiver"

Lorsque la DLE reçoit une demande PH-ENABLE requête produite avec le statut "Enable" provenant de la PhE, la machine XMIT doit passer à l'état "Send packet". Sinon, si le statut retourné est "Disable", le moteur XMIT doit s'arrêter, indiquer que la tentative d'émission a échoué et ne doit pas émettre la DLPDU.

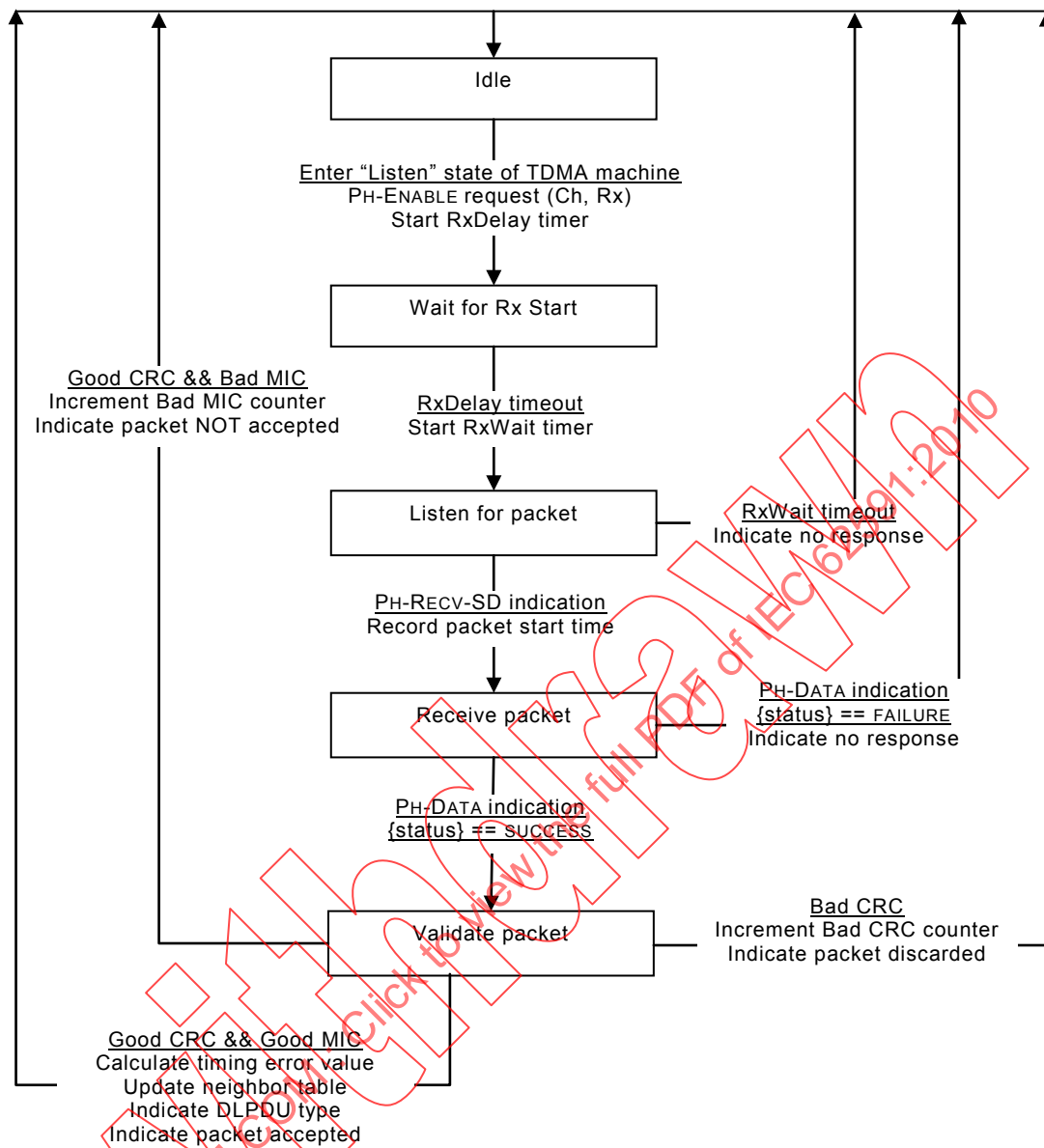
6.3.5.3.6 État envoi de paquet "Send packet"

La DLE doit démarrer l'émission de paquets en utilisant le service de demande PH-DATA request de la PhE. Lorsque la DLE reçoit la confirmation PH-DATA confirm provenant de la PhE, la machine XMIT doit s'arrêter et indiquer que l'émission avait soit réussi, soit échoué, cela dépendant du statut retourné dans la primitive de confirmation "PH-DATA confirm".

6.3.5.4 Machine RECV

6.3.5.4.1 Généralités

La Figure 25 montre la machine qui est utilisée pour recevoir une DLPDU. Cette machine est démarrée pour recevoir une DLPDU qui est émise par l'un des voisins du dispositif ou immédiatement après l'émission d'une DLPDU pour recevoir une DLPDU ACK. Les transitions d'état sont montrées au Tableau 44.



Légende

Anglais	Français
Good CRC && Bad MIC Increment Bad MIC counter Indicate packet NOT accepted	Good CRC && Bad MIC Incrémenter compteur Bad MIC Indiquer paquet NON accepté
Idle	Repos
Enter "Listen" state of TDMA machine PH-ENABLE request (Ch, Rx) Start RxDelay timer	Entrer dans l'état "Listen" de la machine TDMA PH-ENABLE request (Ch, Rx) Démarrer le temporisateur RxDelay
Wait for Rx Start	Attendre le démarrage de Rx
RxDelay timeout Start RxWait timer	Expiration de délai RxDelay Démarrer temporisateur RxWait
Listen for packet	Se mettre à l'écoute pour détecter paquet
PH-RECV-SD indication Record packet start time	PH-RECV-SD indication Consigner l'heure de début du paquet

Anglais	Français
Receive packet	Recevoir paquet
PH-DATA indication {status} == SUCCESS	PH-DATA indication {status} == SUCCESS
Validate packet	Valider le paquet
RxWait timeout Indicate no response	Expiration de délai RxWait Indiquer aucune réponse
PH-DATA indication {status} == FAILURE Indicate no response	PH-DATA indication {status} == FAILURE Indiquer aucune réponse
Bad CRC Increment Bad CRC counter Indicate packet discarded	Bad CRC Incrémenter le compteur de Bad CRC Indiquer paquet rejeté
Good CRC && Good MIC Calculate timing error value Update neighbor table Indicate DLPDU type Indicate packet accepted	Good CRC && Good MIC Calculer la valeur de l'erreur de temporisation Mettre à jour le tableau de voisins Indiquer le type de DLPDU Indiquer paquet accepté

Figure 25 – Diagramme d'état RECV

Tableau 44 – Diagramme d'état RECV

État courant	Événement ou condition => action	Prochain état
Idle	Enter "Listen" state of TDMA machine => PH-ENABLE request (channel number, receive) Start RxDelay timer	Wait for Rx start (Attendre le début de la réception)
Wait for Rx start	RxDelay timeout => Start RxWait timer	Listen for packet (Etre à l'écoute pour un paquet)
Listen for packet	RxWait timeout => Indicate no response	Idle
Listen for packet	PH-RECV-SD indication => Record packet start time	Receive packet (Recevoir le paquet)
Receive packet	PH-DATA indication {status} == FAILURE => Indicate no response	Idle
Receive packet	PH-DATA indication {status} == <u>SUCCESS</u>	Validate packet (Valider le paquet)
Validate packet	Bad CRC => Increment Bad CRC counter; Indicate packet discarded	Idle
Validate packet	Good CRC && Bad MIC => Increment Bad MIC counter; Indicate packet NOT accepted	Idle
Validate packet	Good CRC && Good MIC => Calculate timing error value; Update neighbor table; Indicate DLPDU type; Indicate packet accepted	Idle

6.3.5.4.2 État de repos « Idle »

Lorsque la machine RECV a démarré:

- a) la DLE doit sélectionner une voie en fonction du décalage de voie de départ de la liaison et du nombre courant d'intervalles de temps absolus;
- b) la DLE doit mettre l'émetteur-récepteur dans le mode réception en utilisant le service de demande " PH-ENABLE request" de la PhE;
- c) la DLE doit démarrer le temporisateur RxDelay; et
- d) La machine RECV doit passer à l'état "Wait for Rx start".

Au cours du RxDelay, l'émetteur-récepteur s'établit et se synchronise à la voie correcte.

6.3.5.4.3 État attente début réception « Wait for Rx start »

Lorsque le temporisateur RxDelay expire, la machine RECV doit passer à l'état "Listen for packet" et démarrer le temporisateur RxWait.

6.3.5.4.4 État écoute des paquets « Listen for packet »

L'émetteur-récepteur reçoit l'instruction de commencer à se mettre à l'écoute de la voie en utilisant le service d'indication "Ph-Recv-SD indication" de la PhE. La machine RECV demeure dans l'état "Listen for Packet" jusqu'à ce que soit 1) le début d'une DLPDU comme indiqué par la réception de la fin d'un délimiteur de début de PhpDU soit détecté, soit 2) le temporisateur RxWait expire. Si le temporisateur RxWait expire, la tentative de réception échoue, le moteur RECV doit s'arrêter en indiquant "No Response" et la DLE doit mettre à jour les statistiques de la communication.

Si le délimiteur de début prévu est détecté, son heure d'arrivée doit être enregistrée et le moteur RECV doit passer à l'état "Receive Packet" pour se saisir de la DLPDU.

6.3.5.4.5 État réception de paquets « Receive packet »

Lorsque la machine RECV reçoit la primitive du service d'indication "PH-DATA indication" provenant de la PhE, elle doit effectuer une évaluation initiale de la DLPDU reçue. Si les adresses de DLPDU ne sont pas celles prévues pour la liaison, la machine RECV doit s'arrêter en indiquant "No Response" et la DLE doit mettre à jour les statistiques de communication. Sinon, la machine RECV doit passer à l'état "Validate packet".

6.3.5.4.6 État validation des paquets « Validate packet »

En l'absence d'erreurs dans la DLPDU reçue, une validation supplémentaire doit être effectuée. Si le CRC est incorrect, la PhpDU était corrompue avant ou pendant la réception; la machine RECV doit s'arrêter en indiquant "packet discarded" (paquet rejeté) et la DLE doit mettre à jour les statistiques de la communication.

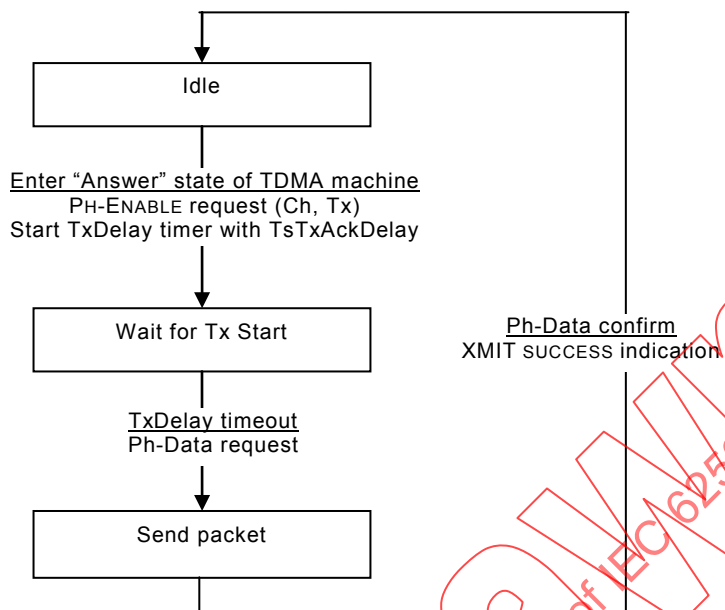
Si le CRC reçu est correct, le MIC chiffré reçu doit être calculé et vérifié. Si le MIC reçu n'est pas tel que prévu, cela indique une possible attaque. Par conséquent, la réception est considérée comme étant un échec et la DLE doit mettre à jour la sécurité et les statistiques de la communication. La machine RECV doit s'arrêter en indiquant "packet NOT accepted" (paquet NON accepté)

Si le CRC et le MIC sont corrects tous les deux, la réception est considérée réussie et la machine RECV doit s'arrêter en indiquant le type de la DLPDU reçue.

6.3.5.5 Transmission d'acquisition par la machine (ACK)

Cette machine est démarrée pour acquitter la réception d'une DLPDU. La DLE doit copier le TsError mesuré dans le champ "Time-adjustment" de la DLPDU ACK en question. La machine

TDMA doit utiliser le diagramme d'état montré à la Figure 26 pour émettre la DLPDU. Les transitions d'état sont montrées au Tableau 45.



Légende

Anglais	Français
Idle	Repos
Enter "Answer" state of TDMA machine PH-ENABLE request (Ch, Tx) Start TxDelay timer with TsTxAckDelay	Entrer dans l'état "Answer" de machine TDMA PH-ENABLE request (Ch, Tx) Démarrer le temporisateur TxDelay avec TsTxAckDelay
Wait for Tx Start	Attendre le début de l'émission Tx
TxDelay timeout Ph-Data request	Expiration de délai TxDelay Ph-Data request
Send packet	Envoyer le paquet
Ph-Data confirm XMIT SUCCESS indication	Ph-Data confirm Indication de succès de XMIT

Figure 26 – Diagramme d'état pour l'émission des ACK

Tableau 45 – Diagramme d'état pour l'émission des ACK

État courant	Événement ou condition => action	Prochain état
Idle	Enter "Answer" state of TDMA machine => Ph-Enable request {channel number, transmit}; Start TxDelay timer with TsTxAckDelay;	Wait for Tx Start
Wait for Tx Start	TxDelay timeout => PH-Data request {ACK DLPDU}	Send Packet
Send Packet	PH-Data confirm => XMIT SUCCESS indication	Idle

6.4 DL-management-information

6.4.1 Généralités

Plusieurs attributs de DL doivent être préconfigurés avant qu'une DLE ne puisse fonctionner correctement. Ces attributs sont définis en 6.4.2. Les actions de la DLE qui peuvent être demandées par le biais de la gestion de la DL sont également récapitulées en 6.4.3.

6.4.2 Attributs de la DL

Le Tableau 36 définit les attributs des supertrames.

Le Tableau 37 définit les attributs des liaisons.

Le Tableau 39 définit les attributs des graphes.

Clé de réseau utilisée pour le calcul du MIC, voir 6.2.4.2.3

D'autres attributs sont définis au Tableau 46.

Tableau 46 – Attributs de la DL

Nom d'attribut	Plage de valeurs	Définition
Keep_alive_time	0 à $2^{32}-1$ en unité de ms	Intervalle pendant lequel un nœud doit communiquer avec succès avec chaque voisin lié
Path_fail_time	0 à $2^{32}-1$ en unités de ms	Intervalle d'une communication ayant échoué avec un voisin donné, indiquant une défaillance de trajet
Advertise_time	0 à $2^{32}-1$ en unités de ms	Intervalle auquel les DLPDU Advertise sont émises
Discovery_time	0 à $2^{32}-1$ en unités de ms	Intervalle maximal entre deux émissions consécutives de DLPDU Advertise sur la liaison Discovery (découverte)
Priority_threshold	ALARM NORMAL PROCESS DATA COMMAND	Spécifie la DLPDU ayant la priorité la plus faible devant être acceptée en provenance d'un autre dispositif
Join_priority	0 à 15	Priorité utilisée dans la DLPDU Advertise envoyée par la DLE; un nombre inférieur indique un meilleur choix pour le rattachement
Promiscuous_mode	TRUE FALSE	Indique si, oui ou non, la DLE est en mode "receive all" (c'est-à-dire "recevoir tout"), à savoir la DLE accepte toutes les DLPDU reçues en provenance de la PhE
Max_backoff_exp	{4, 5, 6, 7}	La valeur maximale qui peut être supposée pour l'exposant de repli utilisé dans des intervalles de temps partagés, la valeur par défaut est 4

6.4.3 Action de la DLM

Les actions de la DLE que peut demander la gestion de la DL sont définies au Tableau 47.

Tableau 47 – Actions de DL

Nom	Description
RESET	Initialiser la DLE
DISCONNECT	Déconnecter la DLE du réseau et cesser toutes les communications
RE-JOIN	Déconnecter la DLE du réseau; purger toutes les demandes en attente de la part de l'utilisateur de DLS, vider tous les tampons et files d'attente et vider tous les tableaux; se rattacher de nouveau au réseau.

Nom	Description
DELETE-SUPERFRAME	Supprimer une supertrame existante et toutes les éventuelles liaisons associées
DELETE-LINK	Supprimer une liaison existante, en mettant éventuellement à jour les tableaux de voisins et de connexions associés à la liaison en question
DELETE-GRAPH	Supprime une connexion de graphe existante

6.5 Couche réseau

6.5.1 Mode à deux niveaux de NL

6.5.1.1 Généralités

La couche réseau est modélisée comme

- une fonction de sécurité de niveau supérieur, et
- une fonction de routage de niveau inférieur.

En plus, la fonction de gestion de NL opère avec les deux niveaux.

6.5.1.2 Fonction de sécurité

Un code d'intégrité de message (MIC) chiffré est utilisé pour l'authentification de la NPDU, le chiffrement et le déchiffrement de la charge utile de la NL. Les clés sont distribuées par le gestionnaire de réseau.

6.5.1.3 Fonction de routage

6.5.1.3.1 Généralités

La couche réseau fournit des services de routage. Elle gère des sessions pour des communications de bout en bout avec les dispositifs de correspondants. Elle reçoit la NPDU provenant de la DLE. Si la destination de la NPDU est l'utilisateur local du NLS, elle transfère les données d'utilisateur vers leur utilisateur. Sinon, elle transmet la NPDU destinée à d'autres dispositifs en utilisant la DLE. Elle traite également la demande "NL-TRANSMIT request" et prépare la NPDU en vue de l'émission.

La couche réseau de la présente norme achemine une NPDU en utilisant l'une de deux méthodes.

- Graph (graphe) – la topologie de réseau peut être représentée comme étant l'ensemble de tous les dispositifs et de toutes les liaisons orientées entre ceux-ci. Un graphe est un sous-ensemble des liaisons orientées et des dispositifs qui fournissent des chemins de communication redondants entre une source et un dispositif de destination. Le chemin effectivement pris dépend des conditions courantes du réseau lorsque la NPDU est acheminée de la source vers la destination. La NPDU contient le Graph_ID, mais ne spécifie pas le chemin complet. Par conséquent, chaque DLE intermédiaire dans le chemin doit être configurée avec un fragment du graphe total avant d'utiliser le chemin de graphe en question. Les chemins du graphe sont redondants et hautement fiables. Il convient d'utiliser cette méthode de routage pour des communications de routine normales, tant en amont qu'en aval de la passerelle.
- Source – il s'agit d'un chemin orienté simple (dispositifs et liaisons) entre une source et un dispositif de destination. Le chemin complet est contenu de manière statique dans la NPDU elle-même. Les DLE intermédiaires n'ont pas besoin de connaître le chemin à l'avance. À mesure que la NPDU est acheminée, chaque DLE intermédiaire propage la NPDU vers la DLE suivante en utilisant le chemin source contenu dans la NPDU. Les chemins source ne sont pas redondants et peuvent défaillir à tout moment. Des conditions courantes du réseau pourraient casser le chemin spécifié dans la NPDU et occasionner une perte de NPDU. Il convient d'utiliser cette méthode de routage uniquement pour

soumettre des chemins à l'essai, dépanner des trajets du réseau ou pour des communications ad-hoc telles que l'acheminement de réponses de rattachement.

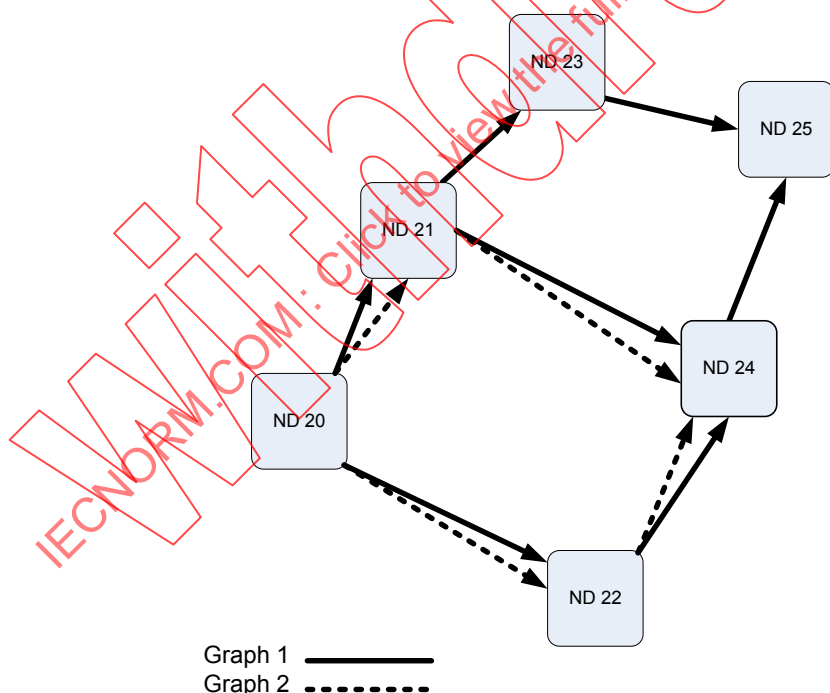
Tous les dispositifs sont tenus de prendre en charge tant le routage source que le routage graphe.

6.5.1.3.2 Routage par graphe

Le graphe est un ensemble de trajets orientés qui relient des points d'extrémité du réseau. Les trajets spécifiques associés à chaque graphe sont configurés explicitement par le gestionnaire de réseau dans tous les dispositifs du réseau. Une simple instance de réseau peut avoir plusieurs graphes, dont certains peuvent se chevaucher. Chaque dispositif de réseau peut avoir plusieurs graphes passant par lui, voire vers les mêmes voisins. Les graphes sont unidirectionnels.

Chaque graphe dans un réseau a un Graph_ID propre. Lors de l'utilisation du routage graphe, la NLE place la valeur du Graph_ID dans l'en-tête de la NPDU. La NLE recevant la NPDU la transmet le long de l'ensemble de trajets appartenant au graphe en question vers la destination.

Il convient que chaque Graph_ID dans le dispositif ait plusieurs voisins associés. Dans un réseau correctement configuré, tous les dispositifs ont au moins deux dispositifs voisins dans le graphe auxquels la NLE peut envoyer la NPDU. Cela assure la redondance dans le chemin et améliore la fiabilité. Une NLE examine le Graph_ID dans la NPDU reçue et transmet la NPDU à n'importe lequel des voisins énumérés dans le graphe en question.



Légende

Anglais	Français
Graph1	Graphe 1
Graph 2	Graphe 2

Figure 27 – Routage par graphe

Comme montré à la Figure 27, pour le graphe 1, ND 20 est la source et ND 25 est la destination; pour le graphe 2, ND 20 est la source et ND 24 est la destination. Pour envoyer

une NPDU en utilisant le graphe 1, ND 20 peut la transmettre soit à ND 21, soit à ND 22. La NPDU peut prendre plusieurs autres chemins, mais de toute manière, en suivant le graphe 1 elle aboutira à ND 25. Ainsi, une NPDU allant de ND 20 à ND 25 pourrait emprunter le chemin passant par ND 21 et ND 23. Une autre NPDU allant de ND 20 à ND 25 pourrait emprunter le chemin passant par ND 21 et ND 24. De même, pour communiquer avec ND 24, ND 20 envoie la NPDU en utilisant le graphe 2 passant soit par ND 21, soit par ND 22.

Une version simplifiée spéciale du routage par graphe, appelée routage de supertrame, est également prise en charge par tous les dispositifs. Lorsque le routage de supertrame est effectué, le Superframe_ID est placé dans la NPDU. Si le Graph_ID dans la NPDU reçue ne concorde avec aucun des Graph_ID configurés dans le dispositif de réception, la NLE recherche une supertrame ayant la même valeur. Si cela aboutit par un succès, la NLE transmet le paquet à n'importe quel voisin avec une liaison dans la supertrame en question. Dans le cas du routage de supertrame, les liaisons candidates sont sélectionnées en fonction de leur numéro de Superframe_ID. Dans le cas du routage par graphe, les liaisons candidates sont sélectionnées en fonction des entrées du tableau de graphe (Graph_ID et voisin).

6.5.1.3.3 Routage source

Pour le routage source, le chemin complet allant du dispositif source au dispositif de destination est spécifié dans la NPDU. À mesure que la NPDU est propagée, chaque NLE intermédiaire consulte la liste d'adresses du chemin de source et transmet la NPDU à la prochaine DLE de la liste. Le chemin doit être configuré dans la NLE, habituellement par le gestionnaire de réseau. Le routage source spécifie un seul trajet et, si l'une des liaisons intermédiaires a une défaillance, la DLPDU est perdue. Le routage source est moins fiable que le routage par graphe. Lorsqu'un chemin source a une défaillance, la NLE au point de la défaillance notifie le gestionnaire de réseau. Il est de la responsabilité du gestionnaire de réseau d'entreprendre une action corrective.

Comme montré à la Figure 28, le chemin de ND 20 à ND 24 passe par ND 21, ND 23 et ND 25. Le tableau de routage dans la NLE du ND 20 est configuré avec ce trajet. La NLE source ajoute la liste d'adresses pour les ND 21, ND 23 et ND 25 dans l'en-tête de la NPDU et envoie la NPDU au ND 21. La NLE dans le ND 21 reçoit la NPDU en question; examine l'en-tête et transmet la NPDU au ND 23. La NLE dans le ND 23 transmet la NPDU au ND 25; qui alors la transmet au ND 24. Le chemin allant de ND 20 à ND 24 pourrait aussi avoir été configuré via ND 22 et ND 25.

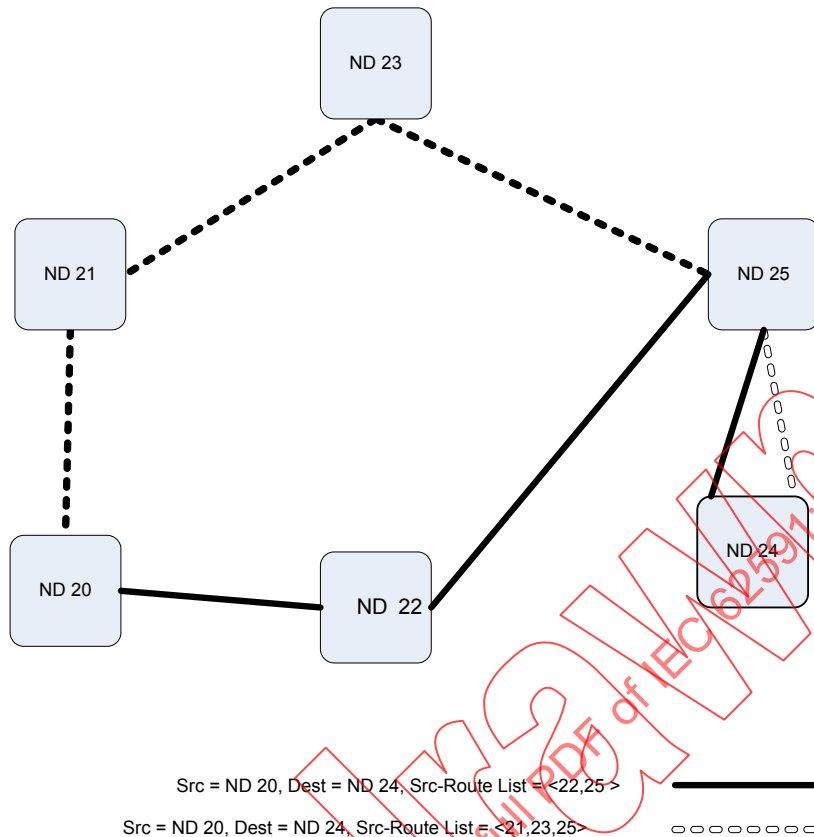


Figure 28 – Routage source

6.5.2 Structure de NPDU structure

6.5.2.1 Généralités

La Figure 29 montrera la structure d'une NPDU. Chaque NPDU doit être constituée des champs dans une séquence décrite ci-dessous:

- Le contrôle de NL ("NL Control") de 1 octet,
- Le compteur Hop-to-live (HTL) de 1 octet,
- Le numéro de séquence (Sequence number) de 2 octets,
- Le Graph ID (identificateur de graphe) de 2 octets,
- Les adresses de destination et source dans cet ordre, l'une ou l'autre pouvant avoir une longueur de 2 octets ou de 8 octets,
- Zéro ou plus d'octets de routage,
- Security control ("contrôle de sécurité") de 1 octet,
- Counter (compteur),
- Un code d'intégrité de message de 4 octets, et
- La charge utile de la NL.

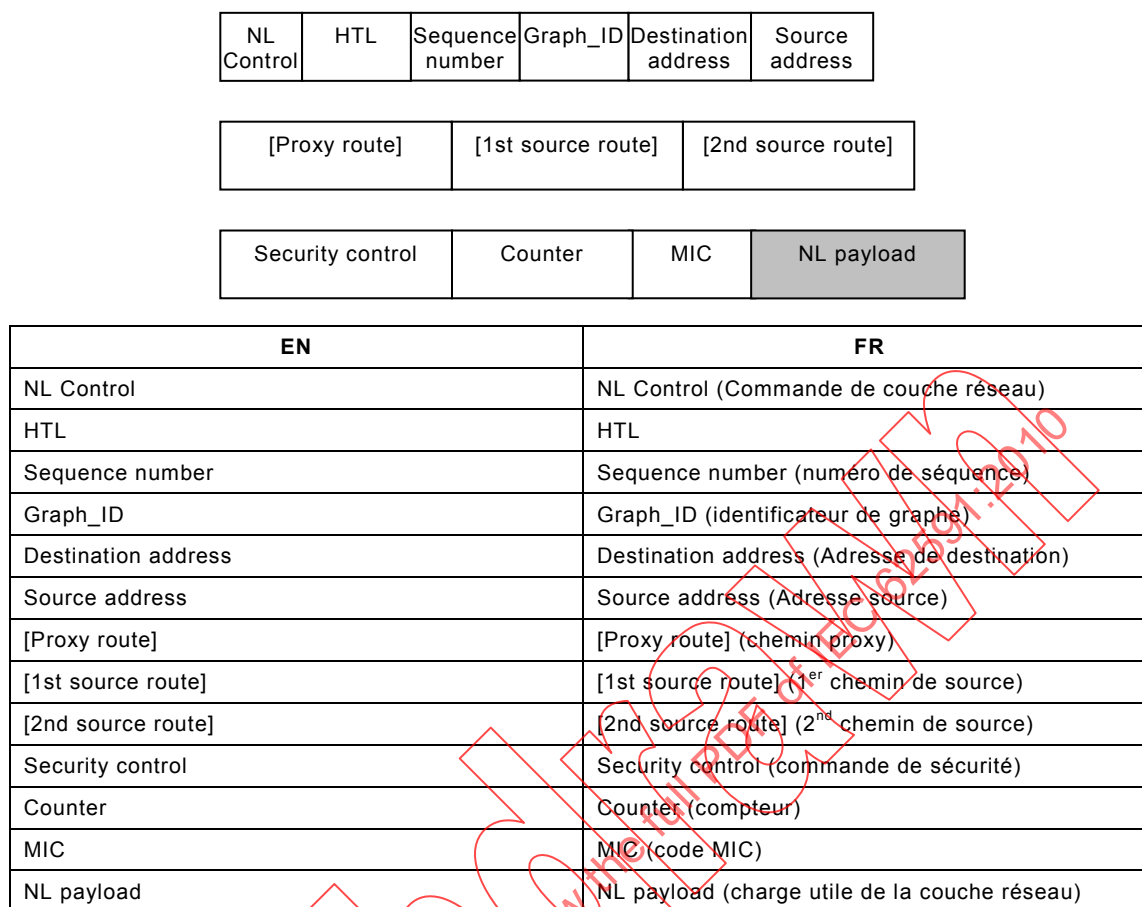
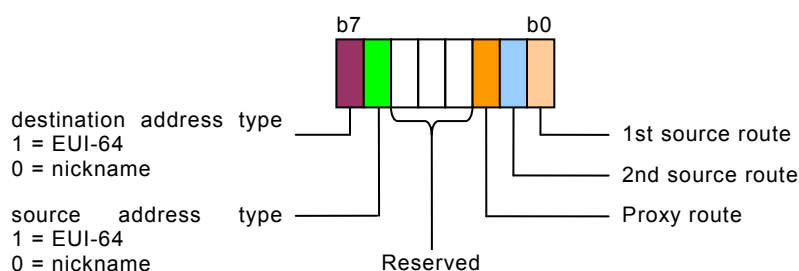


Figure 29 – Structure de NPDU

6.5.2.2 Octet de commande de NL

Ce champ spécifie la longueur et le type des adresses de destinations, source et la présence de champs de routage, comme montré à la Figure 30. Le bit 7 spécifie le type d'adresse de destination; le bit 6 spécifie le type d'adresse source; Si la valeur de l'un de ces deux bits est '1', l'adresse dans la NPDU correspondant au bit en question est EUI-64, autrement il s'agit d'un surnom (voir 6.2.1.2.4). N'importe quelle combinaison de longueurs d'adresse peut être utilisée dans une DLPDU.

Les bits 5, 4 et 3 sont réservés pour usage futur. La NLE doit mettre la valeur de ces bits à '000' avant l'émission. La NLE doit ignorer toute valeur de ces bits dans la NPDU reçue.



Légende

Anglais	Français
destination address type 1 = EUI-64 0 = nickname	Type d'adresse de destination 1 = EUI-64 0 = surnom
source address type 1 = EUI-64 0 = nickname	Type d'adresse source 1 = EUI-64 0 = surnom
Reserved	Réservé
1st source route	1 ^{er} chemin de source
2nd source route	2 nd chemin de source
Proxy route	Chemin de proxy
Reserved	Réservé

Figure 30 – Octet de commande de NPDU

Si la valeur de l'un des bits 2, 1 et 0 est '1', le champ de routage correspondant doit être présent dans la NPDU.

- Le bit 2 spécifie la présence d'un champ de routage Proxy de 2 octets de longueur.
- Le bit 1 spécifie la présence d'un 2nd champ de routage source de 8 octets de longueur.
- Le bit 0 spécifie la présence d'un 1er champ de routage source de 8 octets de longueur.

NOTE La NPDU peut spécifier simultanément un Graph_ID valide et un chemin source.

6.5.2.3 Hop-to-live

Ce champ tient le compte du nombre de sauts restants que la NPDU peut parcourir avant d'être rejetée.

6.5.2.4 Sequence number

Le numéro de séquence (Sequence Number) doit être fixé égal à l'octet de poids faible du Nombre d'intervalles absolus (ASN) au moment où la demande "NL-TRANSMIT request" est reçue. Ce champ donne une métrique de performance grossière mais critique et des informations de diagnostic relatives au fonctionnement du réseau. Il fournit aussi, lorsque l'ASN complet est recréé, l'âge du paquet.

6.5.2.5 Graph_ID

Le Graph_ID est utilisé pour acheminer le paquet vers sa destination finale. Le Graph_ID identifie indirectement une liste de dispositifs, n'importe lesquels de ceux-ci pouvant être utilisés pour transmettre la NPDU vers sa destination finale.

6.5.2.6 Destination and source addresses (Adresse de destination et adresse source)

La NL prend en charge deux types d'adresses: une adresse surnom de 2 octets et une adresse IEEE EUI-64 de 8 octets. Voir 6.2.1.2.4 pour la définition et les formats de ces adresses. Ces deux adresses ne sont pas modifiées pendant la propagation de la NPDU.

6.5.2.7 Proxy route (chemin Proxy)

Le routage proxy doit être utilisé pour communiquer avec les dispositifs qui n'ont pas encore été intégrés dans le réseau par le gestionnaire de réseau. Cela inclut les dispositifs qui sont en quarantaine ou sont en cours de rattachement. Le surnom de 2 octets du parent proxy pour la destination finale est indiqué dans l'adresse du Proxy.

6.5.2.8 Source route (chemin de source)

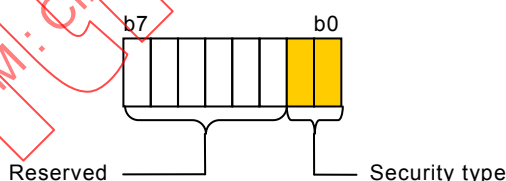
Chaque champ Source-route contient un jeu de quatre adresses qui désignent le chemin que le paquet doit suivre du dispositif source d'origine jusqu'à la destination finale du paquet. Chaque adresse est un surnom de 2 octets. Toute adresse non utilisée doit être mise à 0xFFFF.

Si le champ Source-route est présent, les adresses qu'il contient doivent être utilisées pour acheminer le paquet vers sa destination finale. Il convient d'utiliser les chemins source uniquement pour soumettre à essai ou dépanner les trajets du réseau.

6.5.2.9 Security control (contrôle de sécurité)

Ce champ spécifie la stratégie de sécurité utilisée pour la NPDU, comme montré à la Figure 31. Le sous-champ constitué des bits 1 à 0 doit être codé comme suit:

- 0: session chiffrée.
- 1: rattachement chiffré.
- 2: contenu maintenu chiffré.



Légende

Anglais	Français
Reserved	Réservé
Security type	Type de sécurité

Figure 31 – Octet de commande de sécurité

Les bits 7, 6, 5 et 4 sont réservés pour usage futur. La NLE doit mettre la valeur de ces bits à '0000' dans la NPDU à émettre. La NLE doit ignorer toute valeur de ces bits dans la NPDU reçue.

6.5.2.10 Counter

Ce champ doit être mis à la valeur pleine ou tronquée du compteur de nonces. La valeur et la longueur de ce champ dépendent du type de sécurité comme montré au Tableau 48.

Tableau 48 – Valeur du Compteur

Type de sécurité	Valeur du Compteur
Session	8 bits de poids faible du compteur de nonces
Join	32 bits pleins du compteur de nonces
Handheld	32 bits pleins du compteur de nonces

6.5.2.11 Code d'intégrité de message

Un code d'intégrité de message (MIC) chiffré est utilisé pour l'authentification de la NPDU et le déchiffrement de la charge utile de la sous-couche sécurité. Il a une longueur de 4 octets.

6.5.2.12 Charge utile de la NL

Les données d'utilisateur de NLS constituent la charge utile de la NL.

6.5.3 Fonction de sécurité

6.5.3.1 Généralités

La fonction de sécurité est fournie en diverses couches et au fonctionnement du dispositif global. L'Annexe A spécifie la fonctionnalité de sécurité des communications et les aspects sécuritaires associés.

6.5.3.2 Session

6.5.3.2.1 Types de session

La sécurité est orientée session. Une session permet une communication privée et sécurisée entre une paire de dispositifs de réseau. Quatre sessions sont établies dès qu'un dispositif quelconque rejoint le réseau.

- Session de monodiffusion entre le dispositif et le gestionnaire de réseau – elle est utilisée par le gestionnaire de réseau pour gérer le dispositif;
- Session de diffusion entre le dispositif et le gestionnaire de réseau – elle est utilisée pour gérer de manière globale les dispositifs. Par exemple, elle peut être utilisée pour rendre une nouvelle clé de réseau disponible à tous les dispositifs de réseau. Tous les dispositifs dans le réseau ont la même clé pour cette session.
- Monodiffusion entre le dispositif et la passerelle – elle porte des communications normales (par exemple, données de processus) entre la passerelle et le dispositif.
- Session de diffusion entre le dispositif et la passerelle – elle est utilisée par la passerelle pour envoyer à tous les dispositifs les données d'application identiques.

En outre, chaque dispositif a une session de rattachement et celle-ci ne peut pas être supprimée. La clé Join_key est unique; elle est la seule clé qui peut être écrite en se connectant directement au port de maintenance du dispositif. Elle peut aussi être écrite par le gestionnaire de réseau.

Des sessions supplémentaires peuvent être ajoutées, si un dispositif tenu à la main ou une autre passerelle rejoint le réseau. Le gestionnaire de réseau crée toutes les sessions et leur clé correspondante, sauf la join-session. Les sessions Join sessions doivent toujours être présentes dans le dispositif. Seule le gestionnaire de réseau doit créer ou modifier toutes les sessions, sauf la join-session. D'autres types de dispositifs ne sont pas permis.

Il est possible de déployer une session qui relie deux dispositifs arbitraires dans le réseau. Cependant, cela peut déboucher sur une communication non détectée et non surveillée entre les dispositifs et représente une menace à la sûreté et à la sécurité. Si le gestionnaire de réseau prend en charge des sessions entre homologues (peer-to-peer) entre des dispositifs de terrain, il convient que toutes les communications résultantes soient acheminées via la passerelle pour permettre la détection et l'interruption de comportements malveillants.

6.5.3.2.2 Tableau Session

Le Tableau 49 montre les attributs de la session. La combinaison des adresses de correspondants et du Session_type doit être unique, à savoir deux sessions ne sont pas autorisées à avoir des valeurs identiques pour adresses de correspondants et Session_type.

Lorsque et seulement lorsqu'une nouvelle session est créée, My_nonce_counter doit être remis à zéro. Chaque élément dans Nonce_counter_history doit être mis à la valeur du compteur de nonces dans la commande NLM-SET pour écrire les valeurs des attributs de session. Le dispositif n'est pas autorisé à changer la valeur de Session_key qui a été écrite par le gestionnaire de réseau. Le dispositif de réseau ne doit permettre à aucun moyen de lire une clé de session.

Tableau 49 – Attributs de Session

Nom d'attribut	Plage de valeurs	Définition
Session_type	UNICAST BROADCAST JOIN	Type de session
Peer_unique_ID	0 à $2^{40}-1$	ID propre au dispositif de correspondant comme défini en 3.3.91
Peer_nickname	0 à 65 534	Surnom du dispositif de correspondant comme défini en 3.3.67
Session_key	Nombre de 128 bits	Clé de la session
Peer_nonce_counter	Nombre de 32 bits	Valeur la plus élevée du compteur de nonces reçue en provenance du dispositif de correspondant.
Nonce_counter_history	matrice de 32 bits	Une matrice de bits enregistrant les compteurs de nonces reçus; le bit de poids fort est toujours mis et correspond à la valeur courante de Peer_nonce_counter. Le bit de poids faible correspond à '1 + Peer_nonce_counter – size of Nonce_counter_history'. Cela crée une fenêtre glissante avec chaque bit enregistrant une NPDU reçue avec le compteur de nonces correspondant. La taille de Nonce_counter_history est égale au nombre des NPDU reçues qui ont été enregistrées. Lorsque le Nonce_counter_history est décalé vers la droite, l'historique de la plus ancienne NPDU est éliminé.
My_nonce_counter	Nombre de 32 bits	Compteur de nonces pour les paquets fournis par les dispositifs
Transport_table	Référence interne	Référence aux tableaux de transport (normalement 2) associés au correspondant
Route_table	Référence interne	Référence aux tableaux de chemins associés à la destination de correspondant.

6.5.3.2.3 Join session (session de rattachement)

Join session est unique et est créée par le dispositif de réseau lorsque la Join_key est d'abord écrite via le port de maintenance. Les adresses de la 'join session' doivent être mises aux adresses de gestionnaire de réseau standard. L'entrée My_nonce_counter doit être remise à zéro lorsque la session est créée. Le My_nonce_counter du dispositif de réseau est utilisé tant pour les demandes de rattachement provenant du dispositif de réseau que pour les

réponses de rattachement provenant du gestionnaire de réseau. Pour se protéger des attaques de rejeu sur Join_key, My_nonce_counter doit être stocké dans une mémoire non volatile.

Plusieurs stratégies peuvent être utilisées pour créer la 'join session' sur le gestionnaire de réseau. Par exemple, une 'join session' pour un dispositif de terrain spécifique pourrait être créée en utilisant l'ID propre au dispositif de terrain et la Join_key. Quelle que soit la stratégie utilisée pour créer la 'join session', lorsque le gestionnaire de réseau reçoit et valide une demande de rattachement, le champ Counter de NPDU est copié dans les entrées de compteur de nonces de la session et est utilisé pour construire la réponse de rattachement.

6.5.3.3 Éléments de procédure de la fonction sécurité

6.5.3.3.1 Réception de la primitive "NL-Transmit request"

Lorsqu'une demande NL-TRANSMIT request est reçue, la NLE doit construire la partie en-tête de sécurité de la NPDU pour la demande en question. Un MIC chiffré est utilisé pour assurer que la NPDU arrive avec succès et sans être violée en provenance du dispositif source indiqué. Le MIC est généré et confirmé en utilisant le mode CCM* (compteur combiné avec CBC-MAC, cipher block chaining-message authentication code (corrigé)) en association avec le chiffre à bloc de 128 bits de l'AES³ pour fournir l'authentification comme spécifié dans l'IEEE 802.15.4-2006, Annexe B. Ce chiffre exige quatre chaînes d'octets comme paramètres:

- 'a', les données complémentaires devant être authentifiées mais pas chiffrées;
- 'm', le message devant être chiffré;
- 'N', le nonce de 13 octets; et
- 'K', la clé AES de 128 bits.

La charge utile de la NL est chiffrée et est la chaîne d'octets 'm'. L'en-tête de la NPDU, à partir de l'octet de commande de la NPDU jusqu'au MIC, est la chaîne d'octets 'a'. Les champs HTL, Counter et MIC dans la chaîne d'octets 'a' sont mis à zéro lors de l'authentification de la NPDU. Ils sont remplacés par leurs valeurs effectives avant que le paquet ne soit émis.

Le nonce de la couche sécurité (la chaîne d'octets 'N') a une longueur de 13 octets. Le nonce est créé comme résumé au Tableau 50 et décrit ci-dessous.

Si le Session_type est "join" et la primitive de demande "NL-TRANSMIT request primitive" est une réponse de rattachement (join response)

a) alors

- 1) le LSB (octet numéro zéro) de 'N' doit être mis à '1';
- 2) les quatre octets suivants de 'N' doivent être mis à la valeur de Peer_nonce_counter issue de la demande de rattachement correspondante en commençant par le MSB de Peer_nonce_counter;
- 3) les huit octets suivants de 'N' doivent être mis à l'adresse EUI-64 du dispositif candidat au rattachement;

NOTE La demande et la réponse de rattachement utilisent toujours l'adresse EUI-64 car le dispositif ne reçoit son surnom qu'après le rattachement réussi du dispositif au réseau. La réponse de rattachement ne peut être émise que du gestionnaire de réseau.

b) sinon,

- 1) le LSB (octet numéro zéro) de 'N' doit être mis à '0';
- 2) le My_nonce_counter doit être préincrémenté de 1;

³ AES = Advanced Encryption Standard

- 3) les quatre octets suivants de 'N' doivent être mis à la valeur de My_nonce_counter en commençant par le MSB de My_nonce_counter;
- 4) les cinq octets suivants de 'N' doivent être mis au champ adresse de source dans la NPDU, qui est soit l'adresse EUI-64, soit le surnom avec des zéros de bourrage.

Tableau 50 – Nonce de NPDU

Numéro d'octet	Description
0	Mis à 1 si réponse de rattachement, sinon mis à 0
1 – 4	Compteur de nonces commençant par le MSB en N[1]
Pour l'adresse de source EUI-64:	
5 – 12	L'adresse EUI-64 est copiée (de MSB à LSB) dans les octets N[5] à N[12]
Pour le surnom de deux octets:	
5 – 10	Chaque octet est mis à 0x00
11 – 12	Le surnom est copié (de MSB à LSB) dans les octets N[11] et N[12]

Après que le nonce est construit, la NPDU est chiffrée; et ensuite les champs de sécurité de la NPDU sont remplis.

6.5.3.3.2 Réception de la NPDU adressée au dispositif

6.5.3.3.2.1 Généralités

La NLE doit traiter la NPDU reçue en utilisant le moteur de chiffrement AES de 128 bits pour authentifier la NPDU et déchiffrer la charge utile. Ce chiffre exige quatre chaînes d'octets comme paramètres:

- 'a', qui est l'en-tête de la PDU, à partir de l'octet de commande de la NPDU jusqu'au MIC de la NPDU avec les champs HTL, Counter et MIC de la NPDU mis à zéro;
- 'm', qui est la charge utile de la NL;
- 'N', qui est le nonce de 13 octets; et
- 'K', qui est la clé AES de 128 bits.

L'authentification et le déchiffrement commencent par localiser la session correcte (voir 6.5.2.9) pour déterminer la clé et les compteurs de nonces.

Si la NPDU reçue n'est pas rejetée, la charge utile doit être transmise à l'utilisateur de NLS dans la primitive d'indication "in NL-Transmit indication". La charge utile de la NL doit être constituée de NLS-user-data (données d'utilisateur de NLS).

6.5.3.3.2.2 Reconstruction de nonce

La NLE doit reconstruire le nonce pour la NPDU reçue, comme résumé au Tableau 50 et décrit ci-dessous.

Si le Session_type est "join"

a) alors

- 1) Si la NPDU reçue est une demande de rattachement, alors
 - i) le LSB (octet numéro zéro) de 'N' doit être mis à '0';
 - ii) les quatre octets suivants de 'N' doivent être mis au champ Counter de la NPDU reçue en commençant par le MSB de Counter;

NOTE La NPDU de demande de rattachement est reçue uniquement par le gestionnaire de réseau.

- 2) sinon,
 - i) le LSB (octet numéro zéro) de 'N' doit être mis à '1';
 - ii) si le champ Counter de la NPDU reçue est égal à My_nonce_counter, les quatre octets suivants de 'N' doivent être mis à la valeur de My_nonce_counter en commençant par le MSB de My_nonce_counter;
 - iii) sinon, la NLE doit rejeter la NPDU reçue.
 - 3) Si la NPDU n'a pas été rejetée, les cinq octets suivants de 'N' doivent être mis à l'adresse EUI-64 de la source provenant de la NPDU reçue;
- b) sinon,
- 1) le LSB (octet numéro zéro) de 'N' doit être mis à '0';
 - 2) les trois octets suivants de 'N' doivent être mis aux trois octets de poids fort de la valeur de Peer_nonce_counter correspondante en commençant par le MSB de Peer_nonce_counter;
 - 3) si la valeur de Counter de 1 octet de la NPDU reçue est inférieure à la quantité $(1 + \text{LSB of Peer\_nonce\_counter} - \text{size of Nonce\_counter\_history})$, la valeur 24 bits dans les octets N[1] à N[3] doit être incrémentée;
 - 4) l'octet suivant de 'N' doit être mis au champ Counter de la NPDU reçue;
 - 5) les cinq octets suivants de 'N' doivent être mis au champ adresse de source de la NPDU reçue, qui est soit l'adresse EUI-64, soit le surnom avec des zéros de bourrage.

6.5.3.3.2.3 Authentification

Si le nonce est construit avec succès et la NPDU reçue n'est pas un doublon d'une NPDU reçue précédemment, l'authentification et le déchiffrement doivent être effectués. Si cette authentification échoue, le paquet doit être rejeté.

6.5.3.3.2.4 Protection contre le jeu

Si le paquet est authentique, le Nonce_counter_history doit être mis à jour. Sachant que la NPDU peut arriver en désordre ou être complètement perdue, cet algorithme de fenêtre glissante est utilisé pour faciliter les communications et éliminer les paquets dupliqués. Le Peer_nonce_counter et le Nonce_counter_history font ensemble la chronique des NPDU reçues par le dispositif.

Les octets 1 à 4 de 'N' constituent le compteur de nonces reconstitués. Le compteur de nonces reconstitués doit être comparé au Nonce_counter_history. S'il correspond à n'importe lesquels des bits mis dans le Nonce_counter_history, la NPDU reçue est une duplication de NPDU reçue précédemment et la NPDU doit être rejetée. Si le compteur de nonces reconstitués se situe à l'extérieur de la fenêtre du Nonce_counter_history, la DLE doit rejeter la NPDU reçue.

Si le compteur de nonces reconstitués est inférieur au Peer_nonce_counter, le bit approprié dans le Nonce_counter_history doit être mis. Autrement, le Nonce_counter_history doit être décalé vers la droite et la valeur du Peer_nonce_counter mise à jour. Cette opération s'effectue en soustrayant le Peer_nonce_counter du compteur de nonces reconstitués. La différence obtenue indique le nombre de fois que le Nonce_counter_history doit être décalé vers la droite. Après le décalage, le bit de poids fort du Nonce_counter_history doit être mis. Enfin, le compteur de nonces reconstitués doit être copié vers le Peer_nonce_counter dans l'entrée du tableau de session.

6.5.4 Tableaux de la couche réseau

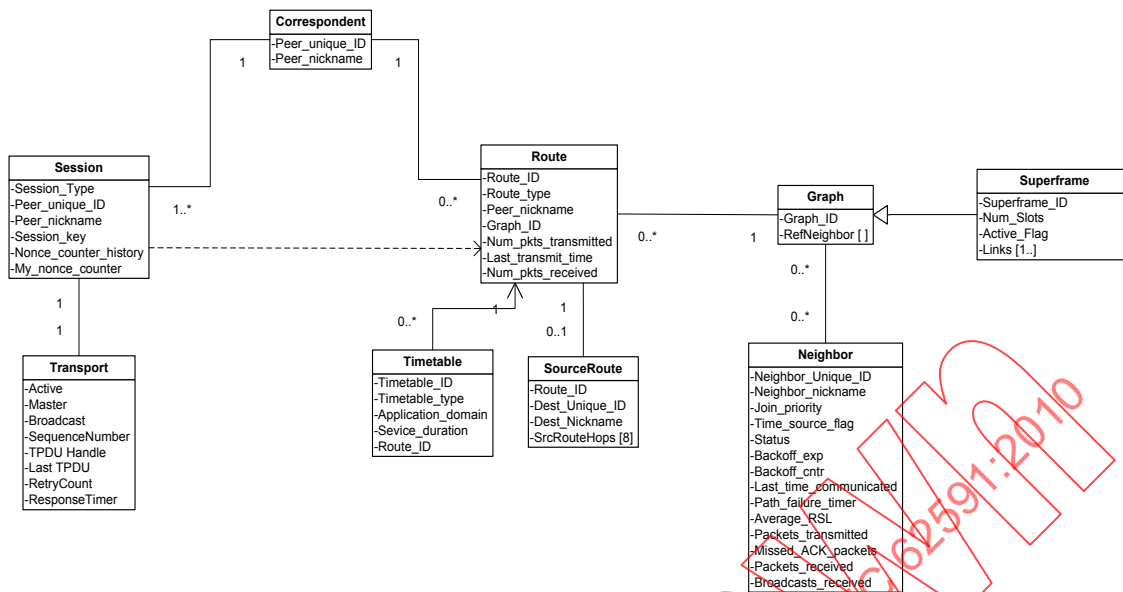
6.5.4.1 Généralités

Tous les dispositifs doivent maintenir une série de tableaux qui commandent les communications effectuées par le dispositif, fournir des informations de routage, prendre en charge les acquittements de bout en bout, assurer la confidentialité des communications et rassembler des statistiques sur ces communications.

Les tableaux commandant les activités de communication sont:

- Tableau Session – une entrée dans ce tableau établit un tube sécurisé entre le dispositif et un dispositif de correspondant spécifique. Aucune communication vers un dispositif de correspondant n'est possible sans une session de sécurité (voir 6.5.3.2.1). Voir 6.5.3.2.2 pour les attributs de session.
- Tableau de transport – il est utilisé pour prendre en charge la transaction d'acquittements de bout en bout avec nouvelles tentatives automatiques (voir 6.6.3).
- Tableau de chemins – il sert de lieu pour ajouter des informations de routage à une nouvelle NPDU générée comme le résultat d'une demande "NL-TRANSMIT request" (voir 6.5.4.2).
- Tableau de chemin source – il est joint à certains chemins (voir 6.5.4.3).
- Tableau Service – il indique le chemin (Route) associé à un service alloué par le gestionnaire de réseau (voir 6.5.4.4).

La relation entre ces tableaux est montrée à la Figure 32. Au sein du dispositif, le tableau session est central. La NLE utilise aussi le tableau de voisins Neighbor (voir 6.3.3.4) et le tableau Graph (voir 6.3.3.5) de la DLE. La figure montre les relations autour du chemin. Un chemin est identifié par un route ID. Un chemin est associé à un correspondant spécifique. Un chemin peut contenir un chemin source facultatif. Un chemin est toujours associé à un graphe. Dans certains cas, la forme spécialisée de graphe, "superframe", est utilisée. Un graphe peut être associé à 0 (supertrame) voisin ou plus. Un Timetable (indicateur horaire) est toujours associé à un seul chemin. Un chemin est associé à 0 Timetable ou plus. La relation avec les Timetables est orientée, car le Timetable a des informations sur le chemin mais le chemin peut ne pas en avoir sur le Timetable.



Légende

Anglais	Français
Correspondent	Correspondent (correspondant)
Route	Route (chemin)
Graph	Graph (graphe)
Superframe	Superframe (supertrame)
Session	Session
TimeTable	TimeTable (indicateur horaire)
Transport	Transport
SourceRoute	SourceRoute (chemin de source)
Neighbor	Neighbor (voisin)

Figure 32 – Tableaux de NLE et relation

Les dispositifs doivent prendre en charge le nombre minimum d'entrées de tableau et l'espace de tampon minimum montré au Tableau 51.

Tableau 51 – Exigence d'espace du au nombre minimum des NLE

Nom du tableau	Nombre minimum d'entrées requis
Session	8
Transport	16 ^a
Route	8
Source-route	2
Service	16
^a 2 par session, si la session est supérieur à 8.	

NOTE Certains des attributs de ces tableaux décrits dans la présente norme peuvent être calculés ou dérivés d'autres d'informations et il n'est pas nécessaire de les stocker séparément. Bien que la mise en œuvre spécifique du stockage de données et de configuration soit laissée aux concepteurs du dispositif, une description générale des champs contenus dans les structures de données est importante pour le fonctionnement global.

6.5.4.2 Tableau de chemins Route

Le Tableau 52 montre les attributs de chemin. Le tableau des chemins est associé à une session de sécurité et à une adresse de correspondant. Il est utilisé pour aider à sélectionner le graphe utilisé pour la communication vers le dispositif de correspondant. La NLE doit rassembler des statistiques pour permettre au gestionnaire de réseau d'optimiser les ressources de communication au fil du temps.

Tableau 52 – Attributs de chemin

Nom d'attribut	Plage de valeurs	Définition
Route_ID	0 à 255	Identificateur d'itinéraire
Route_type	MAINTENANCE PUBLISH BLOCK TRANSFER EVENT	indique le type de chemin; MAINTENANCE est la valeur par défaut.
Dest_nickname	0 à 65 534 – voir 6.2.1.2.4	Surnom du Dispositif de destination avec lequel le chemin est établi
Graph_ID	256 à 65 535	Identificateur pour le graphe associé avec ce chemin
Src_route_ID	0 à 255	Si non nul, l'identificateur du tableau chemin de source, sinon absence de chemin de source pour ce chemin
Num_pkts_transmitted	0 à 65 535 ^a	Nombre de paquets émis sur ce chemin
Last_transmit_time	0 à 2 ³² -1 dans les unités de 1/32 ms	L'heure à laquelle la dernière NPDU a été générée pour ce chemin
Num_pkts_received	0 à 65 535 ^a	Nombre de paquets reçus sur ce chemin mis à jour en fonction du Graph_ID dans la NPDU
a La valeur des compteurs de paquets repasse de 65 535 à 0.		

6.5.4.3 Tableau de chemins de source

Le tableau chemins de source contient une liste de 8 adresses de dispositif au maximum traçant le chemin allant de ce dispositif au dispositif de correspondant comme montré au Tableau 53.

Tableau 53 – Attributs de chemins de source

Nom d'attribut	Plage de valeurs	Définition
Route_ID	1 à 255	Identificateur d'itinéraire
dest_Unique_ID	0 à 2 ⁴⁰ -1	ID propre au dispositif de destination comme défini en 3.3.91
dest_Nickname	0 à 65 534	L'adresse courte du voisin est celle définie en 3.3.67
Src_route_hops [8]	Device_Nickname	jusqu'à 8 surnoms peuvent conduire à la destination Les adresses non utilisées doivent être mises au surnom de la destination.

6.5.4.4 Tableau d'indicateur horaire Timetable

Lorsqu'un dispositif a besoin d'établir une connexion de communication avec un autre dispositif, il envoie une demande au gestionnaire de réseau pour allouer un Timetable pour la communication en question. Chaque ensemble destination et Application_domain a son propre Timetable. Le Tableau 54 montre les attributs d'indicateur horaire Timetable. Il identifie le chemin conduisant au dispositif de destination et les quotas de programmation alloués pour cette communication. Lorsque le dispositif supprime un Timetable, le gestionnaire de réseau est censé désallouer les ressources réseau prenant en charge le Timetable en question. Le même chemin peut être utilisé par plus d'un Timetable.

Tableau 54 – Attributs de la table des temps

Nom d'attribut	Plage de valeurs	Définition
Timetable_ID	0 à 255	Identificateur de Timetable
Timetable_type	SOURCE SINK INTERMITTENT	Indique le type de Timetable
Application_domain	MAINTENANCE PUBLISH BLOCK TRANSFER EVENT	Le domaine d'application utilisant ce Timetable
Dest_nickname	0 à 65 534	L'adresse courte de la destination pour laquelle ce Timetable a été alloué
Service_duration	0 à $2^{32}-1$ dans les unités de 1/32 ms	Pour les communications cycliques (par exemple, Publish data), les paquets sont générés avec cette période. La latence pour la communication cyclique ne doit pas dépasser 1/3 de la période. Pour les communications intermittentes, il s'agit de la latence maximale requise.
Route_ID	0 à 255	Identificateur de chemin pour ce Timetable

6.5.5 Machine NLE

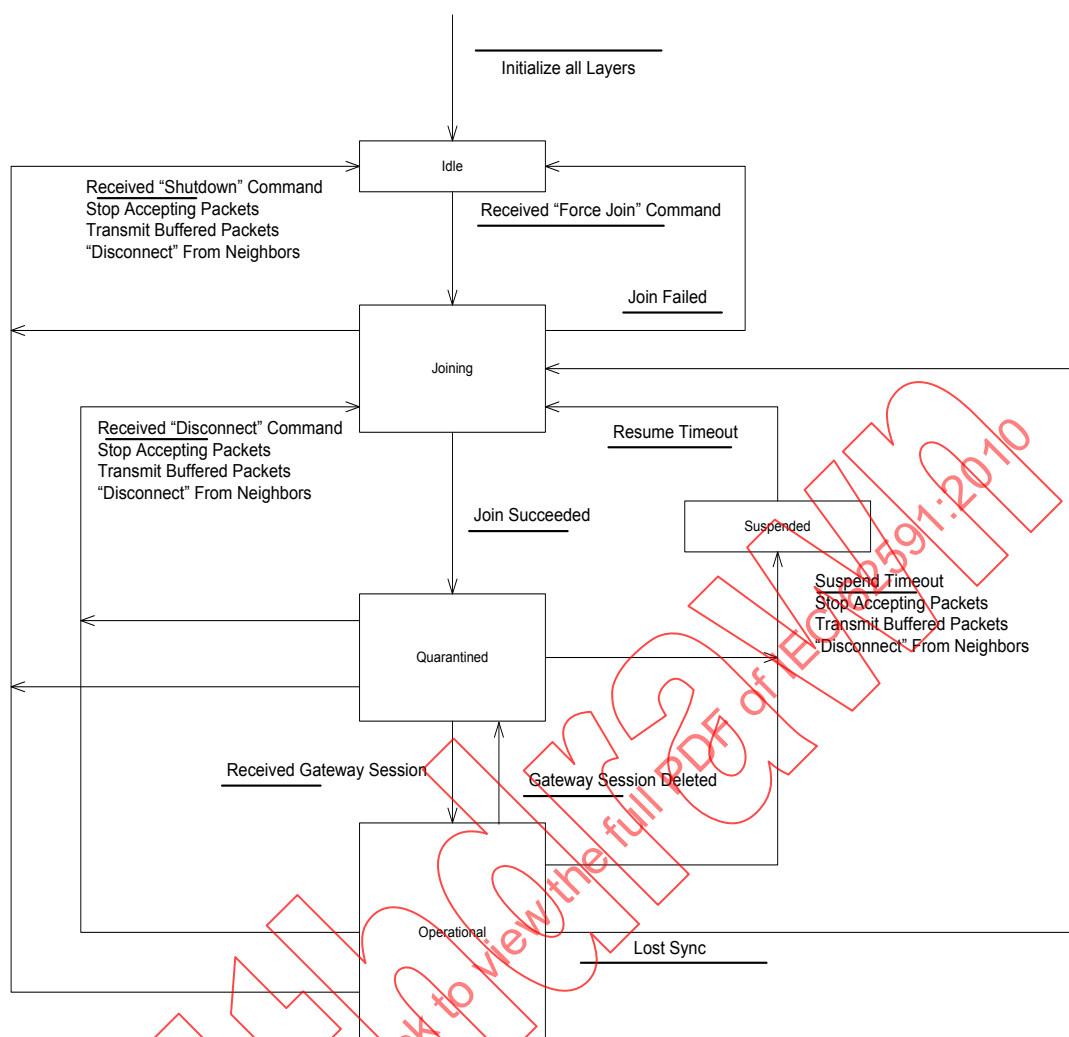
6.5.5.1 Fonctionnement général

La NLE avance par une série d'états commençant par l'état 'Idle' et continuant jusqu'à ce que le dispositif soit 'Operational' et un plein participant dans le réseau maillé. Il existe cinq états qui sont décrits au Tableau 55.

Tableau 55 – États de NLE

État	Description
Idle (au repos)	Le dispositif est au repos et son émetteur-récepteur sans fil n'est pas actif. Il n'a pas connaissance du réseau.
Joining (en cours de rattachement)	Le dispositif est à l'écoute pour déceler le réseau, tentant d'acquérir une DLPDU Advertise et demandant l'admission au réseau.
Quarantined (en quarantaine)	Le dispositif a rejoint avec succès le réseau, mais n'a qu'une autorisation de sécurité de parler au gestionnaire de réseau. Il n'est ni disponible ni autorisé à accomplir une acquisition de données ou des fonctions de commande ou autrement à communiquer avec la passerelle.
Operational (opérationnel)	Le dispositif est accessible à des applications d'hôtes via la passerelle. Il est intégré dans le fonctionnement du système.
Suspended (suspendu)	Le dispositif est au repos. Tous ces tableaux de réseau sont intacts.

Le diagramme des transitions d'états est montré à la Figure 33. Ce diagramme décrit les événements qui permettent au dispositif de démarrer, localiser et rejoindre le réseau et finalement progression de son statut de nouveau membre du réseau à son statut de dispositif de réseau totalement opérationnel. En outre, le diagramme montre le dispositif (et le réseau) suspendu et resynchronisé. De surcroît, le dispositif peut à tout instant être forcé de se déconnecter (par exemple d'être retiré du processus pour la maintenance au niveau dépôt) ou de se rattacher de nouveau.



Légende

Anglais	Français
Initialize all layers	Initialiser toutes les couches
Idle	Repos
Joining	En rattachement
Suspended	Suspendu
Quarantined	En quarantaine
Operational	Opérationnel
Received "Shutdown"...Neighbors	A reçu la commande d'arrêt ("Shutdown") Cesser d'accepter de paquets Émettre les paquets "bufférisés" Se déconnecter des voisins
Received "Force Join" Command	A reçu la commande de rattachement de force ("Force Join")
Join Failed	Échec du rattachement
Received "Disconnect"...Neighbors	A reçu la commande de déconnexion ("Disconnect") Cesser d'accepter de paquets Émettre les paquets "bufférisés" Se déconnecter des voisins

Anglais	Français
Suspend Timeout...Neighbors	Temporisation de suspension Cesser d'accepter de paquets Émettre les paquets "bufférisés" Se déconnecter des voisins
Received Gateway Session	A reçu une session de passerelle
Gateway Session Deleted	Session de passerelle supprimée
Lost Sync	A perdu le synchronisme
Resume Timeout	Reprendre Temporisation
Join succeeded	Rattachement réussi

Figure 33 – Diagramme d'état NLE

6.5.5.2 État "Idle"

Après réinitialisation d'un dispositif ou à la mise sous tension, le dispositif doit entrer dans l'état 'Idle'. Pendant qu'il est dans cet état, le dispositif ne doit pas tenter d'établir de communications sans fil. Le dispositif doit mettre à jour la Join_key et le network_ID, s'ils sont écrits via le port de maintenance. Lorsque le dispositif reçoit la commande Join provenant de la gestion de NL, la NLE doit passer à l'état 'Joining'.

6.5.5.3 État "Joining"

La NLE doit localiser le réseau et se synchroniser au réseau. Elle doit attendre de recevoir des paquets Advertise. Après avoir reçu un ou plusieurs paquets Advertise, la NLE doit demander l'admission au réseau, en utilisant la Join_Key. Le gestionnaire de réseau est censé envoyer la clé réseau et la session du gestionnaire de réseau à la NLE demandeuse. Le gestionnaire de réseau est également censé fournir à la DLE une supertrame, des graphes et des liaisons. Après que la NLE a reçu ces paramètres, elle doit passer à l'état 'Quarantined'.

La demande Join peut recevoir immédiatement une réponse de la part du gestionnaire de réseau ou bien la NLE peut avoir à répéter la demande. Un échec doit se produire si le dispositif envoie le nombre de demandes de rattachements égal à max_join_retries (voir Tableau 58), sans recevoir les clés de sécurité et les paramètres de session de la part du gestionnaire de réseau. Dans ce cas, la NLE doit retourner à l'état 'Idle'.

Si la NLE reçoit la commande Disconnect provenant de la gestion de NL, elle doit passer à l'état 'Idle'.

6.5.5.4 État "Quarantined"

Les dispositifs en quarantaine sont des partenaires réseau limités. La NLE doit rester dans cet état jusqu'à ce qu'elle soit approuvée pour le déploiement dans l'application réseau. L'approbation peut avoir la forme d'une autorisation d'utiliser le dispositif dans le système par l'opérateur humain et d'une mise en service du dispositif par celui-ci. Cette étape de quarantaine correspond à des pratiques normales dans l'industrie des procédés et ajoute une couche supplémentaire de sécurité au processus de rattachement.

Pendant qu'il est en quarantaine, le dispositif doit être seulement capable de servir de source et de puits pour les paquets, mais pas de transmettre le moindre paquet. Des rapports réseau normaux (par exemple des rapports de voisins) doivent être générés.

Après avoir reçu les paramètres de session de la passerelle, la NLE doit passer à l'état 'Operational'.

6.5.5.5 Operational

Cela permet au dispositif d'accéder à l'application d'hôte et lance son interaction avec le dispositif.

Dans l'état opérationnel, le dispositif est un partenaire réseau complet et doit commencer à accomplir les tâches désignées par sa configuration. Il doit acheminer les paquets à travers le réseau vers leur destination finale. Le dispositif doit aussi participer dans l'annonce et prendre en charge les nouveaux dispositifs rejoignant le réseau. S'il est configuré pour éditer des données, il doit demander la bande passante pour ce faire. Il doit recevoir des demandes et réponses d'utilisateur pour l'émission

6.5.5.6 État "suspended"

Lorsqu'il est à l'état 'Quarantined' ou 'Operational', le dispositif peut être mis de force à l'état 'Suspended' pendant la durée spécifiée. Il s'agit normalement d'une opération de brève durée qui est faite par exemple pour des raisons de sécurité pendant des opérations de sautage de mines. Dans cet état, le dispositif continue de suivre le temps écoulé tout en laissant la radio désactivée. Après la durée spécifiée, le dispositif doit passer à l'état 'Joining'.

6.5.6 Éléments de procédure NLE

6.5.6.1 Réception d'une primitive "NL-TRANSMIT request"

Lorsqu'une demande "NL-TRANSMIT request" est reçue de l'utilisateur de NLS, la NLE doit utiliser le Timetable_ID reçu dans la primitive de service pour localiser le Timetable pour cette demande. Si la destination n'est pas le gestionnaire de réseau et si la bande passante restante dans le Timetable est insuffisante pour cette demande, une erreur doit être retournée et la demande doit être supprimée.

La NLE doit construire la NPDU pour la demande en question. Le champ doit être mis conformément aux procédures indiquées au Tableau 56. La NLE doit trouver le Route_ID dans l'attribut de Timetable comme montré au Tableau 54. Une fois le chemin identifié, le Graph_ID trouvé dans l'attribut de Route, comme montré au Tableau 52, doit être copié dans la NPDU.

Les paquets doivent être acheminés par graphe, à moins que le chemin ne soit en cours d'essai. Cependant, si un source-route est attaché au chemin, le chemin de source doit être inclus dans la NPDU. Les entrées d'adresse de chemin de source non utilisées dans la NPDU doivent être mises à 0xFFFF. Si l'acheminement source est utilisé, le Graph_ID doit aussi être valide.

Tableau 56 – Construction de la NPDU

Champ	Procédure
Control	Mettre les bits d'adresse en fonction des tailles des adresses source et de destination. Les adresses de destination sont indiquées dans la primitive "request". Si la destination est une adresse de diffusion, la taille de l'adresse de destination doit être courte. L'adresse source doit être l'adresse EUI-64 du dispositif jusqu'à ce que le surnom du dispositif soit initialisé. Mettre les bits de routage en fonction des résultats de la sélection de chemins.
HTL	Mettre à Default_HTL.
Sequence number	Mettre à la valeur ASN courante.
Graph_ID	Mettre en fonction des résultats de la sélection de chemin.
Destination address	Mettre à la valeur contenue dans la primitive "request". Si l'adresse de destination est une adresse de diffusion, la valeur de ce champ doit être 0xFFFF.
Source address	Mettre à l'adresse EUI-64 si le surnom n'est pas initialisé, sinon mettre au surnom.
Proxy route	Mettre en fonction des résultats de la sélection de chemin.

Champ	Procédure
Source route(s)	Mettre en fonction des résultats de la sélection de chemin.

6.5.6.2 Réception d'une primitive d'indication "DL-RECEIVE indication"

La NLE doit examiner l'adresse de destination dans la NPDU reçue en provenance de la DLE. Elle doit acheminer ces paquets vers la destination correcte: transmettre à la DLE suivante dans le chemin, transmettre au dispositif candidat au rattachement ou livrer à l'utilisateur de NLS local.

6.5.6.3 Réception d'une primitive de confirmation "DL-TRANSMIT confirm"

6.5.6.3.1 Généralités

La NLE doit supprimer toutes les données et tous les éléments de tableau pour le service associé au descripteur et retourner une confirmation à l'utilisateur de la NL.

6.5.6.3.2 NPDU adressée au dispositif

Si la destination indiquée dans la NPDU reçue est l'utilisateur de NLS local, la NLE doit la transmettre à la fonction de sécurité.

6.5.6.3.3 NPDU avec adresse de diffusion

Si l'adresse de destination contenue dans la NPDU reçue est l'adresse de diffusion, la NLE doit transmettre la NPDU comme décrit en 6.5.6.3.4 et la transmettre aussi à la fonction de sécurité.

6.5.6.3.4 Transmission de la NPDU reçue

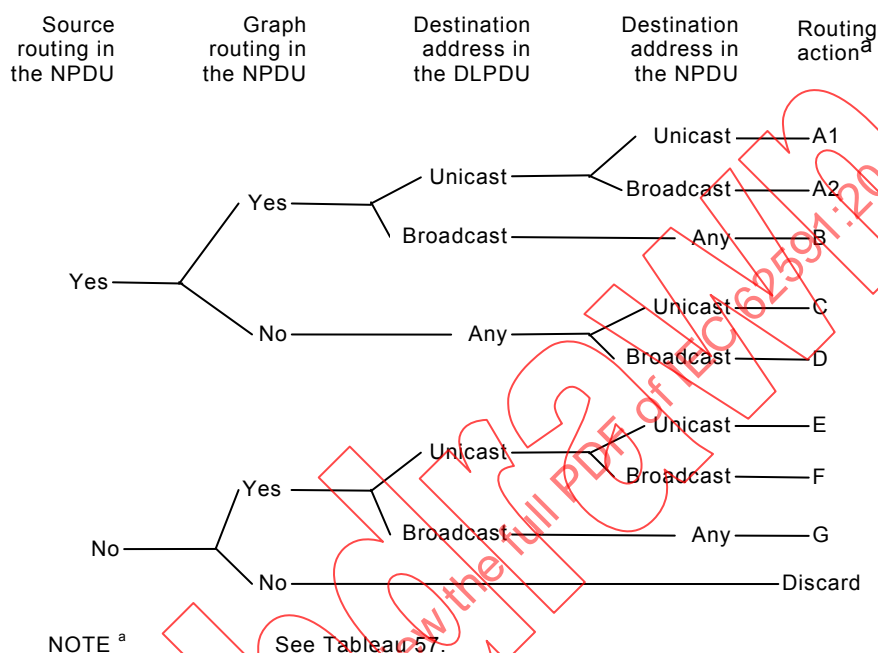
La NLE doit traiter la NPDU reçue dans la séquence décrite ci-dessous.

La NLE doit vérifier et mettre la jour le HTL. Si le compte des HTL dans la NPDU reçue est zéro,

- a) alors la NPDU doit être rejetée;
- b) sinon,
 - 1) si le HTL n'est pas 0xFF, la NLE doit la décrémenter de 1 et utiliser cette valeur pour construire la NPDU devant être transmise;
 - 2) le numéro de séquence doit être examiné et, si l'âge de la NPDU reçue est supérieur à max_packet_age, elle doit être rejetée;
 - 3) sinon, la NLE doit vérifier l'adresse de destination;
 - 4) si la NPDU contient une adresse proxy qui concorde avec l'adresse de NLE locale, la destination finale est un dispositif candidat au rattachement. La NLE doit acheminer le paquet vers le dispositif candidat au rattachement en transmettant la NPDU à la DLE indiquant comme adresse de destination de DLL celle du dispositif candidat au rattachement;
 - 5) sinon, si l'adresse de destination de la NPDU est Unicast (monodiffusion) et correspond à un voisin, la NPDU doit être acheminée directement vers le voisin en question. Le dispositif doit acheminer le paquet vers le dispositif voisin en transmettant la NPDU à la DLE indiquant comme adresse de destination de DLL celle du dispositif voisin;
 - 6) sinon le paquet doit être transmis en fonction du Graph_ID et du "Source route" contenus dans la NPDU. Les décisions de routage sont montrées à la Figure 34. Lors de l'analyse d'une NPDU reçue, les colonnes dans la figure doivent être évaluées de gauche à droite, c'est-à-dire en commençant par "Source routing in the NPDU"

(Routing source dans la NPDU)". Un "Non" dans la colonne "Graph routing in the NPDU" (Routing par graphe dans la NPDU) signifie que le Graph_ID reçu ne concorde pas soit avec un Graph_ID, soit avec un Superframe_ID trouvé dans le dispositif de terrain qui a reçu la NPDU. La décision se termine par un identificateur littéral pour la décision de routage, qui est décrite au Tableau 57.

NOTE Si l'adresse de destination contenue dans la DLPDU reçue est une adresse de diffusion, le Graph_ID dans la NPDU en question détient le Superframe_ID pour identifier les liaisons de diffusion correspondantes.



Légende

Anglais	Français
Sourcer routing in the NPDU	Routage de source dans la NPDU
Graph routing in the NPDU	Routage par graphe dans la NPDU
Destination address in the DLPDU	Adresse de destination dans la DLPDU
Destination address in the NPDU	Adresse de destination dans la NPDU
Routing action ^a	Action de routage ^a
Yes	Oui
No	Non
Unicast	Unicast (monodiffusion)
Broadcast	Broadcast (diffusion)
Any	Any (tout type)
Discard	Rejeter
NOTE ^a See Table 57	NOTE ^a Voir Tableau 57.

Figure 34 – Décision de routage de la NPDU

Tableau 57 – Actions de routage

Code d'action	Description d'action
A1	S'il s'est agi de la fin de la liste des chemins de source dans la NPDU reçue ou si l'adresse suivante dans le chemin de source n'est pas celle d'un voisin, alors transmettre la NPDU reçue le long du chemin de graphe dans la NPDU reçue, sinon transmettre la NPDU reçue vers l'adresse suivante dans le chemin de source.

Code d'action	Description d'action
A2	S'il s'est agi de la fin de la liste des chemins de source dans la NPDU reçue, alors transmettre vers le Superframe_ID (Graph_ID contient le Superframe_ID), sinon transmettre la NPDU reçue vers l'adresse suivante dans le chemin de source.
B	Continuer la diffusion selon le graphe. Utiliser la liaison de diffusion dans la supertrame.
C	Si la NPDU reçue a une autre destination valide dans la liste de chemins de source, alors transmettre la NPDU vers l'adresse en question. Sinon, indiquer une erreur de chemin de source au NLM.
D	Transmettre la NPDU reçue vers la nouvelle adresse dans le chemin de source.
E	Transmettre la NPDU reçue le long du graphe, en utilisant n'importe quelle liaison conduisant au voisin sur le graphe; Si fin du graphe, alors indiquer une erreur de chemin de graphe au NLM.
F	Transmettre la NPDU reçue à tous les voisins dans le graphe; rejeter la NPDU si à la fin du graphe.
G	Graph_ID est le Superframe_ID; continuer à diffuser la NPDU en utilisant la liaison de diffusion dans la supertrame
Rejeter	La NPDU reçue n'est pas valide; la rejeter.

La valeur d'expiration de délai à transmettre à la DLE doit être recalculer en utilisant l'ASN reçu.

La NLE doit transmettre la NPDU à la DLE afin qu'elle puisse être émise dans le prochain intervalle, c'est-à-dire l'intervalle suivant immédiatement l'intervalle dans lequel la PDU a été reçue.

6.5.7 Informations de gestion de NL

6.5.7.1 Généralités

Plusieurs attributs de NL doivent être configurés avant qu'une NLE ne puisse fonctionner correctement. Ces attributs sont définis en 6.5.7.2. Les actions de la NLE qui peuvent être demandées par le biais de la gestion de la NL sont également récapitulées en 6.5.7.3.

6.5.7.2 Attributs de la NL

Les attributs d'une session sont montrés au Tableau 49.

Les attributs d'un chemin sont montrés au Tableau 52.

Les attributs source-route (chemin de source) sont montrés au Tableau 53.

D'autres attributs sont définis au Tableau 58.

Tableau 58 – Attributs de la NL

Nom d'attribut	Plage de valeurs	Définition
Unique ID (ID propre)	0 à $2^{40}-1$	ID propre au dispositif comme défini en 3.3.91
Nickname	0 à 65 534	Surnom du dispositif comme défini en 3.3.67
Default_HTL	0 à 255	Limite de durée de vie d'un paquet en termes de sauts; spécifiant le nombre de sauts qu'un paquet peut parcourir avant d'être rejeté; avec une valeur par défaut de 32
max_packet_age	0 à 65 535	Limite de durée de vie d'un paquet en termes de temps utilisant l'ASN; indiquant le nombre d'intervalles de temps après lequel un paquet doit être rejeté; avec une valeur par défaut de 30 000 temps ASN (égal à 300 s)

Nom d'attribut	Plage de valeurs	Définition
min_ads_needed	0 à 255	Nombre préférentiel d'annonces différentes avant de produire une demande de rattachement; avec une valeur par défaut de 3
Ad_wait_timeout	0 à $2^{32}-1$ en unités de ms	Durée à attendre tout en tentant de recevoir des annonces supplémentaires; avec une valeur par défaut de 30 s
max_join_retries	0 à 255	Limite des tentatives de rattachement; avec une valeur par défaut de 5
Join_rsp_time	0 à $2^{32}-1$ en unités de ms	Durée de l'expiration de délai pour la réponse de rattachement "Join", sa valeur par défaut étant Keep_alive_time (voir Tableau 46)
Bcast_reply_time	0 à $2^{32}-1$ en unités de ms	Durée maximale pour répondre à un message de diffusion; avec une valeur par défaut de 60 s
Health_report_time	0 à 65 535 en unités de ms	Intervalle auquel éditer des rapports de santé; avec une valeur par défaut de 15 minutes
Session_table_size	jusqu'à 255	Taille du tableau de session (à savoir, nombre maximal d'entrées)
Num_session_table_entries	jusqu'à 255	Nombre d'entrées actuelles dans le tableau de session
Route_table_size	jusqu'à 255	Taille du tableau de chemins (à savoir, nombre maximal d'entrées)
Num_route_table_entries	jusqu'à 255	Nombre d'entrées actuelles dans le tableau de chemins

6.5.7.3 Actions du NLM

Les actions de la NLE que peut demander la gestion de la NL sont définies au Tableau 59.

Tableau 59 – Actions de la NL

Nom	Description
RESET	Réinitialiser et initialiser la NLE. Tous les tableaux du réseau sont effacés lorsque cette primitive est invoquée. Cette primitive est normalement invoquée à la mise sous tension du dispositif ou lorsque le dispositif est en cours d'installation dans un nouveau réseau.
WRITE-SESSION	Ajoute une nouvelle session.
DELETE-SESSION	Supprime une session existante.
ADD-ROUTE	Ajoute un nouveau chemin.
DELETE-ROUTE	Supprime un chemin existant.
DEFAULT-ROUTE	Établit un chemin comme chemin par défaut.
ADD-TIMETABLE	Ajoute un nouvel indicateur horaire (Timetable).
DELETE- TIMETABLE	Supprime un indicateur horaire (Timetable) existant.
JOIN	Change l'état de "Idle" en "Joining".
DISCONNECT	Fait passer l'état à "Idle".

6.5.7.4 Gestion de réseau

Le gestionnaire de réseau a la responsabilité de la gestion globale, de la programmation et de l'optimisation du réseau. La fonction de gestion de réseau est spécifiée à l'Annexe C.

6.6 Couche transport

6.6.1 Généralités

La couche transport est utilisée pour assurer que les données d'utilisateur de TLS sont communiquées à leur destination finale. La couche transport prend en charge les transactions tant acquittées que non acquittées.

Le service Unacknowledged (non acquitté) permet à un dispositif d'envoyer des données d'utilisateur de TLS sans exiger d'acquiescement de la part de l'extrémité de réception. Les données peuvent être livrées au dispositif de destination dans un ordre différent de celui dans lequel elles ont été initialement envoyées du dispositif source.

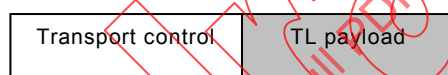
Le service acknowledged (acquitté) est utilisé pour construire un tube de transport synchrone à travers le réseau reliant deux dispositifs. Ce tube permet aux dispositifs d'envoyer des données et de confirmer leur livraison. L'utilisateur du TLS peut envoyer des données qui seront livrées à l'utilisateur de TLS de réception sans perte, duplication ou ordre incorrect. Ce service convient à un trafic de demandes et de réponses ainsi qu'à des notifications d'événements. Lorsque le service acquitté est utilisé, la communication est synchrone. Une seule transaction en cours à la fois est autorisée à travers le réseau pour le tube de transport en question.

6.6.2 Structure de TPDU

6.6.2.1 Généralités

La Figure 35 montre la structure d'une TPDU. Elle doit être constituée des champs dans une séquence décrite ci-dessous:

- Une commande de transport d'un seul octet, et
- La charge utile de la TL.



Légende

Anglais	Français
Transport control	Commande de transport
TL payload	Charge utile de couche transport

Figure 35 – Structure de TPDU

6.6.2.2 Transport control (Commande de transport)

Ce champ spécifie la longueur et le type de services comme montré à la Figure 36. Il comporte quatre sous-champs.

TACK: si le bit 7 est mis à '1', la TPDU est pour le service acquitté, sinon il est pour le service non acquitté.

TRESP: si le bit 6 est mis à '1', la TPDU est pour la réponse comme paramètre de type transport spécifié de la primitive de TLS,

TBROAD: si le bit 5 est mis à '1', la TPDU est pour le tube de diffusion; ce bit est mis à '1' aussi bien pour la demande que pour la réponse,

TSEQN: les bits de 4 à 0 forment le numéro de séquence pour la TPDU afin de détecter un doublon de TPDU, une TPDU manquante et de corréler la réponse à la demande.

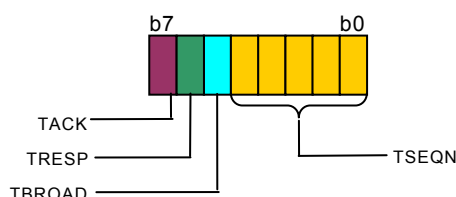


Figure 36 – Octet de commande de TPDU

6.6.3 Tableau de tube de transport

Le Tableau 60 montre les attributs de tube de transport. Pour chaque échange de données acquitté, une nouvelle entrée doit être créée dans le tableau de tube de transport. Un transfert de données non acquitté n'exige pas d'entrée dans ce tableau.

Le dispositif de terrain est tenu d'avoir la capacité pour suivre les tubes de transport:

- esclave pour échange de données de monodiffusion avec le gestionnaire de réseau,
- esclave pour échange de données de diffusion avec le gestionnaire de réseau,
- esclave pour échange de données de monodiffusion avec la passerelle,
- maître pour les notifications d'événements au gestionnaire de réseau, et
- maître pour les notifications d'événements à la passerelle.

La passerelle et le gestionnaire de réseau sont tenus de suivre plusieurs tubes de transport pour chaque dispositif et tubes pour tous les dispositifs de terrain connectés au réseau. Lorsqu'une diffusion acquittée est générée par le gestionnaire de réseau ou la passerelle, la TLE doit assurer que les acquittements sont reçus en provenance de tous les dispositifs concernés, qui peuvent potentiellement être tous les dispositifs.

Chaque entrée dans le tableau de tubes de transport inclut des attributs indiquant si, oui ou non, elle est active (une transaction est en cours), le dispositif agit en maître et si, oui ou non, elle est de diffusion. La demande "TL-Data-exchange request" émane de l'extrémité maître du tube. L'indication "TL-Data-exchange indication" est fourni au niveau de l'extrémité esclave du tube.

Si l'entrée du tableau de tubes de transport est pour l'extrémité maître du tube, le compteur de répétitions de tentatives et le temporisateur de tentatives sont également requis.

Tableau 60 – Attributs du tube de transport

Nom d'attribut	Plage de valeurs	Définition
Active	TRUE, FALSE	TRUE si le tube de transport est actif
Master	TRUE, FALSE	TRUE s'il est l'extrémité maître du tube
Broadcast	TRUE, FALSE	TRUE si le tube de transport est pour la demande de diffusion
Bcast_reply_time	0 à 65 535 en unités de ms	Durée maximale pour répondre à un message de diffusion; la valeur par défaut est 60 s.
sequence_number	0 à 31	Sur l'extrémité maître du tube, le numéro de séquence de la demande en cours; Sur l'extrémité esclave du tube, le numéro de séquence acquitté le dernier
TPDU_handle	0 à 255	Sur l'extrémité maître du tube seulement, le descripteur contenu dans la demande "TL-DATA-EXCHANGE request" correspondante à last_TPDU; n'est pas utilisé sur l'extrémité esclave du tube.
last_TPDU		Sur l'extrémité maître du tube, la dernière TPDU acquittée (ou NULL si aucune); Sur l'extrémité esclave du tube, la dernière TPDU acquittée
retry_count	0 à 255	Sur l'extrémité maître du tube seulement, le nombre de tentatives de communication; n'est pas utilisé sur l'extrémité esclave du tube.
max_reply_time		Durée à attendre une réponse à la demande en cours, avant de réessayer; la valeur par défaut est 30 s.
max_retries	0 à 255	Nombre de tentatives par la couche transport pour accomplir une émission de paquet acquittée; la valeur par défaut est 5 s.

6.6.4 Éléments de procédure TLE

6.6.4.1 Réception de la primitive de demande "TL-DATA-TRANSFER request"

La TLE doit coder l'octet de commande de transport comme montré au Tableau 61.

Tableau 61 – Codage de commande de transport pour la demande "TL-DATA-TRANSFER request"

Transport-type	tbroad	tack	tresp
Block-transfer request (demande de transfert par blocs),	0	0	0
Block-transfer response (réponse de transfert par blocs),	0	0	1
Search-broadcast (scrutation en diffusion),	1	0	0
Publish-broadcast (édition en diffusion),	1	0	1
Response-broadcast (réponse en diffusion),	1	1	1
Publish-unicast (édition en monodiffusion);	0	0	1
NOTE D'autres valeurs de Transport-type sont utilisées par d'autres services – voir Tableau 62 et Tableau 63.			

Sur le côté maître du tube de transport, la TLE doit incrémenter `sequence_number` and est copier ses cinq bits de poids faible dans le sous-champ `TSEQN`. Sur le côté esclave (récepteur) du tube de transport, la TLE doit copier les cinq bits de poids faible du `Handle` (descripteur), reçu dans la primitive "request", dans le sous-champ `TSEQN`. La TLE doit transmettre la TPDU vers la NLE en utilisant la primitive de demande "NL-TRANSMIT request". La valeur du paramètre `Timetable_ID` dans cette primitive "request" doit être utilisée dans la primitive de demande "NL-TRANSMIT request".

6.6.4.2 Réception de la primitive de demande "TL-DATA-EXCHANGE request"

6.6.4.2.1 Généralités

Si ce tube de transport attend un acquittement pour une TPDU précédente et aucun autre tampon n'est disponible, le service échoue et la primitive de confirmation "TL-DATA-EXCHANGE confirm" doit être retournée à l'utilisateur de TLS pour signaler l'erreur et la demande doit être rejetée.

Sinon, la TLE doit vérifier le paramètre "transport-type" (type de transport) et coder l'octet de commande de transport comme montré au Tableau 62. La valeur du paramètre `Timetable_ID` dans cette primitive "request" doit être utilisée dans la primitive de demande "NL-TRANSMIT request".

Tableau 62 – Codage de commande de transport pour la demande "TL-DATA-EXCHANGE request"

Transport-type	tbroad	tack	tresp
Request-unicast	0	1	0
Request-broadcast	1	1	0
NOTE D'autres valeurs de Transport-type sont utilisées par d'autres services – voir Tableau 61 et Tableau 63.			

6.6.4.2.2 Demande acquittée

La demande exige une réponse de la destination. Par conséquent, le sous-champ TACK du champ commande de transport est mis à '1'. La TLE doit rechercher dans le tableau de tubes de transport une entrée qui correspond à la primitive "service". Si une telle entrée n'existe pas, la TLE doit créer une entrée dans le tableau de tubes de transport et mettre le `sequence_number` pour l'entrée en question à un nombre aléatoire.

Si un tube de transport est disponible et il y a déjà une demande dans le tube de transport attendant un acquittement provenant de la TLE de destination et aucun autre tampon n'est disponible, la demande échoue. La TLE doit transmettre la primitive de confirmation "TL-DATA-EXCHANGE confirm" à l'utilisateur de TLS avec statut d'erreur.

Si un tube de transport est disponible et soit il n'y a aucune demande en attente dans ce tube de transport, soit un tampon est disponible, alors le `sequence_number` issu de l'entrée de tableau doit être incrémenté et les cinq bits de poids faible doivent être copiés dans le sous-champ TSEQN du champ "transport control". Dans l'entrée de tube de transport pour cette primitive "service", le fanion Active doit être mis à TRUE; le `retry_count` doit être remis à 0; et `replyTimer` doit être mis à `max_reply_time` et lancé pour attendre la réponse.

La TLE doit transmettre la TPDU vers la NLE en utilisant la primitive de demande "NL-TRANSMIT request". La TPDU doit aussi être mise en tampon pour de possibles tentatives futures.

6.6.4.3 Réception de la primitive de réponse "TL-DATA-EXCHANGE response"

La TLE doit scruter le tableau de tubes de transport pour trouver l'entrée qui correspond à l'adresse du correspondant. Il peut y avoir deux telles entrées pour chaque adresse de correspondant: de diffusion et de monodiffusion. La TLE doit utiliser le paramètre `transport-type` de la primitive pour trouver le tube de transport correct.

La TLE doit le paramètre `Handle` (`sequence_number`) contenu dans la primitive "response" au `sequence_number` contenu dans l'entrée de tube de transport. Si le `sequence_number` n'est pas supérieur à celui contenu dans l'entrée de tube de transport, le service échoue et la réponse doit être rejetée. La TLE ne doit pas transmettre de TPDU à la NLE.

Sinon, la TLE doit préparer la TPDU, mettre à jour le `sequence_number` dans l'entrée de tube de transport, coder l'octet de commande de transport comme montré au Tableau 63. Si `transport-type` est Response-unicast, la TPDU doit être immédiatement transmise à la NLE en utilisant la primitive de demande "NL-TRANSMIT request". Sinon, le `transport-type` est Response-broadcast, la TLE doit attendre une durée aléatoire entre zéro et la valeur de `Bcast_reply_time` (voir Tableau 60). Puis, à l'expiration de cette durée, la TLE doit transmettre la TPDU à la NLE en utilisant la primitive de demande "NL-TRANSMIT request". La TLE doit aussi placer la TPDU dans un tampon en vue d'une possible future répétition de tentative.

Tableau 63 – Codage de commande de transport pour la réponse "TL-DATA-EXCHANGE response"

Transport-type	tbroad	tack	tresp
Response-unicast	0	1	1
NOTE D'autres valeurs de Transport-type sont utilisées par d'autres services – voir Tableau 61 et Tableau 62.			

6.6.4.4 Réception de la primitive "NL-Transmit indication"

6.6.4.4.1 Généralités

Lorsque la TLE reçoit une primitive "indication" provenant de la NLE, elle doit examiner les sous-champs commande de transport TACK et TRESP, et l'utiliser pour traiter la TPDU reçue.

6.6.4.4.2 Transfert de données non acquitté

Si le sous-champ TACK de la commande de transport est FALSE, la TLE doit transmettre la TPDU reçue vers l'utilisateur de TLS dans la primitive d'indication "TL-DATA-TRANSFER indication". Le sous-champ TSEQN doit être transmis comme sequence-number.

6.6.4.4.3 Transfert de données d'indication acquitté

Si le sous-champ TACK de commande de transport est TRUE et TRESP est FALSE, la TLE doit valider le sous-champ TSEQN. La TLE doit scruter le tableau de tubes de transport pour trouver l'entrée qui correspond à l'adresse du correspondant. Il peut y avoir deux telles entrées pour chaque adresse de correspondant:: de diffusion et de monodiffusion. La TLE doit utiliser le sous-champ TBROAD de commande de transport pour trouver le tube de transport correct.

S'il n'y a pas une telle entrée dans le tableau de tubes de transport, la TLE doit créer une nouvelle entrée. La TLE doit mettre l'attribut Active à true, le bit d'attribut Master à false et l'attribut sequence_number à une unité de moins que le sous-champ TSEQN contenu dans la TPDU reçue. La TLE doit transmettre la primitive d'indication "TL-DATA-EXCHANGE indication" à l'utilisateur de TLS. Le type de transport et le numéro de séquence dans la primitive doivent être mis aux valeurs reçues dans le champ commande de transport.

Si l'entrée existe dans le tableau de tubes de transport, le sous-champ TSEQN dans la TPDU reçue doit être comparé au sequence_number dans l'entrée en question du tableau de tubes de transport. Si les deux sont identiques, la TPDU est un doublon et la TLDPD de réponse précédemment mise en tampon ("bufferisée") doit être transmise à la NLE dans la primitive de demande "NL-TRANSMIT request". Sinon, si le sous-champ TSEQN dans la TPDU reçue n'est pas supérieur d'une unité à celui contenu dans l'entrée du tableau de tubes de transport, la TPDU doit être rejetée.

Sinon, le sous-champ TSEQN dans la TPDU reçue est supérieur d'une unité à celui contenu dans l'entrée du tableau de tubes de transport. La TLE doit transmettre la charge utile de TL reçue comme TLS-user-data dans la primitive d'indication "TL-DATA-EXCHANGE indication" à l'utilisateur de TLS. Le type de transport et le numéro de séquence dans la primitive doivent être mis aux valeurs reçues dans le champ commande de transport.

6.6.4.4.4 Transfert de données de réponse acquitté

Si le sous-champ TACK de commande de transport est TRUE et TRESP est TRUE, la TLE doit valider le sous-champ TSEQN. La TLE doit scruter le tableau de tubes de transport pour trouver l'entrée qui correspond à l'adresse du correspondant. Il peut y avoir deux telles entrées pour chaque adresse de correspondant: de diffusion et de monodiffusion. La TLE doit utiliser le sous-champ TBROAD de commande de transport pour trouver le tube de transport correct.

S'il n'y a pas une telle entrée dans le tableau de tubes de transport, la TLE doit rejeter la TPDU reçue.

Si l'entrée existe dans le tableau de tubes de transport, le sous-champ TSEQN dans la TPDU reçue doit être comparé au sequence_number dans l'entrée en question du tableau de tubes de transport. Si les deux sont identiques, la TPDU est une réponse à une demande envoyée précédemment. La TLE doit mettre l'attribut Active de l'entrée en question à false, désactiver replyTimer et supprimer le tampon qui avait été utilisé pour stocker la TPDU pour la demande.

La TLE doit transmettre la charge utile de TL reçue comme TLS-user-data dans la primitive de confirmation "TL-DATA-EXCHANGE confirm" à l'utilisateur de TLS. Le Handle dans la primitive doit être mis aux valeurs reçues dans la primitive "indication" correspondante.

6.6.4.5 Réception d'une primitive de confirmation "NL-TRANSMIT confirm"

Si la demande correspondante "NL-TRANSMIT request" était due à la demande " TL-DATA-TRANSFER request", la TLE doit transmettre la primitive de confirmation "TL-DATA-TRANSFER confirm" à l'utilisateur de TLE. Si la demande correspondante "NL-TRANSMIT request" était due à la demande " TL-DATA-EXCHANGEREquest", avec la valeur pour Transport-type égale à Request-broadcast, la TLE doit transmettre la primitive de confirmation "TL-DATA-EXCHANGE confirm" à l'utilisateur de TLE.

6.6.4.6 Expiration de replyTimer

Si replyTimer expire, cela indique la perte soit de la TPDU de demande, soit de la TPDU de réponse correspondante. Si le retry_count n'est pas égal à max_retries, la TLE doit envoyer la TPDU de demande et incrémenter retry_count. Sinon, le retry_count est à sa valeur maximale, la demande a échoué et la TLE doit mettre l'attribut Active de l'entrée en question à false, désactiver replyTimer et supprimer le tampon qui avait été utilisé pour stocker la TPDU pour la demande. La TLE doit transmettre la primitive de confirmation "TL-DATA-EXCHANGE confirm" à l'utilisateur de TLS avec statut d'erreur. Le Handle dans la primitive doit être mis à la valeur reçue dans la primitive "request" correspondante.

6.6.5 Diagrammes d'états de TLE

6.6.5.1 Diagramme d'états de transfert de données client

Les états sont montrés à la Figure 37 et le diagramme d'états complet est décrit au Tableau 64.

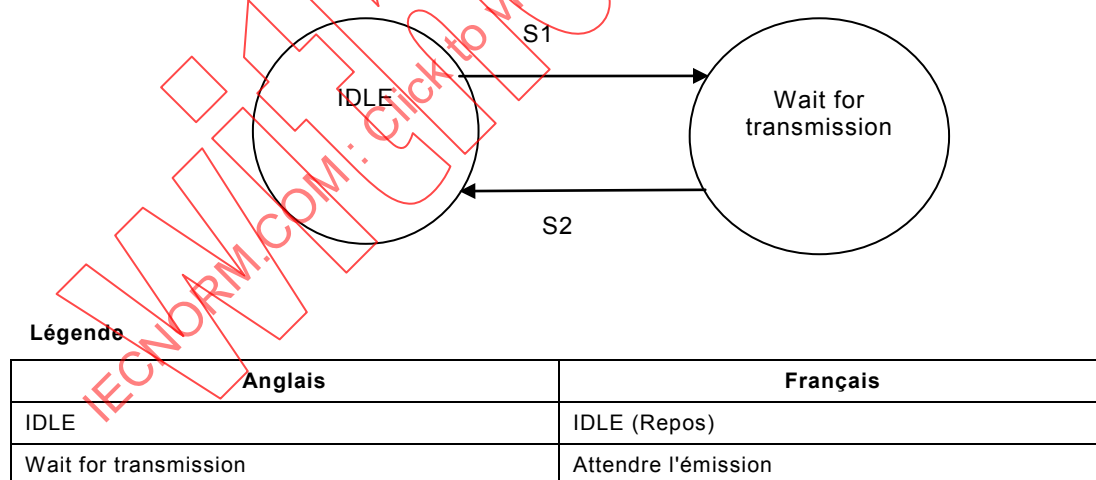


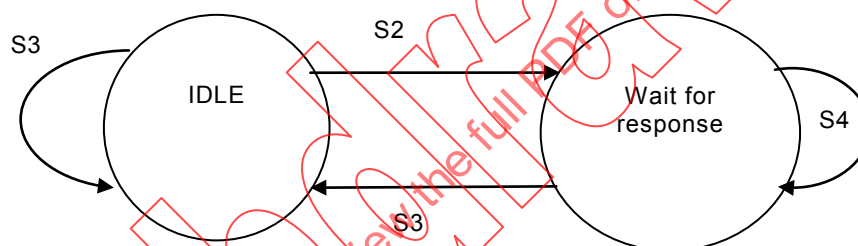
Figure 37 – Diagramme de transition relatif au diagramme d'états de transfert de données client

Tableau 64 –Tableau d'états de transfert de données client

#	État courant	Événement ou condition => action	Prochain état
S1	IDLE	TL-DATA-TRANSFER.request => NL-TRANSMIT.request { destination address:= TL-DATA-TRANSFER.request.destination address, priority:= TL-DATA-TRANSFER.request.priority, Timetable_ID:= TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data:= TLPDU } (voir 6.6.4.1)	Wait for transmission
S2	Wait for transmission	NL-TRANSMIT.confirm => TL-DATA-TRANSFER.confirm { Status:= status } (voir 6.6.4.5)	IDLE

6.6.5.2 Diagramme d'états d'échange de données client

Les états sont montrés à la Figure 38 et le diagramme d'états complet est décrit au Tableau 65.



Légende

Anglais	Français
IDLE	IDLE (Repos)
Wait for response	Attendre la réponse

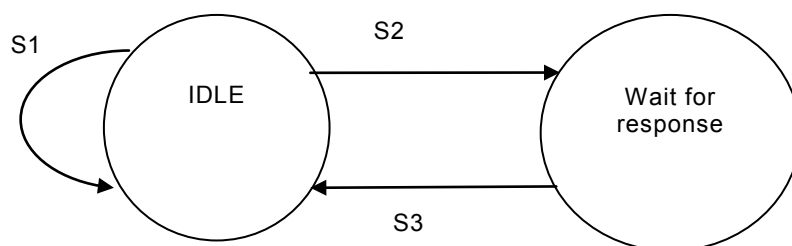
Figure 38 – Diagramme de transition relatif au diagramme d'états d'échange de données client

Tableau 65 –Tableau d'états d'échange de données client

#	État courant	Événement ou condition => action	Prochain état
S1	IDLE	TL-DATA-EXCHANGE.request && waiting for acknowledgement for a previous TPDU => TL-DATA-TRANSFER.confirm { Status:= failure }	IDLE
S2	IDLE	TL-DATA- EXCHANGE.request && !waiting for acknowledgement for a previous TPDU => retry_count:= 0, replyTimer:= max_reply_time, NL-TRANSMIT.request { destination address:= TL-DATA-TRANSFER.request.destination address, priority:= TL-DATA-TRANSFER.request.priority, Timetable_ID:= TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data:= TLPDU } (voir 6.6.4.2)	Wait for response
S3	Wait for response	NL-TRANSMIT.confirm && corresponding Transport-type == Request-broadcast => TL-DATA- EXCHANGE.confirm { Status:= status } (voir 6.6.4.5)	IDLE
S3	Wait for response	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU. TRESP == TRUE && transport pipe table entry exists && TLPDU. TSEQN == sequence_number in the transport pipe table entry => Disable replyTimer, Delete the buffer that was used to store the TPDU for the request, TL-DATA- EXCHANGE.confirm { TLS-user-data:= TLPDU.payload, Status:= status } (voir 6.6.4.4.4)	IDLE
S4	Wait for response	replyTimer expires && retry_count != max_retries => Increment retry_count, NL-TRANSMIT.request { destination address:= TL-DATA-TRANSFER.request.destination address, priority:= TL-DATA-TRANSFER.request.priority, Timetable_ID:= TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data:= TLPDU } (voir 6.6.4.6)	Wait for response
S3	Wait for response	replyTimer expires && retry_count == max_retries => TL-DATA-TRANSFER.confirm { Status:= failure } (voir 6.6.4.6)	IDLE

6.6.5.3 Diagramme d'états de serveur

Les états sont montrés à la Figure 39 et le diagramme d'états complet est décrit au Tableau 66.



Légende

Anglais	Français
IDLE	IDLE (Repos)
Wait for response	Attendre la réponse

Figure 39 – Diagramme de transition relatif au diagramme d'états serveur

Tableau 66 – Tableau d'états serveur

#	État courant	Événement ou condition => action	Prochain état
S1	IDLE	TL-DATA-TRANSFER.request => NL-TRANSMIT.request { destination address:= TL-DATA-TRANSFER.request.destination address, priority:= TL-DATA-TRANSFER.request.priority, Timetable_ID:= TL-DATA-TRANSFER.request.Timetable_ID, NLS-user-data:= TLPDU } (voir 6.6.4.1)	IDLE
S1	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == FALSE => TL-DATA-TRANSFER.indication { source address:= NL-TRANSMIT.indication.source address, Transport-type:= TLPDU.transport-type, priority:= TLPDU priority, TLS-user-data:= TLPDU payload } (voir 6.6.4.4.2)	IDLE
S1	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == FALSE && TLPDU.TSEQN == sequence_number in the transport pipe table entry (duplicate indication, voir 6.6.4.4.3) => NL-TRANSMIT.request { destination address:= previously buffered destination address, priority:= previously buffered priority, Timetable_ID:= previously buffered Timetable_ID, NLS-user-data:= previously buffered TLPDU } (voir 6.6.4.4.3)	IDLE
S2	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == FALSE && transport pipe table entry does not exist => Create transport pipe table entry, sequence_number in the transport pipe table entry = TLPDU.TSEQN -1 TL-DATA-EXCHANGE.indication { source address:= NL-TRANSMIT.indication.source address, sequence number:= TLPDU.TSEQN, Transport-type:= TLPDU.transport-type, priority:= TLPDU priority, TLS-user-data:= TLPDU payload } (voir 6.6.4.4.3)	Wait for response
S2	IDLE	NL-TRANSMIT.indication && TLPDU.TACK == TRUE && TLPDU.TRESP == FALSE && transport pipe table entry exists && TLPDU.TSEQN == sequence_number in the transport pipe table entry + 1 => TL-DATA-EXCHANGE.indication { source address:= NL-TRANSMIT.indication.source address, sequence number:= TLPDU.TSEQN, Transport-type:= TLPDU.transport-type, priority:= TLPDU priority, TLS-user-data:= TLPDU payload } (voir 6.6.4.4.3)	Wait for response
S3	Wait for response	TL-DATA-EXCHANGE.response && transport pipe table entry exists && sequence_number in the transport pipe table entry == response sequence number -1 => Increment sequence_number in the transport pipe table entry, NL-TRANSMIT.request { destination address:= TL-DATA-EXCHANGE.response.destination address, priority:= TL-DATA-EXCHANGE.response.priority, Timetable_ID:= TL-DATA-EXCHANGE.response.Timetable_ID, NLS-user-data:= TLPDU } (voir 6.6.4.3), Buffer TLPDU and NL-TRANSMIT.request parameters	IDLE

6.6.6 Informations de gestion de TL

Plusieurs attributs de TL doivent être configurés avant qu'une TLE ne puisse fonctionner correctement. Ces attributs sont définis en Tableau 60.

7 Définition des services de la couche application – Éléments de Type 20

7.1 Concepts communs

Les concepts et modèles communs utilisés pour décrire le service de couche application dans la présente norme sont détaillés dans le rapport CEI CEI/TR 61158-1, Article 9.

7.2 Data type ASE

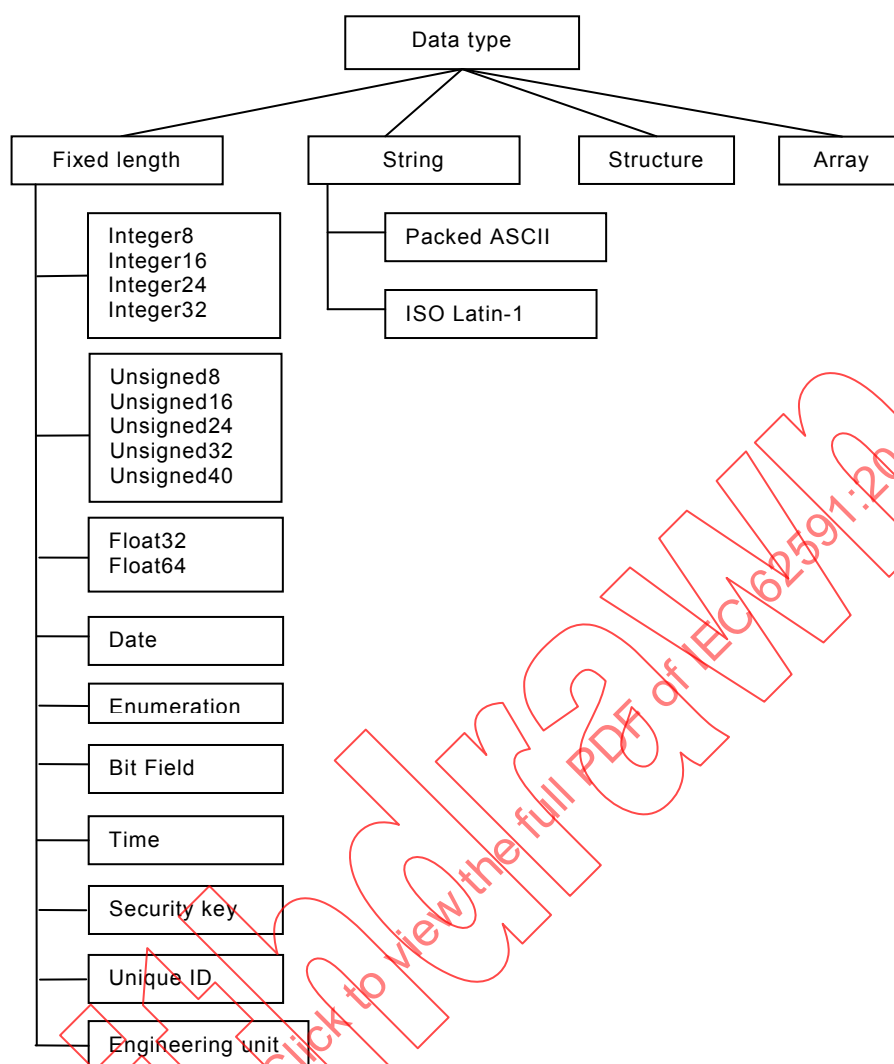
7.2.1 Vue d'ensemble

7.2.1.1 Généralités

Les types de données de réseau de terrain spécifient la syntaxe indépendante de toute machine pour les données d'application acheminées par les services FAL. La couche d'application de réseau de terrain prend en charge la définition et le transfert des types de données tant de base que construits. Les règles de codage pour les types de données spécifiés dans le présent Article sont fournies dans l'Article 8.

Les types de base sont des types atomiques qui ne peuvent pas être décomposés en plus de types élémentaires. Les types construits sont des types composés de types de bases et d'autres types construits. Leur complexité et leur profondeur d'imbrication ne sont pas contraintes par la présente norme.

Les types de données sont définis comme des instances de la classe de types de données, comme montré à la Figure 40.



Légende

Anglais	Français
Data type	Data type (Type de données)
Fixed length	Fixed length (Longueur fixe)
String	String (Chaîne)
Structure	Structure
Array	Array (Matrice)
Integer8	Integer8 (entier 8 bits)
Integer16	Integer16 (entier 16 bits)
Integer24	Integer24(entier 24 bits)
Integer32	Integer32 (entier 32 bits)
Unsigned8	Unsigned8 (entier non signé 8 bits)
Unsigned16	Unsigned16 (entier non signé 16 bits)
Unsigned24	Unsignedr24 (entier non signé non signé 24 bits)
Unsigned32	Unsigned32 (entier non signé 32 bits)
Unsigned40	Unsigned40 (entier non signé 40 bits)
Packed ASCII	Packed ASCII (ASII compressé)
ISO Latin-1	ISO Latin-1

Anglais	Français
Float32	Float32 (virgule flottante 32 bits)
Float64	Float64 (virgule flottante 64 bits)
Date	Date
Enumeration	Enumeration (Énumération)
Bit Field	Bit Field (Champ de bits)
Time	Time (Temps)
Security key	Security key (Clé de sécurité)
Unique ID	Unique ID (ID unique)
Engineering unit	Engineering unit (Unité d'étude)

Figure 40 – Hiérarchie de la classe de types de données

Les définitions des types de données sont représentées sous la forme d'une structure classe/format/instance commençant par la classe de types de données intitulée "Data type". Les formats pour les types de données sont définis par la classe de types de données.

Les classes de données de base sont utilisées pour définir des types de données de longueur fixe. Les types normalisés pris dans l'ISO/CEI 8824 sont appelés types de données *simples*. Les autres types de données de base normalisés sont définis spécifiquement pour les applications de réseau de terrain et sont appelés types spécifiques.

Les types construits spécifiés dans la présente norme sont des chaînes, des matrices et des structures.

7.2.1.2 Vue d'ensemble des types de base

La plupart des types de base sont définis à partir d'un ensemble de types de l'ISO/CEI 8824 (types simples). Certains types de l'ISO/CEI 8824 ont été étendus pour une utilisation spécifique dans le réseau de terrain (types spécifiques).

Les types simples sont des types universels de l'ISO/CEI 8824. Ils sont définis dans la présente norme pour leur fournir des identificateurs de classe de réseau de terrain.

Les types spécifiques sont des types de base définis spécifiquement pour utilisation dans l'environnement de réseau de terrain. Ils sont définis comme des sous-types de classe simples.

Les types de base ont une longueur constante. Deux variantes sont définies, l'une pour définir les types de données dont la longueur est un nombre entier d'octets et l'autre pour définir les types de données dont la longueur est exprimée en bits.

NOTE Integer, Packed ASCII, ISO Latin-1 et Date sont définis dans la présente norme dans le but de leur attribuer des identificateurs de classes de réseau de terrain. La présente norme ne change pas leurs définitions telles que spécifiées dans l'ISO/CEI 8824.

7.2.1.3 Types de longueur fixe

La longueur des types de longueur fixe est un nombre entier d'octets.

7.2.1.4 Types construits

7.2.1.4.1 Vue d'ensemble

Les types de données construits sont nécessaires pour transmettre complètement la diversité des informations présentes sur le réseau de terrain. Les types construits définis pour la présente norme sont de trois sortes: string (chaîne), array (matrice, tableau) et structure.

7.2.1.4.2 String

Une chaîne est composée d'un ensemble ordonné d'un nombre variable d'éléments de longueur fixe typés de façon homogène.

7.2.1.4.3 Structure

Une structure est constituée d'un ensemble ordonné d'éléments typés de façon hétérogène appelés champs. La présente norme ne limite pas le type de données des champs. Cependant, les champs dans une structure peuvent ne pas être du même type.

7.2.1.4.4 Array

Une matrice est composée d'un jeu ordonné d'éléments typés de façon homogène. Le type de données des éléments d'une matrice peut être un type de base ou une structure de longueur fixe. Tous les éléments d'une matrice doivent être du même type.

7.2.1.4.5 Niveau d'imbrication

La présente norme permet aux structures et aux matrices de contenir des structures et des matrices.

7.2.1.5 Spécification des types de données définis par l'utilisateur

Les types définis par l'utilisateur ne sont pas pris en charge par la présente norme.

7.2.1.6 Transfert de données d'utilisateur

Les données utilisateur sont transférées entre applications par le protocole FAL. Tout le codage et le décodage sont accomplis par l'utilisateur de la FAL.

Les règles pour coder les données d'utilisateur dans les unités de données du protocole FAL dépendent du type de données. Ces règles sont définies dans la CEI 61158-6-20.

7.2.2 Définition formelle des objets de types de données

7.2.2.1 Classe de types de données

La classe de types de données spécifie la racine de l'arbre de la classe des types de données. Sa classe parent "top" indique le sommet de l'arbre de la classe FAL.

FAL ASE: Data type ASE

CLASS: Data type

CLASS ID: non utilisé

PARENT CLASS: TOP

ATTRIBUTES:

1	(m)	Attribut-clé:	Data type Name
2	(m)	Attribut:	Format (FIXED LENGTH, STRING, STRUCTURE, ARRAY)
3	(c)	Contrainte:	Format = FIXED LENGTH STRING
3.1	(m)	Attribut:	Octet Length (Longueur en octets)
4	(c)	Contrainte:	Format = STRUCTURE
4.1	(m)	Attribut:	Number of Fields
4.2	(m)	Attribut:	List of Fields
4.2.1	(m)	Attribut:	Field Name
4.2.2	(m)	Attribut:	Field Data type (type de donnée de champ)
5	(c)	Contrainte:	Format = ARRAY
5.1	(m)	Attribut:	Number of Array Elements
5.2	(m)	Attribut:	Array Element Data type (type de donnée d'élément de matrice)

7.2.2.2 Attributs

Format

Cet attribut identifie le type de donnée comme étant fixed-length, string, structure ou array.

Octet Length

Cet attribut conditionnel définit la représentation des dimensions de l'objet de type associé. Il est présent lorsque la valeur de l'attribut format est "FIXED LENGTH" ou "STRING". Pour les types de données FIXED LENGTH, il représente la longueur en octets. Pour les types de données STRING, il représente la longueur en octets pour un élément simple d'une chaîne.

Number of Fields

Cet attribut conditionnel définit le nombre de champs dans une structure. Il est présent lorsque la valeur de l'attribut format est "STRUCTURE".

List of Fields

Cet attribut conditionnel est une liste ordonnée de champs contenus dans la structure. Chaque champ est spécifié par son type. Les champs sont décrits dans l'ordre de leur apparition. L'accès partiel à des champs dans une structure n'est pas pris en charge.

Nom dun Champ "Field Name"

Cet attribut conditionnel spécifie le nom du champ. Il est présent lorsque la valeur de l'attribut format est "STRUCTURE". **Type de donnée du champ** Field Data type

Cet attribut conditionnel spécifie le type de donnée du champ. Il est présent lorsque la valeur de l'attribut format est "STRUCTURE". Cet attribut peut lui-même spécifier un type de donnée construit en référençant une définition de type de donnée construite en assemblant une définition de type de donnée construit ici.

Number of Array Elements

Cet attribut conditionnel définit le nombre d'éléments de type array (matrice, tableau). Cet attribut est présent lorsque la valeur de l'attribut format est "ARRAY".

Array Element Data type

Cet attribut conditionnel spécifie le type de donnée pour le d'éléments d'une matrice. Tous les éléments de la matrice ont le même type de donnée. Il est présent lorsque la valeur de l'attribut format est "ARRAY". Cet attribut peut lui-même spécifier un type de donnée construit en référençant une définition de type de donnée construite par son identificateur numérique id.

7.2.3 Types de données définis pour la FAL

7.2.3.1 Types de longueur fixe

7.2.3.1.1 Integer8

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Integer8
2 Format	= FIXED LENGTH
2.1 Octet Length	= 1

Ce type integer est un nombre binaire en complément à deux avec une longueur égale à un octet.

7.2.3.1.2 Integer16

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Integer16

2 Format = FIXED LENGTH

2.1 Octet Length = 2

Ce type integer est un nombre binaire en complément à deux avec une longueur égale à deux octets.

7.2.3.1.3 Integer24

CLASS: **Data type**

ATTRIBUTES:

1 Data type Name = Integer24

2 Format = FIXED LENGTH

2.1 Octet Length = 3

Ce type integer est un nombre binaire en complément à deux avec une longueur égale à trois octets.

7.2.3.1.4 Integer32

CLASS: **Data type**

ATTRIBUTES:

1 Data type Name = Integer32

2 Format = FIXED LENGTH

2.1 Octet Length = 4

Ce type integer est un nombre binaire en complément à deux avec une longueur égale à quatre octets.

7.2.3.1.5 Unsigned8

CLASS: **Data type**

ATTRIBUTES:

1 Data type Name = Unsigned8

2 Format = FIXED LENGTH

2.1 Octet Length = 1

Ce type est un nombre entier. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type a une longueur égale à un octet.

7.2.3.1.6 Unsigned16

CLASS: **Data type**

ATTRIBUTES:

1 Data type Name = Unsigned16

2 Format = FIXED LENGTH

2.1 Octet Length = 2

Ce type est un nombre entier. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de deux octets.

7.2.3.1.7 Unsigned24

CLASS: **Data type**

ATTRIBUTES:

1 Data type Name = Unsigned24

2 Format = FIXED LENGTH

2.1 Octet Length = 3

Ce type est un nombre entier. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de trois octets.

7.2.3.1.8 Unsigned32

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Unsigned32
2 Format	= FIXED LENGTH
2.1 Octet Length	= 4

Ce type est un nombre entier. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de quatre octets.

7.2.3.1.9 Unsigned40

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Unsigned40
2 Format	= FIXED LENGTH
2.1 Octet Length	= 5

Ce type est un nombre entier. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de cinq octets.

7.2.3.1.10 Float32

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Float32
2 Format	= FIXED LENGTH
2.1 Octet Length	= 4

Ce type a une longueur de quatre octets. Le format pour Float32 est celui défini par l'ANSI/IEEE 754 comme simple précision ("single precision").

7.2.3.1.11 Float64

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Float64
2 Format	= FIXED LENGTH
2.1 Octet Length	= 8

Ce type a une longueur de huit octets. Le format pour Float64 est celui défini par l'ANSI/IEEE 754 comme double précision.

7.2.3.1.12 Date

CLASS:	Data type
ATTRIBUTES:	
1 Data type Name	= Date
2 Format	= FIXED LENGTH
2.1 Octet Length	= 3

Ce type de données est constitué du jour, du mois et de l'année moins 1900. Cela permet la représentation de n'importe quelle date entre le 1er janvier 1900 et le 31 décembre 2155.

7.2.3.1.13 Enumeration (énumération)**CLASS:** Data type**ATTRIBUTES:**

1	Data type Name	= Enumeration
2	Format	= FIXED LENGTH
2.1	Octet Length	= 1

Les éléments de données qui prennent une seule signification dans une liste ou dans un tableau sont codés comme Enumeration. Ce type de données utilise l'entier non signé d'une longueur égale à un octet. La valeur entière la plus grande est réservée et n'est utilisée par aucun service.

7.2.3.1.14 Bit Field**CLASS:** Data type**ATTRIBUTES:**

1	Data type Name	= Bit Field
2	Format	= FIXED LENGTH
2.1	Octet Length	= 1 or 2

Ce type de données est une série de huit bits numérotés de 0 à 7 ou de seize bits numérotés de 0 à 15. La communication d'informations codées sous forme de données d'un seul bit (telles que les informations relatives au statut et au diagnostic) utilise ce type de donnée.

7.2.3.1.15 Time**CLASS:** Data type**ATTRIBUTES:**

1	Data type Name	= Time
2	Format	= FIXED LENGTH
2.1	Octet Length	= 4

Ce type de données est un entier binaire non signé et représente le temps par incréments de 1/32 de milliseconde. Si ce type de données est utilisé pour représenter l'heure du jour, il indique le nombre de 1/32 de millisecondes écoulées depuis minuit.

7.2.3.1.16 Security key (Clé de sécurité)**CLASS:** Data type**ATTRIBUTES:**

1	Data type Name	= Security key
2	Format	= FIXED LENGTH
2.1	Octet Length	= 16

Ce type est une chaîne de seize octets. Chaque octet peut avoir n'importe quelle valeur de 0 à 255. L'interprétation de ce type de donnée dépend de l'application.

7.2.3.1.17 Unique ID**CLASS:** Data type**ATTRIBUTES:**

1	Data type Name	= Unique ID
2	Format	= FIXED LENGTH
2.1	Octet Length	= 5

Ce type est une chaîne de cinq octets. Chaque octet peut avoir n'importe quelle valeur de 0 à 255. L'interprétation de ce type de donnée dépend de l'application.

7.2.3.1.18 Engineering unit (Unité d'étude)**CLASS:** Data type

ATTRIBUTES:

- 1 Data type Name = Engineering unit
- 2 Format = FIXED LENGTH
- 2.1 Octet Length = 1

Ce type définit l'unité de mesure d'une variable mesurée. L'interprétation de ce type de données est spécifiée par le profil de communication (voir 9.3.1) et ne relève pas du domaine d'application de la présente norme.

7.2.3.2 Types String

7.2.3.2.1 Packed ASCII

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Packed ASCII
- 2 Format = STRING
- 2.1 Octet Length = 1 à n

Ce type est un sous-ensemble modifié du jeu de caractères ASCII. Ce sous-ensemble est montré au Tableau 340. Il est produit par la compression de quatre caractères Packed ASCII en trois octets de 8 bits,

7.2.3.2.2 ISO Latin-1

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Latin-1
- 2 Format = STRING
- 2.1 Octet Length = 1 à n

Ce type est un sous-ensemble du type string de l'ISO 8859-1 (ISO Latin-1). Le sous-ensemble est montré au Tableau 341

7.2.4 Spécification de service de Data type ASE

Il n'y a pas de services opérationnels définis pour l'objet data type.

7.2.5 Résumé des types de données

Le présent article contient un résumé des types de données définis comme montré au Tableau 67.

Tableau 67 – Résumé des types de données

Type de données	Paragraphe
Integer8	7.2.3.1.1
Integer16	7.2.3.1.2
Integer24	7.2.3.1.3
Integer32	7.2.3.1.4
Unsigned8	7.2.3.1.5
Unsigned16	7.2.3.1.6
Unsigned24	7.2.3.1.7
Unsigned32	7.2.3.1.8
Unsigned40	7.2.3.1.9
Float32	7.2.3.1.10

Type de données	Paragraphe
Float64	7.2.3.1.11
Date	7.2.3.1.12
Enumeration	7.2.3.1.13
Bit Field	7.2.3.1.14
Time	7.2.3.1.15
Security key	7.2.3.1.16
Unique ID	7.2.3.1.17
Engineering unit	7.2.3.1.18
Packed ASCII	7.2.3.2.1
ISO Latin-1	7.2.3.2.2

7.3 Spécification de modèle de communication

7.3.1 Paramètres communs

Plusieurs services ont les paramètres ci-après. Au lieu de les définir avec chaque service, les définitions communes ci-après sont fournies.

7.3.1.1 AREP ID

Ce paramètre spécifie des informations suffisantes pour identifier l'AREP de l'extrémité distante de l'AR. La valeur de ce paramètre peut être l'Unique ID de l'extrémité distante. Une valeur de ce paramètre est réservée comme adresse de diffusion.

7.3.1.2 Response Code

En l'absence d'erreur de communication, ce paramètre spécifie un report d'achèvement de commande indiquant le statut de l'exécution de la commande par le dispositif. Les valeurs possibles de ce paramètre sont montrées au Tableau 68. Son type de données est Enumeration.

Tableau 68 – Valeurs du code de réponse

Valeur	Description
Success (succès)	La commande (Read ou Write) a été exécutée correctement.
Warning (avertissement)	La commande (Write) a été exécutée avec l'écart décrit dans la réponse (par exemple, une valeur a été mise à sa valeur légale la plus proche.)
Error	La commande (Read ou Write) n'a pas été exécutée correctement. Le code de réponse (Response Code) en indique la raison (par exemple, le dispositif est protégé en écriture (mode "Write Protect")).

7.3.1.3 Statut du processus d'application (Application process status)

Ce paramètre indique le statut fourni par l'utilisateur de la FAL et n'est associé à l'achèvement d'aucune commande. Sa longueur est d'un octet.

7.3.1.4 Statut étendu (Extended status)

Ce paramètre indique le statut du VFD. Il n'est associé à l'achèvement d'aucune commande. Sa longueur est d'un octet.

7.3.2 ASE

7.3.2.1 ASE Dispositif de terrain virtuel

L'ASE dispositif de terrain virtuel (VFD) ASE doit être conforme à la CEI 61158-5-20:2010, 6.2.1.

7.3.2.2 ASE Variable

7.3.2.2.1 Vue d'ensemble

L'ASE Variable fournit des services pour lire ou écrire un objet variable dans le dispositif Server (serveur). La variable est identifiée par un numéro. Cet identificateur numérique décrit complètement le type de données et la structure de l'objet. Cet objet peut être de n'importe lequel des types définis en 7.2.3.

7.3.2.2.2 Modèle de Variable – commun

7.3.2.2.2.1 Simple Variable

7.3.2.2.2.1.1 Modèle formel

L'objet variable simple représente une variable simple unique qui est caractérisée par un Data type défini.

ASE:	VARIABLE ASE
CLASS:	Simple VARIABLE
PARENT CLASS:	TOP
ATTRIBUTES:	
1 (m) Attribut-clé:	Numeric Identifier
2 (o) Attribut:	Variable Name
3 (m) Attribut:	Data type Name
SERVICES:	
1 (m) OpsService:	Read
2 (o) OpsService:	Write
3 (o) OpsService:	Information report

7.3.2.2.2.1.2 Attributs

Numeric Identifier

Identifie une instance de cette classe d'objets.

Variable Name

Il s'agit du nom attribué à l'objet.

Data type Name

Il s'agit du nom attribué au type de données tel que spécifié en 7.2.3.1.

7.3.2.2.2.1.3 Services

Read

Ce service confirmé permet à un client de lire la valeur d'une variable.

Write

Ce service confirmé permet à un client d'écrire la valeur d'une variable.

Information report

Ce service non confirmé permet à un serveur d'envoyer la valeur d'une variable vers n'importe lequel ou l'ensemble des clients.

7.3.2.2.2.2 Variable Structure

7.3.2.2.2.2.1 Généralités

L'objet enregistrement consiste en un ensemble de variables simples ou d'une matrice de variables Structure de différents types de données. L'objet enregistrement est accessible dans son ensemble et ses champs individuels ne sont pas accessibles séparément.

ASE:		ASE VARIABLE
CLASS:		Structure VARIABLE
PARENT CLASS:		TOP
ATTRIBUTES:		
1	(m) Attribut-clé:	Numeric Identifier
2	(o) Attribut:	Variable Name
3	(m) Attribut:	Number of Fields
4	(m) Attribut:	List of Fields
4.1	(o) Attribut:	Field Name
4.2	(m) Attribut:	Field Data type Name
SERVICES:		
1	(m) OpsService:	Read
2	(o) OpsService:	Write
3	(o) OpsService:	Information report

7.3.2.2.2.2.2 Attributs

Numeric Identifier

Identifie une instance de cette classe d'objets.

Variable Name

Il s'agit du nom attribué à l'objet.

Number of Fields

Cet attribut définit le nombre de champs dans une structure.

Field Name

Cet attribut facultatif spécifie le nom du champ.

Field Data type Name

Il s'agit du nom attribué au type de données tel que spécifié en 7.2.3.1.

7.3.2.2.2.2.3 Services

Read

Ce service confirmé permet à un client de lire la valeur d'une variable.

Write

Ce service confirmé permet à un client de lire la valeur d'une variable.

Information report

Ce service non confirmé permet à un serveur d'envoyer la valeur d'une variable vers n'importe lequel ou l'ensemble des clients.

7.3.2.2.2.3 Array of simple variable

7.3.2.2.2.3.1 Modèle formel

Cet Objet Array est utilisé pour définir une variable construite dans laquelle tous les éléments sont de type de données et de longueur identiques et de la classe Simple Variable. Un ou

plusieurs objets de ce type d'objet à la fois sont accessibles en spécifiant le sous-indice de départ et le nombre d'éléments auxquels accéder.

ASE: **ASE VARIABLE**
CLASS: **Array of Simple VARIABLE**
PARENT CLASS: **Simple variable**
ATTRIBUTES:
ATTRIBUTES:
 4 (m) Attribut: Array size

7.3.2.2.3.2 Attributs

Array size

Il énonce le nombre d'éléments contenus dans la matrice Array. Si la valeur de cet attribut est "Variable", le nombre d'éléments n'est pas fixe.

7.3.2.2.4 Array of structure variable

7.3.2.2.4.1 Modèle formel

Cet Objet Array est utilisé pour définir une variable construite dans laquelle tous les éléments sont de type de données et de longueur identiques et de la classe Variable Structure. Un ou plusieurs objets de ce type d'objet à la fois sont accessibles en spécifiant le sous-indice de départ et le nombre d'éléments auxquels accéder.

ASE: **VARIABLE ASE**
CLASS: **Array of Structure VARIABLE**
PARENT CLASS: **Structure variable**
ATTRIBUTES:
ATTRIBUTES:
 5 (m) Attribut: Array size

7.3.2.2.4.2 Attributs

Array size

Il énonce le nombre d'éléments contenus dans la matrice Array. Si la valeur de cet attribut est "Variable", le nombre d'éléments n'est pas fixe.

7.3.2.2.3 Spécification de services pour l'ASE Variable

7.3.2.2.3.1 Services pris en charge

Ce paragraphe contient la définition de services qui sont propres à cet ASE. Les services définis pour cet ASE sont définis:

- Read variable from Server device (Lire la variable dans le dispositif Server),
- Write variable to Server device (Ecrire la variable dans le dispositif Server), et
- Send Information report from a server device.(Envoyer un rapport d'informations à partir du dispositif serveur).

Il y a des services séparés pour chaque variable, qui est défini par un nombre. Le format commun du service Read est défini en 7.3.2.2.3.3. Le format commun du service Write est défini en 7.3.2.2.3.4. Le format commun du service Information report (rapport d'informations) est défini en 7.3.2.2.3.5.

Pour chaque variable, il y a un service de lecture (Read) et un service d'écriture (Write) distincts. Ces services sont définis en 7.3.2.4, 7.3.2.5, 7.3.2.6 et 7.3.2.7.

NOTE Les définitions des services dans le présent paragraphe ne sont pas indépendantes de la variable à laquelle l'accès est effectué. Le présent paragraphe montre seulement le format du service. Les définitions spécifiques de services incluent le modèle de variable et les paramètres spécifiques aux services.

7.3.2.2.3.2 Corrélation de primitives

Au niveau du client, il n'y a qu'une seule demande en cours pour une valeur de l'AREP ID. L'AREP ID est acheminé comme adresse de destination dans les primitives "service" de la couche transport. Par conséquent, la primitive "confirm" est corrélée à la primitive "request" en utilisant l'AREP ID. Au niveau du serveur, il n'y a qu'une seule primitive "indication" en cours à un instant quelconque. Par conséquent, la primitive "response" est corrélée à la primitive "indication" attendant la réponse.

7.3.2.2.3.3 Service Read – format commun

7.3.2.2.3.3.1 Vue d'ensemble du service

Ce service permet à un client de lire la valeur d'une variable sur le serveur.

7.3.2.2.3.3.2 Primitives de service

Les paramètres de service pour ce service sont montrés au Tableau 69.

Tableau 69 – Paramètres du service Read

Nom de paramètre	Req	Ind	Rsp	Cnf
Argument				
AREP ID	M	M (=)		
Numeric Identifier	M	M (=)	M (=)	M (=)
Additional identifier	C	M (=)	M (=)	M (=)
Subindex	C	M (=)	M (=)	M (=)
Element count	C	M (=)	CM	M (=)
Array size			C	M(=)
Result (+)				
Variable value			S	S (=)
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)
Result (-)			S	S (=)
Status				S
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)

Argument

L'argument contient les paramètres de la demande de service.

Numeric Identifier

Ce paramètre identifie la variable. Il est l'attribut clé attribué à la valeur devant être lue. Il est aussi appelé numéro de commande.

Additional identifier

Pour certains des Numeric Identifier, ce paramètre fournit des informations complémentaires indispensables pour identifier la/les variable(s) devant être lue(s).

Subindex

Ce paramètre identifie l'élément individuel dans une variable matrice (array) par sa position au sein de la variable. Il peut s'agir soit d'une valeur numérique, soit d'une Enumeration.

Element count

Ce paramètre spécifie le nombre d'éléments de la matrice devant être lus. Ce paramètre n'est présent que si la variable est une variable array et si le Subindex est une valeur numérique. La valeur par défaut de ce paramètre est un (1). Dans les primitives "request" et "indication", sa valeur est le nombre souhaité d'éléments devant être lus. Dans les primitives "response" et "confirm", sa valeur est le nombre effectif d'éléments lus.

Array size

Ce paramètre spécifie le nombre total d'éléments disponibles de la matrice. Ce paramètre n'est présent que si la variable est une variable array.

Variable value

Ce paramètre spécifie la valeur lue issue du serveur.

Status

Ce paramètre spécifie la raison locale de la défaillance telle qu'une expiration de délai de communication. Si le serveur envoie une réponse, mais ne pouvait pas envoyer la "Variable value", alors cela est montré dans le "Response code".

7.3.2.2.3.4 Service Write – format commun

7.3.2.2.3.4.1.1 Vue d'ensemble du service

Ce service permet à un client d'écrire la valeur d'une variable sur le serveur.

7.3.2.2.3.4.1.2 Primitives de service

Les paramètres de service pour ce service sont montrés au Tableau 70.

Tableau 70 – Paramètres du service Write

Nom de paramètre	Req	Ind	Rsp	Cnf
Argument				
AREP ID	M	M (=)		
Numeric Identifier	M	M (=)	M (=)	M (=)
Additional identifier	C	M (=)	M (=)	M (=)
Subindex	C	M (=)	M (=)	M (=)
Variable value	M	M (=)		
Result (+)				
Variable value			S	S (=)
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)
Result (–)				
Status				S
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)

Argument

L'argument contient les paramètres de la demande de service.

Numeric Identifier

Ce paramètre identifie la variable. Il est l'attribut clé attribué à la valeur devant être lue. Il est aussi appelé numéro de commande.

Additional identifier

Pour certains des Numeric Identifier, ce paramètre fournit des informations complémentaires indispensables pour identifier la/les variable(s) devant être lue(s).

Subindex

Ce paramètre identifie l'élément individuel dans une variable matrice (array) par sa position au sein de la variable. Il peut s'agir soit d'une valeur numérique, soit d'une Énumération.

Variable value

Dans la primitive "request", ce paramètre spécifie la valeur souhaitée pour la variable devant être écrite. Dans la primitive "response", ce paramètre énonce la valeur effective pour la variable qui a été écrite. Dans certains cas, le serveur de réponse écrit une valeur différente de la valeur contenue dans la primitive "indication", par exemple il peut arrondir la valeur à la valeur la plus proche autorisée ou à la valeur la plus proche que le serveur peut prendre en charge.

NOTE La valeur de la variable retournée peut servir à l'utilisateur de la FAL pour valider l'opération d'écriture.

Status

Ce paramètre spécifie la raison locale de la défaillance telle qu'une expiration de délai de communication. Si le serveur envoie une réponse, mais ne pouvait pas écrire la "Variable value", alors cela est montré dans le "Response code".

7.3.2.2.3.5 Service "Information report" (rapports d'informations) – format commun

7.3.2.2.3.5.1 Vue d'ensemble du service

Ce service permet à un maître ou à un esclave d'envoyer la valeur d'une variable. Il est utilisé pour éditer des données de variable, d'envoyer des informations sur des événements ou pour acquitter un événement.

7.3.2.2.3.5.2 Primitives de service

Les paramètres de service pour ce service sont montrés au Tableau 71.

Tableau 71 –Paramètres du service "Information report"

Nom de paramètre	Req	Ind
Argument		
AREP ID	M	M (=)
Numeric Identifier	M	M (=)
Subindex	C	M (=)
Variable value	M	M (=)
Report Code	M	M (=)
Application process status	M	M (=)
Extended status	M	M (=)

Argument

L'argument contient les paramètres de la demande de service.

Numeric Identifier

Ce paramètre identifie la variable. Il est l'attribut clé attribué à la valeur devant être lue. Il est aussi appelé numéro de commande.

Subindex

Ce paramètre identifie l'élément individuel dans une variable matrice (array) par sa position au sein de la variable. Il peut s'agir soit d'une valeur numérique, soit d'une Enumeration.

Variable value

Ce paramètre spécifie la valeur rapportée issue du serveur.

7.3.2.3 ASE Action

7.3.2.3.1 Vue d'ensemble du service

Ce service permet à un client de commander une action spécifiée du serveur. Le format commun du service Action est défini dans le présent paragraphe. Pour chaque action, il y a un service distinct.

NOTE Les définitions des services dans le présent paragraphe ne sont pas indépendantes de l'action demandée. Le présent paragraphe montre seulement le format du service. Les définitions spécifiques de services incluent le comportement et les paramètres spécifiques aux services.

7.3.2.3.2 Corrélation de primitives

Au niveau du client, il n'y a qu'une seule demande en cours pour une valeur de l'AREP ID. L'AREP ID est acheminé comme adresse de destination dans les primitives "service" de la couche transport. Par conséquent, la primitive "confirm" est corréliée à la primitive "request" en utilisant l'AREP ID. Au niveau du serveur, il n'y a qu'une seule primitive "indication" en cours à un instant quelconque. Par conséquent, la primitive "response" est corréliée à la primitive "indication" attendant la réponse.

7.3.2.3.3 Modèle d'actions

7.3.2.3.3.1 Généralités

Certaines des actions exigent une variable pour valider l'action. Le modèle pour de telles actions est défini dans le présent paragraphe.

7.3.2.3.3.2 Modèle formel

ASE:		Simple Action
PARENT CLASS:		TOP
ATTRIBUTES:		
1	(m) Attribut-clé:	Numeric Identifier
ASE:		Qualified Action
PARENT CLASS:		TOP
ATTRIBUTES:		
1	(m) Attribut-clé:	Numeric Identifier
2	(m) Attribut:	Qualifier Name
3	(m) Attribut:	Data type Name

ASE: **Simple Variable Action**
PARENT CLASS: **TOP**

ATTRIBUTES:

1	(m)	Attribut-clé:	Numeric Identifier
2	(m)	Attribut:	Variable Name
3	(m)	Attribut:	Data type Name

ASE: **Structure Variable Action**
PARENT CLASS: **TOP**

ATTRIBUTES:

1	(m)	Attribut-clé:	Numeric Identifier
2	(m)	Attribut:	Variable Name
3	(m)	Attribut:	Number of Fields
4	(m)	Attribut:	List of Fields
4.1	(o)	Attribut:	Field Name
4.2	(m)	Attribut:	Field Data type Name

7.3.2.3.3 Attributs

Voir 7.3.2.2.2 pour la définition des attributs.

7.3.2.3.4 Primitives du service Action – format commun

Les paramètres de service pour ce service sont montrés au Tableau 72.

Tableau 72 – Paramètres du service Action

Nom de paramètre	Req	Ind	Rsp	Cnf
Argument				
AREP ID	M	M (=)		
Numeric Identifier	M	M (=)	M (=)	M (=)
Action qualifier	C	M (=)	M (=)	M (=)
Subindex	C	M (=)	M (=)	M (=)
Variable value	C	M (=)		
Result (+)				
Variable value			S	S (=)
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)
Result (-)				
Status				S
Response Code			M	M (=)
Application process status			M	M (=)
Extended status			M	M (=)

Argument

L'argument contient les paramètres de la demande de service.

Numeric identifier

Ce paramètre identifie l'action. Il est aussi appelé numéro de commande.

Action qualifier

Pour certains des Numeric Identifier, ce paramètre fournit des informations complémentaires indispensables pour identifier l'action à accomplir. Si l'action exige une variable et si le Numeric identifier n'est pas suffisant, ce paramètre identifie la variable en question.

Subindex

Si l'action exige une variable array, ce paramètre identifie l'élément individuel dans la variable array en question. Ce paramètre peut être soit une valeur numérique, d'une Enumeration.

NOTE Ce paramètre est présent si l'action consiste à ajouter un élément à la matrice ou à supprimer un élément de la matrice.

Variable value

Si l'action exige une variable, ce paramètre spécifie la valeur de la variable. Si l'action est "add", alors dans le paramètre derequête, ce paramètre spécifie la valeur souhaitée de la valeur à ajouter; dans la primitive "response", ce paramètre énonce la valeur effective pour la variable qui a été ajoutée.

Status

Ce paramètre spécifie la raison locale de la défaillance telle qu'une expiration de délai de communication. Si le serveur envoie une réponse, mais ne pouvait pas parachever l'action avec succès, alors cela est montré dans le "Response code".

7.3.2.4 Services d'application de dispositif

7.3.2.4.1 Read primary variable

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.4.

7.3.2.4.2 Read loop current and percent of range

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.5.

7.3.2.4.3 Read dynamic variables and loop current

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.6.

7.3.2.4.4 Write loop configuration

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.7.

7.3.2.4.5 Read loop configuration

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.8.

7.3.2.4.6 Read dynamic variable families classifications

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.9.

7.3.2.4.7 Read device variables with status

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.10.

7.3.2.4.8 Read message

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.11.

7.3.2.4.9 Read tag, descriptor and date

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.12.

7.3.2.4.10 Read primary variable transducer information

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.13.

7.3.2.4.11 Read device information

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.14.

7.3.2.4.12 Read final assembly number

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.15.

7.3.2.4.13 Write message

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.16.

7.3.2.4.14 Write tag, descriptor and date

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.17.

7.3.2.4.15 Write final assembly number

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.18.

7.3.2.4.16 Read long_tag

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.19.

7.3.2.4.17 Write long_tag

Ce service est spécifié dans la CEI 61158-5-20:2010, 6.2.2.3.20.

7.3.2.4.18 Reset configuration changed flag**7.3.2.4.18.1 Vue d'ensemble du service**

Ce service réinitialise le bit "Configuration changed" associé au client produisant ce service d'écriture.

7.3.2.4.18.2 Paramètres de demande

CLASS: Simple Variable Action

ATTRIBUTES:

1	Numeric Identifier	= 38
2	Variable Name	= Configuration change counter
3	Data type Name	= Unsigned16

7.3.2.4.19 Perform self test**7.3.2.4.19.1 Vue d'ensemble du service**

Ce service déclenche la fonction d'autotest dans le dispositif.

7.3.2.4.19.2 Paramètres de demande

CLASS: Simple Action

ATTRIBUTES:

1 Numeric Identifier = 41

7.3.2.4.20 Perform device reset

7.3.2.4.20.1 Vue d'ensemble du service

Ce service déclenche la fonction d'autotest dans le dispositif.

7.3.2.4.20.2 Paramètres de demande

CLASS: Simple Action

ATTRIBUTES:

1 Numeric Identifier = 42

7.3.2.4.21 Read additional device status

7.3.2.4.21.1 Vue d'ensemble du service

Ce service lit les informations de statut du dispositif qui ne sont pas incluses dans le "Response code" ou l'octet "Device status".

7.3.2.4.21.2 Modèle de Variable

Il y a trois modèles différents de Variable en fonction du type de dispositif. Le dispositif répond avec un seul de ces trois types de variables. Si le dispositif prend en charge plus d'une voie analogique, il prend en charge "Additional device status 2". Certains des dispositifs peuvent utiliser "Additional device status 3". Tous les autres dispositifs prennent en charge "Additional device status 1".

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 48
- 2 Variable name = Additional device status 1
- 3 Number of fields = variable
- 4 List of Fields
 - 4.1 (m) Field name = Device specific status 0
 - 4.2 (m) Field data type name = Bit Field
 - 4.3 (m) Field name = Extended status
 - 4.4 (m) Field data type name = Bit Field
 - 4.5 (m) Field name = Device operating mode
 - 4.6 (m) Field data type name = Bit Field
 - 4.7 (m) Field name = Standardized status 0
 - 4.8 (m) Field data type name = Bit Field

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 5 Numeric Identifier = 48
- 6 Variable name = Additional device status 2
- 7 Number of fields = variable
- 8 List of Fields
 - 8.1 (m) Field name = Device specific status 0
 - 8.2 (m) Field data type name = Bit Field
 - 8.3 (m) Field name = Extended status

- 8.4 (m) Field data type name = Bit Field
- 8.5 (m) Field name = Device operating mode
- 8.6 (m) Field data type name = Bit Field
- 8.7 (m) Field name = Standardized status 0
- 8.8 (m) Field data type name = Bit Field
- 8.9 (m) Field name = Standardized status 1
- 8.10 (m) Field data type name = Bit Field
- 8.11 (m) Field name = Analog channel saturated
- 8.12 (m) Field data type name = Bit Field
- 8.13 (m) Field name = Standardized status 2
- 8.14 (m) Field data type name = Bit Field
- 8.15 (m) Field name = Standardized status 3
- 8.16 (m) Field data type name = Bit Field
- 8.17 (m) Field name = Analog channel fixed
- 8.18 (m) Field data type name = Bit Field

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 9 Numeric Identifier = 48
- 10 Variable name = Additional device status 3
- 11 Number of fields = variable
- 12 List of Fields
 - 12.1 (m) Field name = Device specific status 0
 - 12.2 (m) Field data type name = Bit Field
 - 12.3 (m) Field name = Extended status
 - 12.4 (m) Field data type name = Bit Field
 - 12.5 (m) Field name = Device operating mode
 - 12.6 (m) Field data type name = Bit Field
 - 12.7 (m) Field name = Standardized status 0
 - 12.8 (m) Field data type name = Bit Field
 - 12.9 (m) Field name = Standardized status 1
 - 12.10 (m) Field data type name = Bit Field
 - 12.11 (m) Field name = Analog channel saturated
 - 12.12 (m) Field data type name = Bit Field
 - 12.13 (m) Field name = Standardized status 2
 - 12.14 (m) Field data type name = Bit Field
 - 12.15 (m) Field name = Standardized status 3
 - 12.16 (m) Field data type name = Bit Field
 - 12.17 (m) Field name = Analog channel fixed
 - 12.18 (m) Field data type name = Bit Field
 - 12.19 (m) Field name = Device specific status 1
 - 12.20 (m) Field Data type name = Array of Status

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Status
- 2 Format = ARRAY
- 2.1 Number of Array Elements = 1 à 11
- 2.2 Array Element Data type = Bit Field

7.3.2.4.22 Reset more status available

7.3.2.4.22.1 Vue d'ensemble du service

Ce service réinitialise le bit "More status available" correspondant au dispositif demandeur (maître primaire, maître secondaire ou passerelle). Le nombre de champs dans la variable structure associée dépend du serveur et il est de 4 au minimum et de 10 au maximum.

7.3.2.4.22.2 Paramètres de demande

CLASS: Structure Variable Action

ATTRIBUTES:

1	Numeric Identifier	= 48
2	Variable Name	= Additional device status
3	Number of fields	= variable
4	List of Fields	
4.1	(m) Field name	= Device specific status 0
4.2	(m) Field data type name	= Bit Field
4.3	(m) Field name	= Extended status
4.4	(m) Field data type name	= Bit Field
4.5	(m) Field name	= Device operating mode
4.6	(m) Field data type name	= Bit Field
4.7	(m) Field name	= Standardized status 0
4.8	(m) Field data type name	= Bit Field
4.9	(m) Field name	= Standardized status 1
4.10	(m) Field data type name	= Bit Field
4.11	(m) Field name	= Analog channel saturated
4.12	(m) Field data type name	= Bit Field
4.13	(m) Field name	= Standardized status 2
4.14	(m) Field data type name	= Bit Field
4.15	(m) Field name	= Standardized status 3
4.16	(m) Field data type name	= Bit Field
4.17	(m) Field name	= Analog channel fixed
4.18	(m) Field data type name	= Bit Field
4.19	(m) Field name	= Device specific status 1
4.20	(m) Data type	= Array of Status

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Array of Status
2	Format	= ARRAY
2.1	Number of Array Elements	= 1 à 11
2.2	Array Element Data type	= Bit Field

7.3.2.4.23 Read device variable information

7.3.2.4.23.1 Vue d'ensemble du service

Ce service lit les paramètres liés à la variable de dispositif.

7.3.2.4.23.2 Paramètres de demande

1	Numeric Identifier	= 54
2	Subindex	
2.1	Subindex Name	= Variable code
2.2	Subindex Data type	= Unsigned8

7.3.2.4.23.3 Paramètres de réponse

1	Numeric Identifier	= 54
2	Subindex	
2.1	Subindex Name	= Variable code
2.2	Subindex Data type	= Unsigned8

7.3.2.4.23.4 Modèle de Variable**CLASS:** Array of Structure VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 54
2	Variable Name	= Device variable information
3	Number of Fields	= 9
4	List of Fields	
4.1	Field Name	= Transducer serial number
4.2	Field Data type Name	= Unsigned24
4.3	Field Name	= Transducer unit
4.4	Field Data type Name	= Enumeration
4.5	Field Name	= Upper transducer limit
4.6	Field Data type Name	= Float32
4.7	Field Name	= Lower transducer limit
4.8	Field Data type Name	= Float32
4.9	Field Name	= Damping value
4.10	Field Data type Name	= Float32
4.11	Field Name	= Minimum span
4.12	Field Data type Name	= Float32
4.13	Field Name	= Device variable classification
4.14	Field Data type Name	= Enumeration
4.15	Field Name	= Device variable family
4.16	Field Data type Name	= Enumeration
4.17	Field Name	= Update period
4.18	Field Data type Name	= Time
5	Array size	= Variable

7.3.2.4.24 Write device variable**7.3.2.4.24.1 Vue d'ensemble du service**

Ce service permet de forcer temporairement une variable Device à avoir une valeur fixe.

7.3.2.4.24.2 Paramètres de demande

1	Numeric Identifier	= 79
2	Subindex	
2.1	Subindex Name	= Variable code
2.2	Subindex Data type	= Unsigned8

7.3.2.4.24.3 Paramètres de réponse

1	Numeric Identifier	= 79
2	Subindex	
2.1	Subindex Name	= Variable code
2.2	Subindex Data type	= Unsigned8

7.3.2.4.24.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 79
2	Variable Name	= Device variable record
3	Number of Fields	= 4
4	List of Fields	
4.1	Field Name	= Device variable command
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Device variable unit
4.4	Field Data type Name	= Engineering unit
4.5	Field Name	= Device variable value
4.6	Field Data type Name	= Float32
4.7	Field Name	= Device variable status
4.8	Field Data type Name	= Bit Field
5	Array size	= Variable

7.3.2.4.25 Read real-time clock

7.3.2.4.25.1 Vue d'ensemble du service

Ce service lit l'horloge temps réel (Real-time clock), y compris l'heure courante telle qu'estimée par le dispositif et la dernière fois que l'horloge a été réglée.

7.3.2.4.25.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 90
2	Variable Name	= Real-time clock
3	Number of Fields	= 5
4	List of Fields	
4.1	Field Name	= Date
4.2	Field Data type Name	= Date
4.3	Field Name	= Current time
4.4	Field Data type Name	= Time
4.5	Field Name	= Date last set
4.6	Field Data type Name	= Date
4.7	Field Name	= Time last set
4.8	Field Data type Name	= Time
4.9	Field Name	= RTC flags
4.10	Field Data type Name	= Bit Field

7.3.2.4.26 Write publish data period

7.3.2.4.26.1 Vue d'ensemble du service

Ce service sélectionne la période de mise à jour minimale et maximale d'un message Publish data.

7.3.2.4.26.2 Paramètres de demande

1	Numeric Identifier	= 103
2	Subindex	
2.1	Subindex Name	= Publish data message ref

2.2 Subindex Data type = Unsigned8

7.3.2.4.26.3 Paramètres de réponse

1 Numeric Identifier = 103
 2 Subindex
 2.1 Subindex Name = Publish data message ref
 2.2 Subindex Data type = Unsigned8

7.3.2.4.26.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1 Numeric Identifier = 103
 2 Variable Name = Publish data period
 3 Number of Fields = 2
 4 List of Fields
 4.1 Field Name = Update period
 4.2 Field Data type Name = Time
 4.3 Field Name = Maximum update period
 4.4 Field Data type Name = Time
 5 Array size = Variable

7.3.2.4.27 Write publish data trigger

7.3.2.4.27.1 Vue d'ensemble du service

Ce service configure le déclencheur qui force l'édition du message Publish data.

7.3.2.4.27.2 Paramètres de demande

1 Numeric Identifier = 104
 2 Subindex
 2.1 Subindex Name = Publish data message ref
 2.2 Subindex Data type = Unsigned8

7.3.2.4.27.3 Paramètres de réponse

1 Numeric Identifier = 104
 2 Subindex
 2.1 Subindex Name = Publish data message ref
 2.2 Subindex Data type = Unsigned8

7.3.2.4.27.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1 Numeric Identifier = 104
 2 Variable Name = Publish data trigger
 3 Number of Fields = 4
 4 List of Fields
 4.1 Field Name = Trigger mode
 4.2 Field Data type Name = Enumeration
 4.3 Field Name = Device variable classification
 4.4 Field Data type Name = Enumeration
 4.5 Field Name = Device variable unit
 4.6 Field Data type Name = Engineering unit

- 4.7 Field Name = Trigger level
- 4.8 Field Data type Name = Float32
- 5 Array size = Variable

7.3.2.4.28 Read publish data mode configuration

7.3.2.4.28.1 Vue d'ensemble du service

Ce service lit la configuration du mode Publish data.

7.3.2.4.28.2 Paramètres de demande

- 1 Numeric Identifier = 105
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.28.3 Paramètres de réponse

- 1 Numeric Identifier = 105
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.28.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 105
- 2 Variable Name = Publish data mode configuration
- 3 Number of Fields = 18
- 4 List of Fields
 - 4.1 Field Name = Publish data mode control
 - 4.2 Field Data type Name = Enumeration
 - 4.3 Field Name = Command number expansion
 - 4.4 Field Data type Name = Unsigned8
 - 4.5 Field Name = Device variable code 0
 - 4.6 Field Data type Name = Unsigned8
 - 4.7 Field Name = Device variable code 1
 - 4.8 Field Data type Name = Unsigned8
 - 4.9 Field Name = Device variable code 2
 - 4.10 Field Data type Name = Unsigned8
 - 4.11 Field Name = Device variable code 3
 - 4.12 Field Data type Name = Unsigned8
 - 4.13 Field Name = Device variable code 4
 - 4.14 Field Data type Name = Unsigned8
 - 4.15 Field Name = Device variable code 5
 - 4.16 Field Data type Name = Unsigned8
 - 4.17 Field Name = Device variable code 6
 - 4.18 Field Data type Name = Unsigned8
 - 4.19 Field Name = Device variable code 7
 - 4.20 Field Data type Name = Unsigned8
 - 4.21 Field Name = Max publish count
 - 4.22 Field Data type Name = Unsigned8
 - 4.23 Field Name = Extended command number

4.24	Field Data type Name	= Unsigned16
4.25	Field Name	= Update period
4.26	Field Data type Name	= Time
4.27	Field Name	= Maximum update period
4.28	Field Data type Name	= Time
4.29	Field Name	= Trigger mode
4.30	Field Data type Name	= Enumeration
4.31	Field Name	= Device variable classification
4.32	Field Data type Name	= Enumeration
4.33	Field Name	= Device variable unit
4.34	Field Data type Name	= Engineering unit
4.35	Field Name	= Trigger level
4.36	Field Data type Name	= Float32
5	Array size	= Variable

7.3.2.4.29 Flush delayed responses

7.3.2.4.29.1 Vue d'ensemble du service

Cette commande efface toutes les réponses différées en attente pour le Maître qui a émis la commande.

7.3.2.4.29.2 Paramètres de demande

CLASS: Simple Action

ATTRIBUTES:

1 Numeric Identifier = 106

7.3.2.4.30 Write publish data device variables

7.3.2.4.30.1 Vue d'ensemble du service

Ce service sélectionne les variables Device qui doivent être utilisées par un dispositif éditeur en mode Publish data.

7.3.2.4.30.2 Paramètres de demande

1 Numeric Identifier = 107
 2 Subindex
 2.1 Subindex Name = Publish data message ref
 2.2 Subindex Data type = Unsigned8

7.3.2.4.30.3 Paramètres de réponse

1 Numeric Identifier = 107
 2 Subindex
 2.1 Subindex Name = Publish data message ref
 2.2 Subindex Data type = Unsigned8

7.3.2.4.30.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1 Numeric Identifier = 107
 2 Variable Name = Publish data device variables
 3 Number of Fields = 8
 4 List of Fields

- 4.1 Field Name = Device variable code 0
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Device variable code 1
- 4.4 Field Data type Name = Unsigned8
- 4.5 Field Name = Device variable code 2
- 4.6 Field Data type Name = Unsigned8
- 4.7 Field Name = Device variable code 3
- 4.8 Field Data type Name = Unsigned8
- 4.9 Field Name = Device variable code 4
- 4.10 Field Data type Name = Unsigned8
- 4.11 Field Name = Device variable code 5
- 4.12 Field Data type Name = Unsigned8
- 4.13 Field Name = Device variable code 6
- 4.14 Field Data type Name = Unsigned8
- 4.15 Field Name = Device variable code 7
- 4.16 Field Data type Name = Unsigned8
- 5 Array size = Variable

7.3.2.4.31 Write publish data mode command number

7.3.2.4.31.1 Vue d'ensemble du service

Ce service sélectionne le message de réponse que le dispositif éditeur émet en mode Publish data.

7.3.2.4.31.2 Paramètres de demande

- 1 Numeric Identifier = 108
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.31.3 Paramètres de réponse

- 1 Numeric Identifier = 108
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.31.4 Modèle de Variable

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 108
- 2 Variable Name = Publish data mode command number
- 3 Data type Name = Unsigned16 / Unsigned8
- 4 Array size = Variable

7.3.2.4.32 Write publish data mode control

7.3.2.4.32.1 Vue d'ensemble du service

Ce service est utilisé pour activer ou désactiver le mode Publish data pour les dispositifs.

7.3.2.4.32.2 Paramètres de demande

- 1 Numeric Identifier = 109

- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.32.3 Paramètres de réponse

- 1 Numeric Identifier = 109
- 2 Subindex
- 2.1 Subindex Name = Publish data message ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.32.4 Modèle de Variable

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 109
- 2 Variable Name = Publish data mode control
- 3 Data type Name = Enumeration
- 4 Array size = Variable

7.3.2.4.33 Read event notification summary

7.3.2.4.33.1 Vue d'ensemble du service

Ce service lit la configuration de l'Event Notification (notification d'événement).

7.3.2.4.33.2 Paramètres de demande

- 1 Numeric Identifier = 115
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.33.3 Paramètres de réponse

- 1 Numeric Identifier = 115
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.33.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 115
- 2 Variable Name = event notification summary
- 3 Number of Fields = 8
- 4 List of Fields
- 4.1 Field Name = Maximum event count
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Event status
- 4.4 Field Data type Name = Bit Field
- 4.5 Field Name = Event notification control
- 4.6 Field Data type Name = Bit Field
- 4.7 Field Name = Event occurrence time
- 4.8 Field Data type Name = Time

- 4.9 Field Name = Event notification retry time
- 4.10 Field Data type Name = Time
- 4.11 Field Name = Maximum update time
- 4.12 Field Data type Name = Time
- 4.13 Field Name = Event de-bounce interval
- 4.14 Field Data type Name = Time
- 4.15 Field Name = Event mask
- 4.16 Field Data type Name = variable size array of Bit Field
- 5 Array size = Variable

7.3.2.4.34 Write event notification bit mask

7.3.2.4.34.1 Vue d'ensemble du service

Ce service établit le masque d'événement qui est utilisé pour déclencher une notification d'événement.

7.3.2.4.34.2 Paramètres de demande

- 1 Numeric Identifier = 116
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.34.3 Paramètres de réponse

- 1 Numeric Identifier = 116
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.34.4 Modèle de Variable

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 116
- 2 Variable Name = event notification bit mask
- 3 Data type Name = variable size array of Bit Field
- 4 Array size = Variable

7.3.2.4.35 Write event notification timing

7.3.2.4.35.1 Vue d'ensemble du service

Ce service fixe les paramètres de temporisation pour la notification d'événement.

7.3.2.4.35.2 Paramètres de demande

- 1 Numeric Identifier = 117
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.35.3 Paramètres de réponse

- 1 Numeric Identifier = 117
- 2 Subindex

- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.35.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 117
- 2 Variable Name = Event notification timing
- 3 Number of Fields = 3
- 4 List of Fields
 - 4.1 Field Name = Event notification retry time
 - 4.2 Field Data type Name = Time
 - 4.3 Field Name = Maximum update time
 - 4.4 Field Data type Name = Time
 - 4.5 Field Name = Event de-bounce interval
 - 4.6 Field Data type Name = Time
- 5 Array size = Variable

7.3.2.4.36 Write event notification control

7.3.2.4.36.1 Vue d'ensemble du service

Ce service est utilisé pour activer ou désactiver la notification d'événement.

7.3.2.4.36.2 Paramètres de demande

- 1 Numeric Identifier = 118
- 2 Subindex
 - 2.1 Subindex Name = Event number ref
 - 2.2 Subindex Data type = Unsigned8

7.3.2.4.36.3 Paramètres de réponse

- 1 Numeric Identifier = 118
- 2 Subindex
 - 2.1 Subindex Name = Event number ref
 - 2.2 Subindex Data type = Unsigned8

7.3.2.4.36.4 Modèle de Variable

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 118
- 2 Variable Name = Event notification control
- 3 Data type Name = Bit Field
- 4 Array size = Variable

7.3.2.4.37 Write event notification acknowledgement

7.3.2.4.37.1 Vue d'ensemble du service

Ce service est utilisé pour acquitter l'Event Notification (notification d'événement).

7.3.2.4.37.2 Paramètres de demande

- 1 Numeric Identifier = 119

- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.37.3 Paramètres de réponse

- 1 Numeric Identifier = 119
- 2 Subindex
- 2.1 Subindex Name = Event number ref
- 2.2 Subindex Data type = Unsigned8

7.3.2.4.37.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 119
- 2 Variable Name = Event notification acknowledgement
- 3 Number of Fields = 4
- 4 List of Fields
- 4.1 Field Name = Event occurrence time
- 4.2 Field Data type Name = Time
- 4.3 Field Name = Configuration change counter
- 4.4 Field Data type Name = Unsigned16
- 4.5 Field Name = Application process status
- 4.6 Field Data type Name = Bit Field
- 4.7 Field Name = Additional device status
- 4.8 Field Data type Name = Voir Note
- 5 Array size = Variable

L'Additional device status est le même que l'entière variable Structure définie en 7.3.2.4.21.2.

7.3.2.5 Services de gestion de couche

7.3.2.5.1 Write Join_key

7.3.2.5.1.1 Vue d'ensemble du service

Cette commande permet à un outil de maintenance local (par exemple un dispositif tenu à la main) de modifier une Join_key de dispositif de réseau.

7.3.2.5.1.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 768
- 2 Variable Name = Join_key
- 3 Data type Name = Security key

7.3.2.5.2 Read join status

7.3.2.5.2.1 Vue d'ensemble du service

Ce service lit le statut du dispositif de terrain lorsqu'il effectue des transitions à travers le processus de rattachement.

7.3.2.5.2.2 Modèle de Variable**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 769
2	Variable Name	= Join status
3	Number of Fields	= 7
4	List of Fields	
4.1	Field Name	= Wireless mode
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Join status
4.4	Field Data type Name	= Bit Field
4.5	Field Name	= Neighbors
4.6	Field Data type Name	= Unsigned8
4.7	Field Name	= Advertising packets
4.8	Field Data type Name	= Unsigned8
4.9	Field Name	= Max_join_retries
4.10	Field Data type Name	= Unsigned8
4.11	Field Name	= Join_rsp_time
4.12	Field Data type Name	= Time
4.13	Field Name	= Search time
4.14	Field Data type Name	= Time

7.3.2.5.3 Write active_search_shed_time**7.3.2.5.3.1 Vue d'ensemble du service**

Cette commande demande une annonce plus rapide. En général, elle est reçue localement par un membre de réseau à partir d'un outil de maintenance local (par exemple, un dispositif tenu à la main) et est propagée vers le gestionnaire de réseau.

7.3.2.5.3.2 Request variable model**CLASS: Simple VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 770
2	Variable Name	= Active_search_shed_time
3	Data type Name	= Time

7.3.2.5.3.3 Response variable model**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 770
2	Variable Name	= Shed_time
3	Number of Fields	= 3
4	List of Fields	
4.1	Variable Name	= Active_search_shed_time
4.2	Data type Name	= Time
4.3	Variable Name	= Advertise_time
4.4	Data type Name	= Time
4.5	Field Name	= Neighbors
4.6	Field Data type Name	= Unsigned8

7.3.2.5.4 Write join mode configuration

7.3.2.5.4.1 Vue d'ensemble du service

Cette commande permet à un système hôte ou dispositif tenu à la main de forcer un dispositif de terrain à entrer dans le mode active join (rattachement actif).

7.3.2.5.4.2 Modèle de Variable

CLASS:	STRUCTURE VARIABLE
ATTRIBUTES:	
1 Numeric Identifier	= 771
2 Variable Name	= Join mode configuration
3 Number of Fields	= 2
4 List of Fields	
4.1 Field Name	= Join mode
4.2 Field Data type Name	= Enumeration
4.3 Variable Name	= Active_search_shed_time
4.4 Data type Name	= Time

7.3.2.5.5 Read join mode configuration

7.3.2.5.5.1 Vue d'ensemble du service

Cette commande lit la configuration du mode Join (rattachement) à partir d'un dispositif de réseau.

7.3.2.5.5.2 Modèle de Variable

CLASS:	STRUCTURE VARIABLE
ATTRIBUTES:	
1 Numeric Identifier	= 772
2 Variable Name	= Join mode configuration
3 Number of Fields	= 2
4 List of Fields	
4.1 Field Name	= Join mode
4.2 Field Data type Name	= Enumeration
4.3 Variable Name	= Active_search_shed_time
4.4 Data type Name	= Time

7.3.2.5.6 Write network_ID

7.3.2.5.6.1 Vue d'ensemble du service

Cette commande configure le dispositif pour reconnaître le network_ID correct. Il convient que cette commande soit émise seulement par le gestionnaire de réseau ou via le port de maintenance. Le dispositif recevant cette commande est tenu de rejeter cette commande issue de n'importe quelle autre source.

7.3.2.5.6.2 Modèle de Variable

CLASS:	Simple VARIABLE
ATTRIBUTES:	
1 Numeric Identifier	= 773
2 Variable Name	= Network_ID
3 Data type Name	= Unsigned16

7.3.2.5.7 Read network_ID

7.3.2.5.7.1 Vue d'ensemble du service

Cette commande lit la valeur de réglage courante du network_ID dans un dispositif.

7.3.2.5.7.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 774
2	Variable Name	= Network_ID
3	Data type Name	= Unsigned16

7.3.2.5.8 Write network_tag

7.3.2.5.8.1 Vue d'ensemble du service

Cette commande lit le marqueur pour le réseau. Le Network_tag est un proxy pour le network_ID et, de ce fait, le réseau peut être identifié sous une forme textuelle au niveau de l'application.

7.3.2.5.8.2 Modèle de Variable

CLASS: String VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 775
2	Variable Name	= Network_tag
3	Data type Name	= Latin-1
3.1	Octet Length	= 32

7.3.2.5.9 Read network_tag

7.3.2.5.9.1 Vue d'ensemble du service

Cette commande lit la valeur de réglage courante du marqueur du réseau.

7.3.2.5.9.2 Modèle de Variable

CLASS: String VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 776
2	Variable Name	= Network_tag
3	Data type Name	= Latin-1
3.1	Octet Length	= 32

7.3.2.5.10 Read wireless device capabilities

7.3.2.5.10.1 Vue d'ensemble du service

Ce service est utilisé pour déterminer des caractéristiques opérationnelles du dispositif de réseau. Le gestionnaire de réseau est censé utiliser le débit de paquets de crête (Peak) et le temps de Recovery (récupération) pour prendre en charge des dispositifs rechargeables ou récupérateurs d'énergie.

7.3.2.5.10.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 777
2	Variable Name	= Wireless device capabilities
3	Number of Fields	= 8
4	List of Fields	
4.1	Field Name	= Power source
4.2	Field Data type Name	= Enumeration
4.3	Field Name	= Peak packet rate
4.4	Field Data type Name	= Float32
4.5	Field Name	= Load duration
4.6	Field Data type Name	= Time
4.7	Field Name	= Recovery time
4.8	Field Data type Name	= Time
4.9	Field Name	= Good RSL
4.10	Field Data type Name	= Integer8
4.11	Field Name	= Min Keep_alive_time
4.12	Field Data type Name	= Time
4.13	Field Name	= Neighbor count
4.14	Field Data type Name	= Unsigned16
4.15	Field Name	= Packet buffer
4.16	Field Data type Name	= Unsigned16

7.3.2.5.11 Read battery life

7.3.2.5.11.1 Vue d'ensemble du service

Ce service lit l'état courant de la batterie dans un dispositif de réseau.

7.3.2.5.11.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 778
2	Variable Name	= Battery life
3	Data type Name	= Unsigned16

7.3.2.5.12 Report device health

7.3.2.5.12.1 Vue d'ensemble du service

Cette réponse de commande est périodiquement éditée par le dispositif de terrain.

7.3.2.5.12.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 779
2	Variable Name	= Device health
3	Number of Fields	= 7
4	List of Fields	
4.1	Field Name	= Packets generated
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Packets terminated
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= DLL MIC failures
4.6	Field Data type Name	= Unsigned8
4.7	Field Name	= NL MIC failures

- 4.8 Field Data type Name = Unsigned8
- 4.9 Field Name = Power status
- 4.10 Field Data type Name = Enumeration
- 4.11 Field Name = CRC errors
- 4.12 Field Data type Name = Unsigned8
- 4.13 Field Name = Packets missed
- 4.14 Field Data type Name = Unsigned8

7.3.2.5.13 Read neighbor health list

7.3.2.5.13.1 Vue d'ensemble du service

Ce service lit les entrées du tableau Neighbor (voisins) à partir du dispositif de réseau.

7.3.2.5.13.2 Paramètres de demande

- 1 Numeric Identifier = 780
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8

7.3.2.5.13.3 Paramètres de réponse

- 1 Numeric Identifier = 780
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Neighbor count
 - 4.2 Array size Data type = Unsigned8

7.3.2.5.13.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 780
- 2 Variable Name = Neighbor health
- 3 Number of Fields = 6
- 4 List of Fields
 - 4.1 Field Name = Neighbor_nickname
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = Neighbor flags
 - 4.4 Field Data type Name = Bit Field
 - 4.5 Field Name = Average_RSL
 - 4.6 Field Data type Name = Integer8
 - 4.7 Field Name = Packets_transmitted
 - 4.8 Field Data type Name = Unsigned16
 - 4.9 Field Name = Missed_ACK_packets
 - 4.10 Field Data type Name = Unsigned16

- 4.11 Field Name = Packets_received
- 4.12 Field Data type Name = Unsigned16
- 5 Array size = Variable

7.3.2.5.14 Read device nickname

7.3.2.5.14.1 Vue d'ensemble du service

Ce service lit un surnom de dispositif de réseau (adresse courte).

7.3.2.5.14.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 781
- 2 Variable Name = Device nickname
- 3 Data type Name = Unsigned16

7.3.2.5.15 Read session list

7.3.2.5.15.1 Vue d'ensemble du service

Ce service lit les entrées du tableau Session à partir du dispositif de réseau.

7.3.2.5.15.2 Paramètres de demande

- 1 Numeric Identifier = 782
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8

7.3.2.5.15.3 Paramètres de réponse

- 1 Numeric Identifier = 782
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Session count
 - 4.2 Array size Data type = Unsigned8

7.3.2.5.15.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 782
- 2 Variable Name = Session
- 3 Number of Fields = 5
- 4 List of Fields
 - 4.1 Field Name = Session_type

- 4.2 Field Data type Name = Enumeration
- 4.3 Field Name = Peer_nickname
- 4.4 Field Data type Name = Unsigned16
- 4.5 Field Name = Peer_unique_ID
- 4.6 Field Data type Name = Unique ID
- 4.7 Field Name = Peer_nonce_counter
- 4.8 Field Data type Name = Unsigned32
- 4.9 Field Name = My_nonce_counter
- 4.10 Field Data type Name = Unsigned32
- 5 Array size = Variable

7.3.2.5.16 Read superframe list

7.3.2.5.16.1 Vue d'ensemble du service

Ce service lit les entrées du tableau Superframe (superframes) à partir du dispositif de réseau.

7.3.2.5.16.2 Paramètres de demande

- 1 Numeric Identifier = 783
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8

7.3.2.5.16.3 Paramètres de réponse

- 1 Numeric Identifier = 783
- 2 Subindex
 - 2.1 Subindex Name = Table index
 - 2.2 Subindex Data type = Unsigned8
- 3 Element count
 - 3.1 Element count Name = Entry count
 - 3.2 Element count Data type = Unsigned8
- 4 Array size
 - 4.1 Array size Name = Superframe count
 - 4.2 Array size Data type = Unsigned8

7.3.2.5.16.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 783
- 2 Variable Name = Superframe
- 3 Number of Fields = 3
- 4 List of Fields
 - 4.1 Field Name = Superframe_ID
 - 4.2 Field Data type Name = Unsigned8
 - 4.3 Field Name = Num_Slots
 - 4.4 Field Data type Name = Unsigned16
 - 4.5 Field Name = Superframe mode
 - 4.6 Field Data type Name = Bit Field

5 Array size = Variable

7.3.2.5.17 Read link list

7.3.2.5.17.1 Vue d'ensemble du service

Ce service lit les entrées du tableau Link (liaisons) à partir du dispositif de réseau.

7.3.2.5.17.2 Paramètres de demande

1 Numeric Identifier = 784
 2 Subindex
 2.1 Subindex Name = Table index
 2.2 Subindex Data type = Unsigned16
 3 Element count
 3.1 Element count Name = Entry count
 3.2 Element count Data type = Unsigned8

7.3.2.5.17.3 Paramètres de réponse

1 Numeric Identifier = 784
 2 Subindex
 2.1 Subindex Name = Table index
 2.2 Subindex Data type = Unsigned16
 3 Element count
 3.1 Element count Name = Entry count
 3.2 Element count Data type = Unsigned8
 4 Array size
 4.1 Array size Name = Link count
 4.2 Array size Data type = Unsigned16

7.3.2.5.17.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1 Numeric Identifier = 784
 2 Variable Name = Link
 3 Number of Fields = 6
 4 List of Fields
 4.1 Field Name = Link_ID
 4.2 Field Data type Name = Unsigned8
 4.3 Field Name = Slot
 4.4 Field Data type Name = Unsigned16
 4.5 Field Name = Channel_Offset
 4.6 Field Data type Name = Unsigned8
 4.7 Field Name = Neighbor_nickname
 4.8 Field Data type Name = Unsigned16
 4.9 Field Name = Link options
 4.10 Field Data type Name = Bit Field
 4.11 Field Name = Link_type
 4.12 Field Data type Name = Enumeration
 5 Array size = Variable

7.3.2.5.18 Read graph list

7.3.2.5.18.1 Vue d'ensemble du service

Ce service lit un élément dans le tableau Graph (graphes) à partir du dispositif de réseau.

7.3.2.5.18.2 Paramètres de demande

- 1 Numeric Identifier = 785
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned8

7.3.2.5.18.3 Paramètres de réponse

- 1 Numeric Identifier = 785
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned8
- 4 Array size
- 4.1 Array size Name = Graph count
- 4.2 Array size Data type = Unsigned8

7.3.2.5.18.4 Modèle variable de réponse

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 785
- 2 Variable Name = Graph
- 3 Number of Fields = 3
- 4 List of Fields
- 4.1 Field Name = Graph_ID
- 4.2 Field Data type Name = Unsigned16
- 4.3 Field Name = Neighbor count
- 4.4 Field Data type Name = Unsigned8
- 4.5 Field Name = Neighbor list
- 4.6 Field Data type Name = Array of Neighbor_nickname
- 5 Array size = Variable

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Neighbor_nickname
- 2 Format = ARRAY
- 2.1 Number of Array Elements = valeur de Neighbor count
- 2.2 Array Element Data type = Unsigned16

7.3.2.5.19 Read neighbor attribute flag

7.3.2.5.19.1 Vue d'ensemble du service

Cette commande lit les attributs d'un voisin à partir du dispositif de réseau.

7.3.2.5.19.2 Paramètres de demande

- 1 Numeric Identifier = 786
- 2 Additional identifier
- 2.1 Name = Neighbor_nickname

2.2 Data type = Unsigned8

7.3.2.5.19.3 Paramètres de réponse

1 Numeric Identifier = 786
 2 Additional identifier
 2.1 Name = Neighbor_nickname
 2.2 Data type = Unsigned8

7.3.2.5.19.4 Modèle variable de réponse

CLASS: Simple VARIABLE

ATTRIBUTES:

1 Numeric Identifier = 786
 2 Variable Name = Neighbor flags
 3 Data type Name = Bit Field

7.3.2.5.20 Read neighbor signal level

7.3.2.5.20.1 Vue d'ensemble du service

Ce service lit le niveau de signal du tableau Neighbor (voisins) à partir du dispositif de réseau.

7.3.2.5.20.2 Paramètres de demande

1 Numeric Identifier = 787
 2 Subindex
 2.1 Subindex Name = Table index
 2.2 Subindex Data type = Unsigned16
 3 Element count
 3.1 Element count Name = Entry count
 3.2 Element count Data type = Unsigned8

7.3.2.5.20.3 Paramètres de réponse

1 Numeric Identifier = 787
 2 Subindex
 2.1 Subindex Name = Table index
 2.2 Subindex Data type = Unsigned16
 3 Element count
 3.1 Element count Name = Entry count
 3.2 Element count Data type = Unsigned8
 4 Array size
 4.1 Array size Name = Neighbor count
 4.2 Array size Data type = Unsigned16

7.3.2.5.20.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1 Numeric Identifier = 787
 2 Variable Name = Neighbor signal level
 3 Number of Fields = 2
 4 List of Fields
 4.1 Field Name = Neighbor_nickname
 4.2 Field Data type Name = Unsigned16

- 4.3 Field Name = Average_RSL
- 4.4 Field Data type Name = Integer8
- 5 Array size = Variable

7.3.2.5.21 Report path down alarm

7.3.2.5.21.1 Vue d'ensemble du service

Le dispositif édite la réponse d'alarme "path down" (trajet défaillant) en utilisant ce service.

7.3.2.5.21.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 788
- 2 Variable Name = Neighbor_nickname
- 3 Data type Name = Unsigned16

7.3.2.5.22 Report source route failed alarm

7.3.2.5.22.1 Vue d'ensemble du service

Le dispositif édite la réponse d'alarme "source route failed" (chemin de source défaillant) en utilisant ce service.

7.3.2.5.22.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 789
- 2 Variable Name = Source route failure
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Unreachable neighbor
 - 4.2 Field Data type Name = Unsigned16
 - 4.3 Field Name = NL MIC
 - 4.4 Field Data type Name = Unsigned32

7.3.2.5.23 Report graph route failed alarm

7.3.2.5.23.1 Vue d'ensemble du service

Le dispositif édite la réponse d'alarme "graph route failed" (chemin de graphe défaillant) en utilisant ce service.

7.3.2.5.23.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 790
- 2 Variable Name = Graph_ID
- 3 Data type Name = Unsigned16

7.3.2.5.24 Report transport layer failed alarm

7.3.2.5.24.1 Vue d'ensemble du service

Le dispositif édit la réponse d'alarme "transport layer failed" (couche transport défailante) en utilisant ce service.

7.3.2.5.24.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 791
2	Variable Name	= Unreachable peer
3	Data type Name	= Unsigned16

7.3.2.5.25 Write UTC time mapping

7.3.2.5.25.1 Vue d'ensemble du service

Ce service met l'heure de début de l'ASN zéro à l'heure contenue dans cette demande, dans un network device.

7.3.2.5.25.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 793
2	Variable Name	= UTC time mapping
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Date
4.2	Field Data type Name	= Date
4.3	Variable Name	= Time of day
4.4	Data type Name	= Time

7.3.2.5.26 Read UTC time mapping

7.3.2.5.26.1 Vue d'ensemble du service

Ce service lit le temps TUC auquel l'ASN zéro a débuté, comme réglé dans le dispositif de réseau.

7.3.2.5.26.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 794
2	Variable Name	= UTC time mapping
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Date
4.2	Field Data type Name	= Date
4.3	Variable Name	= Time of day
4.4	Data type Name	= Time

7.3.2.5.27 Write timer interval**7.3.2.5.27.1 Vue d'ensemble du service**

Ce service établit la durée pour le temporisateur spécifié par le Timer_type (type de temporisateur) dans un dispositif de réseau.

7.3.2.5.27.2 Paramètres de demande

- 1 Numeric Identifier = 795
- 2 Subindex
- 2.1 Subindex Name = Timer_type
- 2.2 Subindex Data type = Enumeration

7.3.2.5.27.3 Paramètres de réponse

- 1 Numeric Identifier = 795
- 2 Subindex
- 2.1 Subindex Name = Timer_type
- 2.2 Subindex Data type = Enumeration

7.3.2.5.27.4 Modèle de Variable

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 795
- 2 Variable Name = Interval
- 3 Data type Name = Unsigned32
- 4 Array size = 7

7.3.2.5.28 Read timer interval**7.3.2.5.28.1 Vue d'ensemble du service**

Ce service lit la durée pour le temporisateur spécifié par le Timer_type (type de temporisateur) dans un dispositif de réseau.

7.3.2.5.28.2 Paramètres de demande

- 1 Numeric Identifier = 796
- 2 Subindex
- 2.1 Subindex Name = Timer_type
- 2.2 Subindex Data type = Enumeration

7.3.2.5.28.3 Paramètres de réponse

- 1 Numeric Identifier = 796
- 2 Subindex
- 2.1 Subindex Name = Timer_type
- 2.2 Subindex Data type = Enumeration

7.3.2.5.28.4 Modèle de Variable

CLASS: Array of Simple VARIABLE

ATTRIBUTES:

- 5 Numeric Identifier = 796
- 6 Variable Name = Interval
- 7 Data type Name = Unsigned32

8 Array size = 7

7.3.2.5.29 Write radio power output

7.3.2.5.29.1 Vue d'ensemble du service

Ce service établit la puissance de sortie radio pour un dispositif ou une passerelle.

7.3.2.5.29.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 797
2	Variable Name	= Output power
3	Data type Name	= Integer8

7.3.2.5.30 Read radio power output

7.3.2.5.30.1 Vue d'ensemble du service

Ce service lit la puissance de sortie radio dans un dispositif ou une passerelle.

7.3.2.5.30.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 798
2	Variable Name	= Output power
3	Data type Name	= Integer8

7.3.2.5.31 Request timetable

7.3.2.5.31.1 Vue d'ensemble du service

Ce service est utilisé par un dispositif sans fil pour ajouter une connexion de service à un autre dispositif sans fil avec des caractéristiques spécifiées de largeur de bande et de latence.

7.3.2.5.31.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 799
2	Variable Name	= Timetable
3	Number of Fields	= 6
4	List of Fields	
4.1	Field Name	= Timetable_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Timetable_type
4.4	Field Data type Name	= Bit Field
4.5	Field Name	= Application domain
4.6	Field Data type Name	= Enumeration
4.7	Field Name	= Peer_nickname
4.8	Field Data type Name	= Unsigned16
4.9	Variable Name	= Duration
4.10	Data type Name	= Time
4.11	Variable Name	= Route_ID

4.12 Data type Name = Unsigned8

7.3.2.5.32 Read timetable list

7.3.2.5.32.1 Vue d'ensemble du service

Ce service lit les entrées du tableau Timetable (indicateur horaire) à partir du dispositif de réseau.

7.3.2.5.32.2 Paramètres de demande

- 1 Numeric Identifier = 800
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned8
- 3 Element count
- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8

7.3.2.5.32.3 Paramètres de réponse

- 1 Numeric Identifier = 800
- 2 Subindex
- 2.1 Subindex Name = Table index
- 2.2 Subindex Data type = Unsigned8
- 3 Element count
- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8
- 4 Array size
- 4.1 Array size Name = Timetable count
- 4.2 Array size Data type = Unsigned8

7.3.2.5.32.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 800
- 2 Variable Name = Timetable
- 3 Number of Fields = 6
- 4 List of Fields
- 4.1 Field Name = Timetable_ID
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Timetable_type
- 4.4 Field Data type Name = Bit Field
- 4.5 Field Name = Application domain
- 4.6 Field Data type Name = Enumeration
- 4.7 Field Name = Peer_nickname
- 4.8 Field Data type Name = Unsigned16
- 4.9 Field Name = Service_duration
- 4.10 Field Data type Name = Time
- 4.11 Field Name = Route_ID
- 4.12 Field Data type Name = Unsigned8
- 5 Array size = Variable

7.3.2.5.33 Delete timetable connection

7.3.2.5.33.1 Vue d'ensemble du service

Ce service notifie la suppression de Timetable à un dispositif.

7.3.2.5.33.2 Paramètres de demande

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 801
2	Variable name	= Timetable_ID
3	Data type name	= Unsigned8
4	Qualifier Name	= Reason
5	Data type Name	= Enumeration

7.3.2.5.33.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 801
2	Variable name	= Timetable_ID
3	Data type name	= Unsigned8
4	Qualifier Name	= Reason
5	Data type Name	= Enumeration

7.3.2.5.33.4 Modèle variable de réponse

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 801
2	Variable Name	= Timetable count
3	Data type Name	= Unsigned8

7.3.2.5.34 Read route list

7.3.2.5.34.1 Vue d'ensemble du service

Ce service lit les entrées du tableau Route (chemin) à partir du dispositif de réseau.

7.3.2.5.34.2 Paramètres de demande

1	Numeric Identifier	= 802
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8
3	Element count	
3.1	Element count Name	= Entry count
3.2	Element count Data type	= Unsigned8

7.3.2.5.34.3 Paramètres de réponse

1	Numeric Identifier	= 802
2	Subindex	
2.1	Subindex Name	= Table index
2.2	Subindex Data type	= Unsigned8
3	Element count	

- 3.1 Element count Name = Entry count
- 3.2 Element count Data type = Unsigned8
- 4 Array size
- 4.1 Array size Name = Route count
- 4.2 Array size Data type = Unsigned8

7.3.2.5.34.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 802
- 2 Variable Name = Route
- 3 Number of Fields = 5
- 4 List of Fields
- 4.1 Field Name = Routes remaining
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Route_ID
- 4.4 Field Data type Name = Unsigned8
- 4.5 Field Name = Destination nickname
- 4.6 Field Data type Name = Unsigned16
- 4.7 Field Name = Graph_ID
- 4.8 Field Data type Name = Unsigned16
- 4.9 Field Name = Source route flag
- 4.10 Field Data type Name = Unsigned8
- 5 Array size = Variable

7.3.2.5.35 Read source route

7.3.2.5.35.1 Vue d'ensemble du service

Ce service lit le contenu d'un chemin Source particulier.

7.3.2.5.35.2 Paramètres de demande

- 1 Numeric Identifier = 803
- 2 Additional identifier
- 2.1 Name = Route_ID
- 2.2 Data type = Unsigned8

7.3.2.5.35.3 Paramètres de réponse

- 3 Numeric Identifier = 803
- 4 Additional identifier
- 4.1 Name = Route_ID
- 4.2 Data type = Unsigned8

7.3.2.5.35.4 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 803
- 2 Variable Name = Source route
- 3 Number of Fields = 2
- 4 List of Fields
- 4.1 Field Name = Hop count
- 4.2 Field Data type Name = Unsigned8

- 4.3 Field Name = Hop list
- 4.4 Field Data type name = Array of Hop

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Hop
- 2 Format = ARRAY
- 2.1 Number of Array Elements = valeur de Hop count
- 2.2 Array Element Data type = Unsigned16

7.3.2.5.36 Read CCA mode

7.3.2.5.36.1 Vue d'ensemble du service

Ce service lit le mode Évaluation de voie libre (CCA) dans un dispositif de réseau.

7.3.2.5.36.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 804
- 2 Variable Name = CCA mode
- 3 Data type Name = Enumeration

7.3.2.5.37 Write CCA mode

7.3.2.5.37.1 Vue d'ensemble du service

Ce service écrit le mode Évaluation de voie libre (CCA) dans un dispositif de réseau.

7.3.2.5.37.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 805
- 2 Variable Name = CCA mode
- 3 Data type Name = Enumeration

7.3.2.5.38 Read handheld superframe

7.3.2.5.38.1 Vue d'ensemble du service

Ce service est utilisé pour déterminer si la supertrame de dispositif tenu à la main dans un équipement en réseau est active.

7.3.2.5.38.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 806
- 2 Variable Name = Superframe
- 3 Number of Fields = 3
- 4 List of Fields
 - 4.1 Field Name = Superframe_ID
 - 4.2 Field Data type Name = Unsigned8
 - 4.3 Field Name = Num_Slots
 - 4.4 Field Data type Name = Unsigned16

4.5 Field Name = Superframe mode

4.6 Field Data type Name = Bit Field

7.3.2.5.39 Write handheld superframe

7.3.2.5.39.1 Vue d'ensemble du service

Ce service permet à un dispositif tenu à la main de demander à ce que l'équipement en réseau active la supertrame de dispositif tenu à la main afin que la maintenance puisse être effectuée.

7.3.2.5.39.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 807
2	Variable Name	= Superframe
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Superframe_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Superframe mode
4.4	Field Data type Name	= Bit Field

7.3.2.5.40 Read packet hop-to-live

7.3.2.5.40.1 Vue d'ensemble du service

Ce service lit la valeur courante du Hop-to-live configuré pour un dispositif de réseau.

7.3.2.5.40.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 808
2	Variable Name	= Hop-to-live
3	Data type Name	= Unsigned8

7.3.2.5.41 Write packet hop-to-live

7.3.2.5.41.1 Vue d'ensemble du service

Ce service écrit la valeur du Hop-to-live dans un dispositif de réseau.

7.3.2.5.41.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 809
2	Variable Name	= Hop-to-live
3	Data type Name	= Unsigned8

7.3.2.5.42 Read join_priority

7.3.2.5.42.1 Vue d'ensemble du service

Ce service lit la valeur courante de Join_priority dans un dispositif de réseau.

7.3.2.5.42.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 810
2	Variable Name	= Join_priority
3	Data type Name	= Unsigned8

7.3.2.5.43 Write join_priority

7.3.2.5.43.1 Vue d'ensemble du service

Ce service écrit la valeur de Join_priority dans un dispositif de réseau.

7.3.2.5.43.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 811
2	Variable Name	= Join_priority
3	Data type Name	= Unsigned8

7.3.2.5.44 Read receive Priority_threshold

7.3.2.5.44.1 Vue d'ensemble du service

Ce service lit la valeur courante de Receive Priority_threshold (seuil de priorité de réception) dans un dispositif de réseau.

7.3.2.5.44.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 812
2	Variable Name	= Priority_threshold
3	Data type Name	= Unsigned8

7.3.2.5.45 Write receive Priority_threshold

7.3.2.5.45.1 Vue d'ensemble du service

Ce service écrit la valeur de Receive Priority_threshold dans un dispositif de réseau.

7.3.2.5.45.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 813
2	Variable Name	= Priority_threshold
3	Data type Name	= Unsigned8

7.3.2.5.46 Read device list

7.3.2.5.46.1 Vue d'ensemble du service

Ce service lit les entrées dans le tableau Device pour le type de liste spécifié par le "Device code" dans la demande.

7.3.2.5.46.2 Paramètres de demande

- 1 Numeric Identifier = 814
- 2 Additional identifier
- 2.1 Name = Device code
- 2.2 Data type = Enumeration
- 3 Subindex
- 3.1 Subindex Name = Table index
- 3.2 Subindex Data type = Unsigned16
- 4 Element count
- 4.1 Element count Name = Entry count
- 4.2 Element count Data type = Unsigned8

7.3.2.5.46.3 Paramètres de réponse

- 1 Numeric Identifier = 802
- 2 Additional identifier
- 2.1 Name = Device code
- 2.2 Data type = Enumeration
- 3 Subindex
- 3.1 Subindex Name = Table index
- 3.2 Subindex Data type = Unsigned16
- 4 Element count
- 4.1 Element count Name = Entry count
- 4.2 Element count Data type = Unsigned8
- 5 Array size
- 5.1 Array size Name = Device count
- 5.2 Array size Data type = Unsigned16

7.3.2.5.46.4 Modèle de Variable**CLASS:** Array of Simple VARIABLE**ATTRIBUTES:**

- 1 Numeric Identifier = 814
- 2 Variable Name = Device list
- 3 Data type Name = Unique ID
- 4 Array size = Variable

7.3.2.5.47 Add device list entry**7.3.2.5.47.1 Vue d'ensemble du service**

Ce service ajoute un dispositif dans le Tableau Device pour le type de liste spécifié par "Device code".

7.3.2.5.47.2 Paramètres de demande**CLASS:** Qualified Action**ATTRIBUTES:**

- 1 Numeric Identifier = 815
- 2 Variable name = Device code
- 3 Data type name = Enumeration
- 4 Qualifier Name = Unique ID
- 5 Data type Name = Unique ID

7.3.2.5.47.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 815
2	Variable name	= Device code
3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.47.4 Modèle variable de réponse

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 815
2	Variable Name	= Device count
3	Data type Name	= Unsigned16

7.3.2.5.48 Delete device list entry

7.3.2.5.48.1 Vue d'ensemble du service

Ce service supprime un dispositif dans le Tableau Device pour le type de liste spécifié par "Device code" dans une passerelle ou un gestionnaire de réseau.

7.3.2.5.48.2 Paramètres de demande

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 816
2	Variable name	= Device code
3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.48.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 816
2	Variable name	= Device code
3	Data type name	= Enumeration
4	Qualifier Name	= Unique ID
5	Data type Name	= Unique ID

7.3.2.5.48.4 Modèle variable de réponse

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 816
2	Variable Name	= Device count
3	Data type Name	= Unsigned16

7.3.2.5.49 Read channel blacklist

7.3.2.5.49.1 Vue d'ensemble du service

Ce service lit la liste noire de voies courante et la liste noire qui doit être utilisée après le prochain redémarrage.

7.3.2.5.49.2 Modèle de Variable

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 817
2	Variable Name	= Channel blacklist
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Channel count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Current channel map
4.4	Field Data type Name	= Bit Field
4.5	Field Name	= Pending channel map
4.6	Field Data type Name	= Bit Field

7.3.2.5.50 Write channel blacklist

7.3.2.5.50.1 Vue d'ensemble du service

Ce service écrit la carte de voies en attente, qui est la liste noire, dans la passerelle. Il est utilisé après le prochain redémarrage.

7.3.2.5.50.2 Modèle de Variable

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 818
2	Variable Name	= Channel blacklist
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Channel count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Pending channel map
4.4	Field Data type Name	= Bit Field

7.3.2.5.51 Read back-off exponent

7.3.2.5.51.1 Vue d'ensemble du service

Ce service lit la valeur courante de Max_backoff_exp dans un dispositif de réseau.

7.3.2.5.51.2 Modèle de Variable

CLASS: **Simple VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 819
2	Variable Name	= Max_backoff_exp
3	Data type Name	= Unsigned8

7.3.2.5.52 Write back-off exponent

7.3.2.5.52.1 Vue d'ensemble du service

Ce service écrit la valeur de Max_backoff_exp dans un dispositif de réseau.

7.3.2.5.52.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 820
2	Variable Name	= Max_backoff_exp
3	Data type Name	= Unsigned8

7.3.2.5.53 Write network access mode

7.3.2.5.53.1 Vue d'ensemble du service

Ce service écrit la valeur de "Network access mode" (mode d'accès au réseau) dans un dispositif de réseau.

7.3.2.5.53.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 821
2	Variable Name	= Network access mode
3	Data type Name	= Enumeration

7.3.2.5.54 Read network access mode

7.3.2.5.54.1 Vue d'ensemble du service

Ce service lit la valeur courante de network access mode (mode d'accès au réseau) dans un dispositif de réseau.

7.3.2.5.54.2 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 822
2	Variable Name	= Network access mode
3	Data type Name	= Enumeration

7.3.2.5.55 Request session

7.3.2.5.55.1 Vue d'ensemble du service

Ce service demande une session de monodiffusion d'un dispositif tenu à la main à un dispositif.

7.3.2.5.55.2 Paramètres de demande

1	Numeric Identifier	= 823
2	Additional identifier	
2.1	Name	= Peer_nickname
2.2	Data type	= Unsigned16

7.3.2.5.55.3 Paramètres de réponse

- 1 Numeric Identifier = 823
- 2 Additional identifier
- 2.1 Name = Peer_nickname
- 2.2 Data type = Unsigned16

7.3.2.5.55.4 Modèle de Variable**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

- 1 Numeric Identifier = 823
- 2 Variable Name = Session
- 3 Number of Fields = 2
- 4 List of Fields
- 4.1 Field Name = Peer_nonce_counter
- 4.2 Field Data type Name = Unsigned32
- 4.3 Field Name = Join_key
- 4.4 Field Data type Name = Security key

7.3.2.6 Services de passerelle et de gestionnaire de réseau**7.3.2.6.1 Read network device identity using Unique ID****7.3.2.6.1.1 Vue d'ensemble du service**

Ce service lit dans la passerelle les informations relatives à l'identité du dispositif avec le Device ID correspondant.

7.3.2.6.1.2 Paramètres de demande

- 1 Numeric Identifier = 832
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.1.3 Paramètres de réponse

- 1 Numeric Identifier = 832
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.1.4 Modèle de Variable**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

- 1 Numeric Identifier = 832
- 2 Variable Name = Device identity
- 3 Number of Fields = 2
- 4 List of Fields
- 4.1 Field Name = Device nickname
- 4.2 Field Data type Name = Unsigned16
- 4.3 Field Name = Long_tag
- 4.4 Field Data type Name = Latin-1
- 4.4.1 Octet Length = 32

7.3.2.6.2 Read device's neighbor health

7.3.2.6.2.1 Vue d'ensemble du service

Ce service lit, à partir de la passerelle, les informations relatives à la qualité de la connexion vers chaque voisin actif ou voisin potentiel pour le dispositif spécifié dans la demande.

7.3.2.6.2.2 Paramètres de demande

- 1 Numeric Identifier = 833
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID
- 3 Subindex
- 3.1 Subindex Name = Table index
- 3.2 Subindex Data type = Unsigned8
- 4 Element count
- 4.1 Element count Name = Entry count
- 4.2 Element count Data type = Unsigned8

7.3.2.6.2.3 Paramètres de réponse

- 1 Numeric Identifier = 833
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID
- 3 Subindex
- 3.1 Subindex Name = Table index
- 3.2 Subindex Data type = Unsigned8
- 4 Element count
- 4.1 Element count Name = Entry count
- 4.2 Element count Data type = Unsigned8

7.3.2.6.2.4 Modèle de Variable

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 833
- 2 Variable Name = Neighbor health
- 3 Number of Fields = 5
- 4 List of Fields
- 4.1 Field Name = Neighbor_nickname
- 4.2 Field Data type Name = Unsigned16
- 4.3 Field Name = Average_RSL
- 4.4 Field Data type Name = Integer8
- 4.5 Field Name = Packets_transmitted
- 4.6 Field Data type Name = Unsigned32
- 4.7 Field Name = Missed_ACK_packets
- 4.8 Field Data type Name = Unsigned32
- 4.9 Field Name = Packets_received
- 4.10 Field Data type Name = Unsigned32
- 5 Array size = Variable

7.3.2.6.3 Read network topology information

7.3.2.6.3.1 Vue d'ensemble du service

Ce service lit un élément dans le tableau Graph pour le dispositif de réseau spécifié à partir de la passerelle. Par lecture de la commande 814, la couche application peut déterminer le nombre de graphes auxquels un dispositif particulier participe en utilisant "Read graph list" (voir 7.3.2.5.18). Ce service peut alors être émis autant de fois que nécessaire pour déterminer les Graph_ID spécifiques auxquels le dispositif participe.

7.3.2.6.3.2 Paramètres de demande

1	Numeric Identifier	= 834
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID
3	Subindex	
3.1	Subindex Name	= Table index
3.2	Subindex Data type	= Unsigned16

7.3.2.6.3.3 Paramètres de réponse

1	Numeric Identifier	= 834
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID
3	Subindex	
3.1	Subindex Name	= Table index
3.2	Subindex Data type	= Unsigned16
5	Array size	
5.1	Array size Name	= Graph count
5.2	Array size Data type	= Unsigned16

7.3.2.6.3.4 Modèle variable de réponse

CLASS: Array of Structure VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 834
2	Variable Name	= Graph
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Graph_ID
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Neighbor count
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= Neighbor list
4.6	Field Data type Name	= Array of Neighbor
5	Array size	= Variable

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Array of Neighbor
2	Format	= ARRAY
2.1	Number of Array Elements	= valeur de Neighbor count
2.2	Array Element Data type	= Unsigned16

7.3.2.6.4 Read publish data message list

7.3.2.6.4.1 Vue d'ensemble du service

Ce service lit, dans la passerelle, la liste de message Publish data à laquelle participe le dispositif spécifié, y compris le nombre de commandes de mode Publish data auxquelles le dispositif spécifié participe. L'application demandeuse peut utiliser ce service pour dresser la liste de tout le trafic de mode Publish data dans le système.

7.3.2.6.4.2 Paramètres de demande

1	Numeric Identifier	= 835
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.4.3 Paramètres de réponse

1	Numeric Identifier	= 835
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.4.4 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 835
2	Variable Name	= Publish data message list
3	Number of Fields	= 2
4	List of Fields	
4.1	Field Name	= Publish data count
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Publish data
4.4	Field Data type Name	= Array of Publish data

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Array of Publish data
2	Format	= ARRAY
2.1	Number of Array Elements	= valeur de Publish data count
2.2	Array Element Data type	= Publish data

CLASS: Data type

ATTRIBUTES:

1	Data type Name	= Publish data
2	Format	= STRUCTURE
2.1	Number of Fields	= 2
2.2	List of Fields	
2.2.1	Field Name	= Publish command
2.2.2	Field Data type Name	= Unsigned16
2.2.3	Field Name	= Packet count
2.2.4	Field Data type Name	= Unsigned32

7.3.2.6.5 Flush cached responses for a device

7.3.2.6.5.1 Vue d'ensemble du service

Ce service purge les réponses mises en cache dans la passerelle, pour le dispositif spécifié.

7.3.2.6.5.2 Paramètres de demande

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 836
2	Variable name	= Unique ID
3	Data type name	= Unique ID

7.3.2.6.6 Write update notification bit mask for a device

7.3.2.6.6.1 Vue d'ensemble du service

Ce service enregistre un client de passerelle pour des mises de jour de notifications.

7.3.2.6.6.2 Paramètres de demande

1	Numeric Identifier	= 837
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.6.3 Paramètres de réponse

1	Numeric Identifier	= 837
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.6.4 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 837
2	Field Name	= Notification mask
3	Field Data type Name	= Bit Field

7.3.2.6.7 Read update notification bit mask for a device

7.3.2.6.7.1 Vue d'ensemble du service

Ce service lit dans la passerelle la liste du masque de notifications de mise à jour pour le dispositif spécifié.

7.3.2.6.7.2 Paramètres de demande

1	Numeric Identifier	= 838
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.7.3 Paramètres de réponse

- 1 Numeric Identifier = 838
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.7.4 Modèle de Variable

CLASS: Simple VARIABLE

ATTRIBUTES:

- 4 Numeric Identifier = 838
- 5 Field Name = Notification mask
- 6 Field Data type Name = Bit Field

7.3.2.6.8 Read change notification

7.3.2.6.8.1 Vue d'ensemble du service

Cette demande est envoyée par l'application client à la passerelle. La réponse énumère les changements pour le dispositif spécifié dans la demande.

7.3.2.6.8.2 Paramètres de demande

- 1 Numeric Identifier = 839
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.8.3 Paramètres de réponse

- 1 Numeric Identifier = 839
- 2 Additional identifier
- 2.1 Name = Unique ID
- 2.2 Data type = Unique ID

7.3.2.6.8.4 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 839
- 2 Variable Name = change notification list
- 3 Number of Fields = 2
- 4 List of Fields
- 4.1 Field Name = Change count
- 4.2 Field Data type Name = Unsigned8
- 4.3 Field Name = Change notification
- 4.4 Field Data type Name = Array of Change notification

CLASS: Data type

ATTRIBUTES:

- 1 Data type Name = Array of Change notification
- 2 Format = ARRAY
- 2.1 Number of Array Elements = valeur de Change count
- 2.2 Array Element Data type = Unsigned16

7.3.2.6.9 Read network device statistics

7.3.2.6.9.1 Vue d'ensemble du service

Ce service lit dans la passerelle les statistiques du dispositif spécifié.

7.3.2.6.9.2 Paramètres de demande

1	Numeric Identifier	= 840
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.9.3 Paramètres de réponse

1	Numeric Identifier	= 840
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.9.4 Modèle variable de réponse

CLASS:

STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 840
2	Variable Name	= Device statistics
3	Number of Fields	= 13
4	List of Fields	
4.1	Field Name	= Graph count
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Frame count
4.4	Field Data type Name	= Unsigned16
4.5	Field Name	= Link count
4.6	Field Data type Name	= Unsigned16
4.7	Field Name	= Neighbor count
4.8	Field Data type Name	= Unsigned8
4.9	Field Name	= Latency
4.10	Field Data type Name	= Time
4.11	Field Name	= Join count
4.12	Field Data type Name	= Unsigned16
4.13	Field Name	= Join date
4.14	Field Data type Name	= Date
4.15	Field Name	= Join time
4.16	Field Data type Name	= Time
4.17	Field Name	= Packets generated
4.18	Field Data type Name	= Unsigned32
4.19	Field Name	= Packets terminated
4.20	Field Data type Name	= Unsigned32
4.21	Field Name	= DLL MIC failures
4.22	Field Data type Name	= Unsigned32
4.23	Field Name	= NL MIC failures
4.24	Field Data type Name	= Unsigned32
4.25	Field Name	= CRC errors
4.26	Field Data type Name	= Unsigned32
4.27	Field Name	= Packets missed

- 4.28 Field Data type Name = Unsigned32
- 4.29 Field Name = Nonce counter missed
- 4.30 Field Data type Name = Unsigned32
- 4.31 Field Name = Latency
- 4.32 Field Data type Name = Time

7.3.2.6.10 Read network device identity using nickname

7.3.2.6.10.1 Vue d'ensemble du service

Ce service lit dans la passerelle les informations relatives à l'identité du dispositif avec le "Device nickname" (surnom de dispositif) correspondant.

7.3.2.6.10.2 Paramètres de demande

- 1 Numeric Identifier = 841
- 2 Additional identifier
 - 2.1 Name = Device_nickname
 - 2.2 Data type = Unsigned16

7.3.2.6.10.3 Paramètres de réponse

- 1 Numeric Identifier = 841
- 2 Additional identifier
 - 2.1 Name = Device_nickname
 - 2.2 Data type = Unsigned16

7.3.2.6.10.4 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 841
- 2 Variable Name = Device identity
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Unique ID
 - 4.2 Field Data type Name = Unique ID
 - 4.3 Field Name = Long_tag
 - 4.4 Field Data type Name = Latin-1
 - 4.4.1 Octet Length = 32

7.3.2.6.11 Write network device's scheduling flags

7.3.2.6.11.1 Vue d'ensemble du service

Ce service permet à l'utilisateur de demander une considération spéciale pour un dispositif lorsque le gestionnaire de réseau crée des programmes.

7.3.2.6.11.2 Paramètres de demande

- 1 Numeric Identifier = 842
- 2 Additional identifier
 - 2.1 Name = Unique ID
 - 2.2 Data type = Unique ID

7.3.2.6.11.3 Paramètres de réponse

1	Numeric Identifier	= 842
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.11.4 Modèle de Variable**CLASS:** Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 842
2	Variable Name	= Scheduling flags
3	Data type Name	= Bit Field

7.3.2.6.12 Read network device's scheduling flags**7.3.2.6.12.1 Vue d'ensemble du service**

Ce service lit les attributs du dispositif de réseau que le gestionnaire de réseau peut prendre en considération lors de la création de programmes.

7.3.2.6.12.2 Paramètres de demande

1	Numeric Identifier	= 843
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.12.3 Paramètres de réponse

1	Numeric Identifier	= 843
2	Additional identifier	
2.1	Name	= Unique ID
2.2	Data type	= Unique ID

7.3.2.6.12.4 Modèle de Variable**CLASS:** Simple VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 843
2	Variable Name	= Scheduling flags
3	Data type Name	= Bit Field

7.3.2.6.13 Read network constraints**7.3.2.6.13.1 Vue d'ensemble du service**

Ce service lit le réglage courant de la stratégie de gestion de réseau et le nombre de messages de demande / réponse par 10 s.

7.3.2.6.13.2 Modèle de Variable**CLASS:** STRUCTURE VARIABLE**ATTRIBUTES:**

1	Numeric Identifier	= 844
2	Variable Name	= Network constraints
3	Number of Fields	= 2

4 List of Fields

- 4.1 Field Name = Optimization flags
- 4.2 Field Data type Name = Bit Field
- 4.3 Field Name = Message pair count
- 4.4 Field Data type Name = Unsigned8

7.3.2.6.14 Write network constraints

7.3.2.6.14.1 Vue d'ensemble du service

Ce service écrit le réglage courant de la stratégie de gestion de réseau et le nombre de messages de demande / réponse par 10 s.

7.3.2.6.14.2 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

- 1 Numeric Identifier = 845
- 2 Variable Name = Network constraints
- 3 Number of Fields = 2
- 4 List of Fields
 - 4.1 Field Name = Optimization flags
 - 4.2 Field Data type Name = Bit Field
 - 4.3 Field Name = Message pair count
 - 4.4 Field Data type Name = Unsigned8

7.3.2.7 Services de configuration de gestion réseau

7.3.2.7.1 Disconnect device

7.3.2.7.1.1 Vue d'ensemble du service

Ce service permet au gestionnaire de réseau de faire sortir de force un dispositif du réseau.

7.3.2.7.1.2 Paramètres de demande

CLASS: Qualified Action

ATTRIBUTES:

- 1 Numeric Identifier = 960
- 2 Variable name = Disconnect cause
- 3 Data type name = Enumeration

7.3.2.7.1.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

- 1 Numeric Identifier = 960
- 2 Variable name = Disconnect cause
- 3 Data type name = Enumeration

7.3.2.7.2 Write network key

7.3.2.7.2.1 Vue d'ensemble du service

Ce service écrit la clé de réseau dans un dispositif de réseau.

7.3.2.7.2.2 Modèle de Variable**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 961
2	Variable Name	= Network key
3	Number of Fields	= 2
4	List of Fields	
4.1	Variable Name	= Join_key
4.2	Data type Name	= Security key
4.3	Field Name	= Key change time
4.4	Field Data type Name	= Unsigned40

7.3.2.7.3 Write device nickname**7.3.2.7.3.1 Vue d'ensemble du service**

Ce service écrit un surnom de dispositif de réseau.

7.3.2.7.3.2 Modèle de Variable**CLASS: Simple VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 962
2	Variable Name	= Device nickname
3	Data type Name	= Unsigned16

7.3.2.7.4 Write session**7.3.2.7.4.1 Vue d'ensemble du service**

Ce service lit les paramètres de session indispensables pour établir une session entre le dispositif de destination (de cette demande) et le dispositif homologué spécifié dans la demande.

7.3.2.7.4.2 Modèle de variable de demande**CLASS: STRUCTURE VARIABLE****ATTRIBUTES:**

1	Numeric Identifier	= 963
2	Variable Name	= Session entry
3	Number of Fields	= 7
3.1	Field Name	= Session_type
3.2	Field Data type Name	= Enumeration
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Peer_unique_ID
3.6	Field Data type Name	= Unique ID
3.7	Field Name	= Peer_nonce_counter
3.8	Field Data type Name	= Unsigned32
3.9	Field Name	= Session_key
3.10	Field Data type Name	= Security key
3.11	Field Name	= Reserved
3.12	Field Data type Name	= Unsigned8
3.13	Field Name	= Session change time
3.14	Field Data type Name	= Unsigned40

7.3.2.7.4.3 Modèle variable de réponse

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 963
2	Variable Name	= Session entry
3	Number of Fields	= 7
3.1	Field Name	= Session_type
3.2	Field Data type Name	= Enumeration
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Peer_unique_ID
3.6	Field Data type Name	= Unique ID
3.7	Field Name	= Peer_nonce_counter
3.8	Field Data type Name	= Unsigned32
3.9	Field Name	= Session_key
3.10	Field Data type Name	= Security key
3.11	Field Name	= Session count
3.12	Field Data type Name	= Unsigned8
3.13	Field Name	= Session change time
3.14	Field Data type Name	= Unsigned40

7.3.2.7.5 Delete session

7.3.2.7.5.1 Vue d'ensemble du service

Ce service supprime une session de bout en bout dans un dispositif de réseau.

7.3.2.7.5.2 Paramètres de demande

CLASS: **Qualified Action**

ATTRIBUTES:

1	Numeric Identifier	= 964
2	Variable name	= Session_type
3	Data type name	= Enumeration
4	Qualifier Name	= Peer_nickname
5	Data type Name	= Unsigned16

7.3.2.7.5.3 Paramètres de réponse

CLASS: **Qualified Action**

ATTRIBUTES:

1	Numeric Identifier	= 964
2	Variable name	= Session_type
3	Data type name	= Enumeration
4	Qualifier Name	= Peer_nickname
5	Data type Name	= Unsigned16

7.3.2.7.5.4 Modèle variable de réponse

CLASS: **Simple VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 964
2	Variable Name	= Session count
3	Data type Name	= Unsigned8

7.3.2.7.6 Write superframe

7.3.2.7.6.1 Vue d'ensemble du service

Ce service écrit ou modifie une Superframe dans un dispositif de réseau.

7.3.2.7.6.2 Modèle de variable de demande

CLASS:	STRUCTURE VARIABLE
ATTRIBUTES:	
1 Numeric Identifier	= 965
2 Variable Name	= Superframe entry
3 Number of Fields	= 5
3.1 Field Name	= Superframe_ID
3.2 Field Data type Name	= Unsigned8
3.3 Field Name	= Num_Slots
3.4 Field Data type Name	= Unsigned16
3.5 Field Name	= Superframe mode
3.6 Field Data type Name	= Bit Field
3.7 Field Name	= Reserved
3.8 Field Data type Name	= Unsigned8
3.9 Field Name	= Superframe change time
3.10 Field Data type Name	= Unsigned40

7.3.2.7.6.3 Modèle variable de réponse

CLASS:	STRUCTURE VARIABLE
ATTRIBUTES:	
1 Numeric Identifier	= 965
2 Variable Name	= Superframe entry
3 Number of Fields	= 5
3.1 Field Name	= Superframe_ID
3.2 Field Data type Name	= Unsigned8
3.3 Field Name	= Num_Slots
3.4 Field Data type Name	= Unsigned16
3.5 Field Name	= Superframe mode
3.6 Field Data type Name	= Bit Field
3.7 Field Name	= Superframe count
3.8 Field Data type Name	= Unsigned8
3.9 Field Name	= Superframe change time
3.10 Field Data type Name	= Unsigned40

7.3.2.7.7 Delete superframe

7.3.2.7.7.1 Vue d'ensemble du service

Ce service supprime une superframe dans un dispositif de réseau.

7.3.2.7.7.2 Paramètres de demande

CLASS:	Qualified Action
ATTRIBUTES:	
1 Numeric Identifier	= 966
2 Variable name	= Superframe_ID
3 Data type name	= Enumeration

7.3.2.7.7.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 966
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration

7.3.2.7.7.4 Modèle variable de réponse

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 966
2	Field Name	= Superframe count
3	Field Data type Name	= Unsigned8

7.3.2.7.8 Write link

7.3.2.7.8.1 Vue d'ensemble du service

Ce service ajoute une affectation de liaison à un dispositif de réseau.

7.3.2.7.8.2 Modèle de variable de demande

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 967
2	Variable Name	= Link entry
3	Number of Fields	= 6
3.1	Field Name	= Superframe_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Slot
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Channel_Offset
3.6	Field Data type Name	= Unsigned8
3.7	Field Name	= Neighbor
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Link options
3.10	Field Data type Name	= Bit Field
3.11	Field Name	= Link_type
3.12	Field Data type Name	= Enumeration

7.3.2.7.8.3 Modèle variable de réponse

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 967
2	Variable Name	= Link entry
3	Number of Fields	= 7
3.1	Field Name	= Superframe_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Slot
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Channel_Offset
3.6	Field Data type Name	= Unsigned8

3.7	Field Name	= Neighbor
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Link options
3.10	Field Data type Name	= Bit Field
3.11	Field Name	= Link_type
3.12	Field Data type Name	= Enumeration
3.13	Field Name	= Link count
3.14	Field Data type Name	= Unsigned16

7.3.2.7.9 Delete link

7.3.2.7.9.1 Vue d'ensemble du service

Ce service supprime une affectation de liaison dans un dispositif de réseau.

7.3.2.7.9.2 Paramètres de demande

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 968
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration
4	Qualifier Name	= Slot
5	Data type Name	= Unsigned16
6	Qualifier Name	= Reason
7	Data type Name	= Unsigned16

7.3.2.7.9.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 968
2	Variable name	= Superframe_ID
3	Data type name	= Enumeration
4	Qualifier Name	= Slot
5	Data type Name	= Unsigned16
6	Qualifier Name	= Neighbor
7	Data type Name	= Unsigned16

7.3.2.7.9.4 Modèle variable de réponse

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 968
2	Field Name	= Link count
3	Field Data type Name	= Unsigned16

7.3.2.7.10 Write graph connection

7.3.2.7.10.1 Vue d'ensemble du service

Ce service ajoute une affectation de connexion de graphe à un dispositif de réseau.

7.3.2.7.10.2 Modèle de variable de demande

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 969
2	Variable Name	= Connection entry
3	Number of Fields	= 2
3.1	Field Name	= Graph_ID
3.2	Field Data type Name	= Unsigned16
3.3	Field Name	= Neighbor_nickname
3.4	Field Data type Name	= Unsigned16

7.3.2.7.10.3 Modèle variable de réponse

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 969
2	Variable Name	= Connection entry
3	Number of Fields	= 3
3.1	Field Name	= Graph_ID
3.2	Field Data type Name	= Unsigned16
3.3	Field Name	= Neighbor_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Connection count
3.6	Field Data type Name	= Unsigned16

7.3.2.7.11 Delete graph connection

7.3.2.7.11.1 Vue d'ensemble du service

Ce service supprime une affectation de connexion de graphe dans un dispositif de réseau.

7.3.2.7.11.2 Paramètres de demande

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 970
2	Variable name	= Graph_ID
3	Data type name	= Unsigned16
4	Qualifier Name	= Neighbor_nickname
5	Data type Name	= Unsigned16

7.3.2.7.11.3 Paramètres de réponse

CLASS: Qualified Action

ATTRIBUTES:

1	Numeric Identifier	= 970
2	Variable name	= Graph_ID
3	Data type name	= Unsigned16
4	Qualifier Name	= Neighbor_nickname
5	Data type Name	= Unsigned16

7.3.2.7.11.4 Delete response variable model

CLASS: Simple VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 970
2	Field Name	= Connection count
3	Field Data type Name	= Unsigned16

7.3.2.7.12 Write neighbor attribute flag

7.3.2.7.12.1 Vue d'ensemble du service

Ce service met les attributs d'un voisin à un dispositif de réseau.

7.3.2.7.12.2 Modèle de Variable

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 971
2	Variable sub index	Absent
3	Variable Name	= Neighbor attribute
4	Number of Fields	= 2
4.1	Field Name	= Neighbor_nickname
4.2	Field Data type Name	= Unsigned16
4.3	Field Name	= Neighbor flags
4.4	Field Data type Name	= Bit Field

7.3.2.7.13 Write network suspend

7.3.2.7.13.1 Vue d'ensemble du service

Ce service place le réseau entier dans un état de non-émission "repos". La commande peut être envoyée à tous les dispositifs sur le réseau ou un dispositif individuel en fonction de l'adressage utilisé.

7.3.2.7.13.2 Modèle de Variable

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 972
2	Variable sub index	Absent
3	Variable Name	= Network suspend
4	Number of Fields	= 2
4.1	Field Name	= Suspend time
4.2	Field Data type Name	= Unsigned40
4.3	Field Name	= Resume time
4.4	Field Data type Name	= Unsigned40

7.3.2.7.14 Write timetable

7.3.2.7.14.1 Vue d'ensemble du service

Ce service ajoute un nouveau Timetable à un dispositif de réseau. Le dispositif de réseau peut commencer à utiliser ce Service après avoir traité cette commande.

7.3.2.7.14.2 Modèle de variable de demande

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 973
2	Variable Name	= Timetable entry

3	Number of Fields	= 6
3.1	Field Name	= Timetable_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Timetable_type
3.4	Field Data type Name	= Bit Field
3.5	Field Name	= Application domain
3.6	Field Data type Name	= Enumeration
3.7	Field Name	= Peer_nickname
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Service_duration
3.10	Field Data type Name	= Time
3.11	Field Name	= Route_ID
3.12	Field Data type Name	= Unsigned8

7.3.2.7.14.3 Modèle variable de réponse

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 973
2	Variable Name	= Timetable entry
3	Number of Fields	= 7
3.1	Field Name	= Timetable_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Timetable_type
3.4	Field Data type Name	= Bit Field
3.5	Field Name	= Application domain
3.6	Field Data type Name	= Enumeration
3.7	Field Name	= Peer_nickname
3.8	Field Data type Name	= Unsigned16
3.9	Field Name	= Service_duration
3.10	Field Data type Name	= Time
3.11	Field Name	= Route_ID
3.12	Field Data type Name	= Unsigned8
3.13	Field Name	= Service count
3.14	Field Data type Name	= Unsigned8

7.3.2.7.15 Write route

7.3.2.7.15.1 Vue d'ensemble du service

Ce service établit ou modifie un chemin pour un Service particulier dans un dispositif de réseau.

7.3.2.7.15.2 Modèle de variable de demande

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 974
2	Variable Name	= Route entry
3	Number of Fields	= 3
3.1	Field Name	= Route_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Graph_ID

3.6 Field Data type Name = Unsigned16

7.3.2.7.15.3 Modèle de Variable

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 974
2	Variable Name	= Route entry
3	Number of Fields	= 4
3.1	Field Name	= Route_ID
3.2	Field Data type Name	= Unsigned8
3.3	Field Name	= Peer_nickname
3.4	Field Data type Name	= Unsigned16
3.5	Field Name	= Graph_ID
3.6	Field Data type Name	= Unsigned16
3.7	Field Name	= Route count
3.8	Field Data type Name	= Unsigned8

7.3.2.7.16 Delete route

7.3.2.7.16.1 Vue d'ensemble du service

Ce service supprime une entrée Route dans un dispositif de réseau.

7.3.2.7.16.2 Paramètres de demande

CLASS: Simple Variable Action

ATTRIBUTES:

1	Numeric Identifier	= 975
2	Variable Name	= Route_ID
3	Data type Name	= Unsigned8

7.3.2.7.16.3 Paramètres de réponse

CLASS: Simple Variable Action

ATTRIBUTES:

1	Numeric Identifier	= 975
2	Variable Name	= Route_ID
3	Data type Name	= Unsigned8

7.3.2.7.16.4 Modèle variable de réponse

CLASS: STRUCTURE VARIABLE

ATTRIBUTES:

1	Numeric Identifier	= 975
2	Field Name	= Route count
3	Field Data type Name	= Unsigned8

7.3.2.7.17 Write source route

7.3.2.7.17.1 Vue d'ensemble du service

Ce service écrit un chemin Source dans un dispositif de réseau. Les adresses de diffusion ne sont pas des valeurs d'adresse légales dans cette demande et ne sont pas incluses dans les chemins Source ou les demandes ou réponses pour cette commande.

7.3.2.7.17.2 Modèle de variable de demande

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 976
2	Variable Name	= Source route entry
3	Number of Fields	= 3
4	List of Fields	
4.1	Field Name	= Route_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Hop count
4.4	Field Data type Name	= Unsigned8
4.5	Field Name	= Hop list
4.6	Field Data type Name	= Array of Hop

CLASS: **Data type**

ATTRIBUTES:

1	Data type Name	= Array of Hop
2	Format	= ARRAY
2.1	Number of Array Elements	= valeur de Hop count
2.2	Array Element Data type	= Unsigned16

7.3.2.7.17.3 Modèle variable de réponse

CLASS: **STRUCTURE VARIABLE**

ATTRIBUTES:

1	Numeric Identifier	= 976
2	Variable Name	= Source route entry
3	Number of Fields	= 4
4	List of Fields	
4.1	Field Name	= Route_ID
4.2	Field Data type Name	= Unsigned8
4.3	Field Name	= Hop count
4.4	Field Data type Name	= Unsigned8
4.5	Field Name	= Hop list
4.6	Field Data type Name	= Array of Hop
4.7	Field Name	= Source route count
4.8	Field Data type Name	= Unsigned8

7.3.2.7.18 Delete source route

7.3.2.7.18.1 Vue d'ensemble du service

Ce service supprime une entrée chemin de source dans un dispositif de réseau.

7.3.2.7.18.2 Paramètres de demande

CLASS: **Simple Variable Action**

ATTRIBUTES:

4	Numeric Identifier	= 977
5	Variable Name	= Route_ID
6	Data type Name	= Unsigned8

7.3.2.7.18.3 Paramètres de réponse

CLASS: Simple Variable Action

ATTRIBUTES:

4	Numeric Identifier	= 977
5	Variable Name	= Route_ID
6	Data type Name	= Unsigned8

7.3.3 AR

7.3.3.1 ASE de relation d'applications

7.3.3.1.1 Vue d'ensemble

Les relations d'applications (AR) sont constituées d'un ensemble de points d'extrémité de classes compatibles. Un point d'extrémité de chaque AR doit être de classe Master (Maître) et l'autre extrémité de classe Slave (Esclave). Chaque dispositif peut avoir une seule instance d'un AREP de classe Slave.

Un dispositif peut utiliser plus d'un AREP de classe Maître pour communiquer avec plusieurs AREP de classe Esclave. L'utilisateur au niveau de l'AREP de classe Maître fournit l'identification de l'AREP de classe Esclave comme un paramètre dans la primitive "request".

7.3.3.1.2 AREP de classe Maître

7.3.3.1.2.1 Modèle formel

ASE:	AR ASE
CLASS:	Master
CLASS ID:	Pas utilisé
PARENT CLASS:	Top
ATTRIBUTES:	
1 (m) Attribut:	Adresses
SERVICES:	
1 (m) OpService:	AR-Get Attribute
2 (m) OpService:	AR-Set Attribute

7.3.3.1.2.2 Attributs

Adresses

Cet attribut spécifie l'adresse réseau du dispositif maître.

7.3.3.1.3 AREP de classe Esclave

7.3.3.1.3.1 Modèle formel

ASE:	AR ASE
CLASS:	Slave
CLASS ID:	Pas utilisé
PARENT CLASS:	Top
ATTRIBUTES:	
1 (m) Physical layer type	Wireless
1.1 (m) Attribut:	Nickname
1.2 (m) Attribut:	Unique ID
SERVICES:	
1 (m) OpService:	AR-Get Attribute
2 (m) OpService:	AR-Set Attribute

7.3.3.1.3.2 Attributs

Nickname

Cet attribut spécifie l'adresse réseau du dispositif Slave. Le dispositif répond uniquement à cette adresse dans une trame Identify reçue en provenance d'un dispositif Master.

Unique ID

Cet attribut est utilisé pour former l'adresse réseau du dispositif Slave. Le dispositif répond uniquement à cette adresse dans toutes les trames reçues en provenance d'un dispositif Master. Une valeur de cet attribut avec des zéros partout doit être réservée comme adresse de diffusion.

7.3.3.1.4 Spécifications de services ASE de relation d'applications

7.3.3.1.4.1 Vue d'ensemble

Le paragraphe contient la définition de services qui sont propres à cet ASE. Les services sont

- AR Get Attribute et
- AR Set Attribute.

7.3.3.1.4.2 Service AR-Get attribute

7.3.3.1.4.2.1 Vue d'ensemble

Ce service confirmé est utilisé pour lire localement la valeur d'attributs d'un AREP.

7.3.3.1.4.2.2 Paramètres de service

Les paramètres de service pour le Service AR Get Attribute sont montrés au Tableau 73.

Tableau 73 – Paramètres du service AR get attributes

Nom de paramètre	Req	Cnf
Argument	M	
Attribute index	M	
Result (+)		S
Valeur		C
Result (-)		S
Status		C
NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa précédente primitive "request" correspondante est une matière locale.		

Argument

L'argument contient les paramètres de la demande de service.

Attribute index

Ce paramètre identifie l'attribut..

Result (+)

Ce paramètre de type sélection indique que la demande a réussi.

Valeur

Ce paramètre est retourné avec la valeur d'attribut demandée si la demande a réussi.

Result (-)

Ce paramètre de type sélection indique que la demande a échoué.

Status

Si la demande a échoué, Status est retourné indiquant la raison de l'échec. Les valeurs possibles sont: Illegal attribute index (indice d'attribut illégal), Unknown value (valeur inconnue).

7.3.3.1.4.3 Service AR Set attributes**7.3.3.1.4.3.1 Vue d'ensemble**

Ce service confirmé est utilisé pour établir localement la valeur courante d'un AREP.

7.3.3.1.4.3.2 Paramètres de service

Les paramètres de service pour le Service AR Set Attribute sont montrés au Tableau 74.

Tableau 74 – Paramètres du service AR set attributes

Nom de paramètre	req	Cnf
Argument	M	
Attribute index	M	
Valeur	M	
Résultat		
Status		M
NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa précédente primitive "request" correspondante est une matière locale.		

Argument

L'argument contient les paramètres de la demande de service.

Attribute index

Ce paramètre identifie l'attribut à mettre à jour.

Valeur

Ce paramètre contient la nouvelle valeur de l'attribut.

Status

Comme résultat de la demande, Status est retourné indiquant OK ou la raison de l'échec. Les valeurs possibles sont: OK, Illegal attribute index, Illegal value (valeur illégale).

7.3.4 Résumé des classes

Le présent paragraphe contient un résumé des classes définies. Le Tableau 75 présente un résumé des classes.

Tableau 75 – Résumé des classes

ASE	Classe
Virtual field device (Dispositif de terrain virtuel)	VFD
Data type (Type de données)	Fixed Length Data type (type de données de longueur fixe)
	String Data type (type de données chaîne)
Variable	Simple Variable
	Structure variable

7.3.5 Service permis par le rôle AREP

Le Tableau 76 et le Tableau 77 définissent les combinaisons valides de services et de classe AREP (lequel protocole de service d'application de données (APDU) et IAREP avec la classe appropriée peuvent envoyer ou recevoir).

Tableau 76 – Services confirmés par la classe AREP

Services	Master		Slave	
	req	cnf	Ind	rsp
VFD ASE				
Identify	X	X	X	X
Variable ASE				
Read service – all (tous)	X	X	X	X
Write service – all (tous)	X	X	X	X

Tableau 77 – Services non confirmés de la classe AREP

Services	Slave	Master
	req	ind
Variable ASE		
Information report	X	X

8 Spécification de protocole de couche application – Éléments de Type 20

8.1 Syntaxe abstraite

La syntaxe abstraite d'une Unité de donnée de protocole application (APDU) est combinée à leur syntaxe de transfert et est spécifiée en 8.2.

8.2 Syntaxe de transfert

8.2.1 Champs communs aux APDU

8.2.1.1 Champ "Command number"

Le Command number (numéro de commande) est le Numeric Identifier du service Read (7.3.2.2.3.3), Write (7.3.2.2.3.4), Information report (7.3.2.2.3.5) ou Action (7.3.2.3.4). Il identifie le type de service et l'objet dans le « Slave » qui n'a pas été accessible ou l'action qui est à accomplir. Il est le Unsigned16 extended Command number (numéro de commande étendu de type Unsigned16). Les commandes 0 à 255 sont remplies de zéros pour former le numéro de commande de 16 bits (par exemple, la Command 9 est émise comme étant 0x0009). La commande 31 est réservée et ne doit pas être utilisée dans la demande.

8.2.1.2 Champ "Octet count"

Ce champ représente le nombre d'octets dans la totalité des champs qui suivent ce champ. Le type de données de ce champ est Unsigned8 et sa valeur peut être de 0 à 255. Si la valeur de ce champ est zéro, il s'agit alors du dernier champ dans l'APDU.

8.2.1.3 Champ "Response code"

Le champ "Response code" (code de réponse) est du type de données Enumeration et il indique soit l'achèvement réussi de la commande demandée, soit l'achèvement de la commande demandée avec un avertissement ou une erreur lors de l'exécution de la commande demandée, comme montré au Tableau 78. Certaines des valeurs dépendent de la valeur du champ "Command". Chaque structure d'APDU spécifique au service montre les valeurs de code de réponse attribuées au service en question. Un dispositif est requis pour utiliser ces valeurs. Le dispositif peut répondre avec une valeur non utilisée tant qu'elle indique un état qui n'est pas défini dans la présente norme.

Tableau 78 – Valeurs du code de réponse

Valeur	Classe	Définition
0	Success	La commande (Read ou Write) a été exécutée correctement.
1 – 127	Warning	La commande (Write) a été exécutée avec l'écart décrit dans la réponse (par exemple, une valeur a été mise à la valeur légale la plus proche.)
1 – 127	Error	L'exécution de la commande n'a pas été accomplie correctement et le "Response code" indique la raison (par exemple, le dispositif est dans le mode Write Protect (protection contre l'écriture)).
1 – 127	Reserved	Un code peut être réservé pour usage futur par le protocole de la présente norme. Un dispositif n'est pas autorisé à retourner une valeur réservée.

Les valeurs de codes de réponse qui sont communes à toutes les commandes et le comportement du dispositif répondeur pour ces codes en question sont spécifiées à l'Annexe E.

8.2.1.4 Application process status

Ce champ est fourni par l'utilisateur de la FAL. Son type de données est un Bit Field d'un octet de longueur. Il n'est associé à l'achèvement d'aucune commande. Ce champ doit avoir au moins le statut suivant:

- a) Configuration changed (configuration modifiée) – une opération exécutée a changé la configuration du dispositif et
- b) More status available (plus de statuts disponibles) – des informations supplémentaires sur le statut sont disponibles via une autre commande Read.

Si "Configuration changed" est à '1', il convient que le client lise d'autres variables dans le serveur, s'il y a lieu et utilise ensuite le service "Reset configuration changed flag" pour réinitialiser ce bit de statut.

Si "More status available" est à '1', il convient que le client utilise le service "Read additional device status" pour récupérer les informations qui ne sont pas incluses dans le champ "Application process status".

8.2.1.5 Extended status

Ce champ indique le statut du VFD. Les valeurs possibles de ce paramètre sont montrées au Tableau 79. Son type de données est un Bit Field d'un octet de longueur. Il n'est associé à l'achèvement d'aucune commande.

Tableau 79 – Valeurs d'Extended status

Code	Valeur	Description
0x01	Maintenance required (maintenance requise)	Le VFD exige une maintenance, même s'il n'a pas mal fonctionné.
0x02	Device variable alert (alerte variable de dispositif)	Une ou plusieurs des "Device variables" sont dans un état Alarm (alarme) ou Warning (avertissement).
0x04	Critical power failure (panne de courant critique)	La puissance disponible atteint un niveau faible critique; utilisée par les dispositifs qui fonctionnent à partir de l'énergie stockée.

8.2.1.6 Champ "Value"

Le champ "Value" (valeur) représente les données utilisateur qui sont transférées entre la couche application et son utilisateur. L'ALE l'assemble à partir de paramètres d'une primitive de service ou l'analyse en paramètres d'une primitive de service. Sa structure dépend du type de l'APDU.

8.2.1.7 Formats des structures d'APDU

La couche application expéditrice prépare une APDU afin de la transférer à la couche application réceptrice. À cet effet, elle utilise les paramètres issus des primitives de service. Il y a plusieurs formats de l'APDU:

- APDU de demande du dispositif Master au dispositif Slave,
- réponse normale du dispositif Slave au dispositif Master, et
- réponse d'erreur de commande du dispositif Slave au dispositif Master.

Les formats communs pour les structures d'APDU sont spécifiés en 8.2.2. Les formats pour les structures d'APDU spécifiques aux services sont spécifiés en 8.2.3, 8.2.4, 8.2.5 et 8.2.6.

Le champ "Command number" est inclus tant dans la demande que dans la réponse. Le demandeur n'est pas obligé de stocker l'APDU de demande pour l'associer à l'APDU de réponse.

8.2.2 Structure commune aux APDU

8.2.2.1 APDU de demande du dispositif maître au dispositif esclave

Toutes les APDU ont une structure commune telle que montrée à la Figure 41.

Command number	Octet count	Value

Légende

Anglais	Français
Command number	Command number (Numéro de commande)
Octet count	Octet count (Compte d'octets)
Value	Value (Valeur)

Figure 41 – Format d'APDU de demande du maître à l'esclave

Pour un service Read, le champ "Value" est soit Null, soit un Variable Sub Index et pour un type de variable "array", il peut inclure le nombre d'éléments à lire. Pour un service Write, le champ "Value" est un Variable Sub Index, suivi le cas échéant de la valeur qui a été inscrite dans les objets Slave identifiés par la commande. L'ALE l'assemble à partir des paramètres d'une primitive de service. Sa structure dépend du type de l'APDU spécifique au service.

8.2.2.2 APDU de réponse normale de l'esclave au dispositif maître

La Figure 42 montre le format des champs de couche application qui sont utilisés pour une réponse sans erreurs de l'esclave au maître.

Application process Status	Extended Status	Command number	Octet count	Response code	Value

Légende

Anglais	Français
Application process Status	Application process Status (Statut de processus d'application)
Extended Status	Extended Status (Statut étendu)
Command number	Command number (Numéro de commande)
Octet count	Octet count (Compte d'octets)
Response code	Response code (Code de réponse)
Value	Value (Valeur)

Figure 42 – Format d'APDU de réponse normale de l'esclave au maître

L'Esclave envoie cette APDU en réponse à une demande du Maître. La valeur du champ "Command number" dans cette APDU est identique à celle du champ "Command number" dans l'APDU de demande correspondante.

Le champ "Octet count" est le nombre octets dans le champ "Value" plus un. La valeur minimale du champ "Octet count" est de un (1).

Si la réponse est pour la demande Read, le champ "Value" est utilisé pour la valeur des objets qui sont lus du dispositif esclave. Si la réponse est pour la demande Write, ce champ est utilisé pour la valeur des objets qui ont été inscrits dans le dispositif esclave. Dans la réponse à la demande Write, il s'agit de la même valeur reçue dans la demande, à moins que

la valeur reçue ait été interdite et le serveur ait écrit une valeur différente. Dans ce cas, il s'agit de la valeur écrite dans le dispositif esclave.

Le Maître doit émettre une seule commande à un seul esclave jusqu'à ce que l'Esclave en question envoie la réponse ou que la demande dépasse son délai imparti et devienne non valide (voir 7.3.2.2.3.2 et 7.3.2.3.2).

8.2.2.3 APDU d'erreur de commande d'un esclave vers le dispositif maître

La Figure 43 montre le format des champs de la couche application qui sont utilisés pour une réponse de l'Esclave au Maître, lorsqu'il y a une erreur lors de l'exécution de la commande. Le champ "Value" n'existe pas.

Application process Status	Extended Status	Command number	Octet count	Response code

Légende

Anglais	Français
Application process Status	Application process Status (Statut de processus d'application)
Extended Status	Extended Status (Statut étendu)
Command number	Command number (Numéro de commande)
Octet count	Octet count (Compte d'octets)
Response code	Response code (Code de réponse)

Figure 43 – Format d'APDU de réponse en cas d'erreur de commande de l'esclave au maître

Le dispositif répondeur envoie cette APDU en réponse à une demande du Maître. La valeur du champ "Command number" dans cette APDU est identique à celle du champ "Command number" dans l'APDU de demande correspondante.

Le champ "Octet count" est mis à un (1).

La commande 31 est réservée et toute réponse à une demande de commande 31 doit être un code d'erreur "Command Not Implemented" (commande non mise en œuvre).

8.2.2.4 APDU de rapport d'informations d'un esclave vers le dispositif maître

L'édition de données et de rapport d'informations utilise l'APDU de réponse définie en 8.2.2.2. L'édition est établie pour utiliser l'un des numéros de commande (voir 8.2.3.32). D'autres numéros de commande de rapport d'informations sont spécifiés dans les paragraphes relatifs aux APDU spécifiques aux services.

8.2.2.5 Agrégat de commandes

La présente norme permet de transporter plusieurs commandes et données associées dans une seule APDU de demande et de réponse. L'agrégat de commandes est particulièrement utile pour lire les configurations des dispositifs. L'APDU de commande agrégée est montrée à la Figure 44. Le champ "Command #n Request" (Demande de Commande numéro *n*) est le même que l'APDU de demande montrée à la Figure 41. Le champ "Command #n Response" (c'est-à-dire Réponses de Commande numéro *n*) est le même que dans l'APDU de réponse montrée à la Figure 42 et à la Figure 43, excepté que les champs "Application process status" et "Extended status" sont utilisés une seule fois au début et ne font pas partie des champs "Command #1 Response" à "Command #n Response".

Command #1 Request		Command #n Request

a) request APDU

Application process Status	Extended Status	Command #1 Response		Command #n Response

b) response APDU

Légende

Anglais	Français
Command #1 Request	Demande de commande n°1
Command #n Request	Demande de commande n°n
a) request APDU	a) APDU de demande
Application process Status	Application process Status (Statut de processus d'application)
Extended Status	Extended Status (Statut étendu)
Command #1 Response	Réponse de commande n°1
Command #n Response	Réponse de commande n°n
b) response APDU	b) APDU de réponse

Figure 44 – APDU de commande agrégée

Si le dispositif a été configuré pour éditer plus d'un message (voir 8.2.3.32), il peut les combiner en un seul message utilisant l'agrégation de commandes, excepté que "Publish data Message 0" ne doit jamais être agrégé.

Chaque fois que possible, le dispositif de terrain conforme de Type 20 doit agréger plusieurs "Publish data Message" en attente en une seule transaction utilisant la demande de rapport d'informations (Information report). Si les "Publish data Message" ne s'insèrent pas dans une seule transaction, ils peuvent être envoyés dans des transactions distinctes.

Les exigences générales pour agréger des commandes sont comme suit.

- Si un dispositif reçoit une APDU de demande agrégée, il doit fournir une réponse à tous les champs "Command" de la demande dans une seule APDU de réponse agrégée.
- Une commande avec un numéro dans la plage de 0x0300 à 0x03FF doit être agrégée uniquement avec une autre commande dans cette même plage.
- Une commande de lecture peut être agrégée avec une commande d'écriture.

NOTE 1 Le mélange de commandes de lecture et d'écriture dans une même agrégation est permis, mais n'est pas recommandé.

- L'APDU de configuration de gestion de réseau (voir 8.2.6) ne doit être agrégée à aucun autre type. La demande de configuration de gestion de réseau est une transaction acquittée contenant une commande d'écriture qui modifie le comportement du réseau.

NOTE 2 Les commandes ne dépendent pas les unes des autres. Cela implique que les commandes peuvent être agrégées. Par exemple, plusieurs commandes de lecture peuvent être agrégées en une seule transaction. Certaines restrictions peuvent être imposées dans les exigences relatives à des commandes spécifiques. Le protocole n'impose aucune autre restriction à la restriction relative aux commandes. Une application d'hôte (utilisateur de la FAL), par exemple un étalonnage de dispositif, peut empêcher l'agrégation de commandes.

8.2.3 Structures des APDU spécifiques aux services application de dispositifs

8.2.3.1 Identify FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.2.

8.2.3.2 Read primary variable FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.3.

8.2.3.3 Read loop current and percent of range FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.4.

8.2.3.4 Read dynamic variables and loop current FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.5.

8.2.3.5 Write loop configuration FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.6.

8.2.3.6 Read loop configuration FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.7.

8.2.3.7 Read dynamic variable families classifications FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.8.

8.2.3.8 Read device variables with status FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.9.

8.2.3.9 Read message FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.10.

8.2.3.10 Read tag, descriptor and date FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.11.

8.2.3.11 Read primary variable transducer FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.12.

8.2.3.12 Read device information FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.13.

8.2.3.13 Read final assembly number FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.14.

8.2.3.14 Write message FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.15.

8.2.3.15 Write tag, descriptor and date FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.16.

8.2.3.16 Write final assembly number FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.17.

8.2.3.17 Read long_tag FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.18.

8.2.3.18 Write long_tag FAL PDU

Cette commande est spécifiée dans la CEI 61158-6-20:2010, 5.3.19.

8.2.3.19 Reset configuration changed flags FAL PDU**8.2.3.19.1 Primitive Request**

La valeur de la Commande est trente-huit (38); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est tel que montré au Tableau 80.

Tableau 80 – Champ "Value" de Reset configuration changed flags

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Configuration change counter	Nombre total de changement de configuration

8.2.3.19.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 80. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 81.

Tableau 81 – Codes de réponse spécifiques à la commande "Reset configuration changed flag"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
9	Error	Discordance des compteurs de changements de configuration
16	Error	Accès limité

8.2.3.19.3 Procédure au niveau du dispositif répondeur

Un dispositif de terrain doit contenir séparément des "Configuration changed" pour chaque Master – primaire, le secondaire et la passerelle. Tous les bits "Configuration changed" doivent être mis à '1' lorsqu'un élément de configuration quelconque dans le dispositif de terrain a changé. Cette commande doit remettre le bit "Configuration changed" bit à '0', qui est associé au dispositif Maître émettant cette commande. Le bit en question doit rester à la valeur '0', jusqu'à ce qu'une autre modification soit apportée à la configuration du dispositif. La valeur de ce bit doit être maintenue même en cas de coupure de l'alimentation en énergie du dispositif ou de réinitialisation du dispositif. Le bit "Configuration changed" de l'octet "Application process status" pour le dispositif Maître n'ayant pas émis cette commande est

inaltéré par cette commande et reste mis à '1' jusqu'à ce que l'autre dispositif Maître émette cette commande.

Lorsqu'il reçoit cette commande, le dispositif doit comparer le compteur de changements de configuration ("Configuration change counter") reçu à la valeur courante que ce paramètre a dans le dispositif. Si les valeurs ne concordent pas, le dispositif doit retourner un "Configuration Change Counter Mismatch" (Discordance des compteurs de changements de configuration) et ne pas réinitialiser le bit "Configuration changed".

8.2.3.20 Perform self test FAL PDU

8.2.3.20.1 Primitive Request

La valeur de la Commande est quarante et un (41); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.3.20.2 Primitive Response

Le champ "Value" de la réponse est null (vide) Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 82.

Tableau 82 – Codes de réponse spécifiques à la commande "Perform self test"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
16	Error	Accès limité
32	Error	"Busy" (Occupé)

8.2.3.20.3 Procédure au niveau du dispositif répondeur

Cette commande déclenche la fonction d'autotest dans le dispositif. L'exécution de cette commande peut prendre un temps relativement long pour s'achever. Au cours de l'autotest, le dispositif de terrain doit continuer à communiquer. Cependant, il lui est permis de répondre 'Busy' ('Occupé') à toutes les commandes reçues au cours de cette durée.

Le courant de boucle (Loop Current) peut ne pas refléter le processus alors que l'autotest s'exécute. Un Maître ne doit ni générer de messages d'erreur parasites ni se déconnecter d'un dispositif alors que l'autotest est en cours dans le dispositif en question.

8.2.3.21 Perform device reset FAL PDU

8.2.3.21.1 Primitive Request

La valeur de la Commande est quarante-deux (42); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.3.21.2 Primitive Response

Le champ "Value" de la réponse est null (vide) Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 83.

Tableau 83 – Codes de réponse spécifiques à la commande "Perform device reset"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
16	Error	Accès limité
32	Error	Occupé

8.2.3.21.3 Procédure au niveau du dispositif répondeur

Le dispositif doit répondre à cette commande immédiatement et dans la limite du délai d'expiration de l'esclave et puis se réinitialiser lui-même. Cela équivaut à effectuer un cycle de coupure et de remise de courant sur le dispositif.

L'exécution de cette commande peut prendre une durée relativement longue pour s'achever. Le dispositif ne peut pas répondre à des commandes consécutives tant que la réinitialisation n'est pas achevée. Le dispositif ne peut pas répondre tant qu'il est dans le processus de réinitialisation. Cela peut ressembler à une perte de communications avec le dispositif. Un Maître ne doit ni générer de messages d'erreur parasites ni se déconnecter du dispositif alors que la réinitialisation est en cours dans le dispositif en question.

8.2.3.22 Read additional device status FAL PDU**8.2.3.22.1 Primitive Request**

La valeur de la Commande est quarante-huit (48); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.3.22.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 84. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 85.

Tableau 84 – Champ "Value" de "Read additional device status"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Bit Field (total de 6)	Device specific status 0	Statut spécifique du type de dispositif et de fabricant
6	Bit Field	Extended status	Statut opérationnel étendu du dispositif VFD
7	Bit Field	Device operating mode	Mode opérationnel du dispositif
8	Bit Field	Standardized status 0	Statut du dispositif tel que spécifié au Tableau F.8
9	Bit Field	Standardized status 1	Statut du dispositif tel que spécifié au Tableau F.9
10	Bit Field	Analog channel saturated	Statut de la voie analogique comme spécifié au Tableau F.6
11	Bit Field	Standardized status 2	Statut du dispositif tel que spécifié au Tableau F.10
12	Bit Field	Standardized status 3	Statut du dispositif tel que spécifié au Tableau F.11
13	Bit Field	Analog channel fixed	Statut de la voie analogique comme spécifié au Tableau F.7
14	Bit Field (1 à 11)	Device specific status 1	Statut spécifique du type de dispositif et de fabricant

NOTE Le Device specific status 0 et le Device specific status 1 peuvent être des types de données soit Bit Field, soit Enumeration.

**Tableau 85 – Codes de réponse spécifiques à la commande
"Read additional device status"**

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Mis à jour en cours
16	Error	Accès limité

8.2.3.22.3 Procédure au niveau du dispositif répondeur

Si le dispositif est capable de répondre à la demande, mais les informations sur le statut sont en attente de l'achèvement d'une commande qui requiert une durée relativement longue pour s'achever, la valeur de code de réponse "Update in progress" (Mise à jour en cours) doit être retournée. Il convient que le Client répète la demande après un délai.

Sinon, le dispositif doit répondre en indiquant 'Success' et inclure tous les champs Value pris en charge par la mise en œuvre par le dispositif de la Commande 48 dans la réponse.

Les champs Response Value retournés sont tronqués après le dernier octet de statut pris en charge par le dispositif. Tous les dispositifs doivent prendre en charge au moins les octets numéros 0 à 8, y compris "Extended status", "Device operating mode" et "Standardized status 0". Si le dispositif prend en charge plus d'une voie analogique, les octets numéros 9 à 13 doivent également être renvoyés.

8.2.3.23 Reset more status available FAL PDU

8.2.3.23.1 Primitive Request

La valeur de la Commande est quarante-huit (48); la valeur du champ "Octet count" est de neuf (9) à vingt-cinq (25) selon la mise en œuvre du dispositif. Le champ "Value" est tel que montré au Tableau 86.

Tableau 86 – Champ "Value" de "Read additional device status"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Bit Field (total de 6)	Device specific status 0	Statut spécifique du type de dispositif et de fabricant
6	Bit Field	Extended status	Statut opérationnel étendu du dispositif VFD
7	Bit Field	Device operating mode	Mode opérationnel du dispositif
8	Bit Field	Standardized status 0	Statut du dispositif tel que spécifié au Tableau F.8
9	Bit Field	Standardized status 1	Statut du dispositif tel que spécifié au Tableau F.9
10	Bit Field	Analog channel saturated	Statut de la voie analogique comme spécifié au Tableau F.6
11	Bit Field	Standardized status 2	Statut du dispositif tel que spécifié au Tableau F.10
12	Bit Field	Standardized status 3	Statut du dispositif tel que spécifié au Tableau F.11
13	Bit Field	Analog channel fixed	Statut de la voie analogique comme spécifié au Tableau F.7
14	Bit Field (1 à 11)	Device specific status 1	Statut spécifique du type de dispositif et de fabricant

NOTE Le Device specific status 0 et le Device specific status 1 peuvent être des types de données soit Bit Field, soit Enumeration.

8.2.3.23.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 86. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 85.

**Tableau87 – Codes de réponse spécifiques à la commande
"Read additional device status"**

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Mis à jour en cours
16	Error	Accès limité

8.2.3.23.3 Procédure au niveau du dispositif répondeur

Si le dispositif est capable de répondre à la demande, mais les informations sur le statut sont en attente de l'achèvement d'une commande qui requiert une durée relativement longue pour s'achever, la valeur de code de réponse "Update in progress" (Mise à jour en cours) doit être retournée. Il convient que le Client répète la demande après un délai.

Sinon, il doit comparer les valeurs reçues aux valeurs courantes contenues dans le dispositif. S'il y a une concordance exacte, le bit "More status available" correspondant au dispositif demandeur (maître primaire, maître secondaire ou passerelle) doit être remis à '0'. S'il n'y a pas une concordance exacte ou si peu d'octets sont reçus, le bit "More status available" doit rester inchangé. Si la demande contient plus de champs "Value" que ceux pris par le dispositif, ces champs supplémentaires doivent être ignorés. Le dispositif doit toujours envoyer les valeurs courantes des champs Value dans la réponse.

Les champs Response Value retournés sont tronqués après le dernier octet de statut pris en charge par le dispositif. Tous les dispositifs doivent prendre en charge au moins les octets numéros 0 à 8, y compris "Extended status", "Device operating mode" et "Standardized status 0". Si le dispositif prend en charge plus d'une voie analogique, les octets numéros 9 à 13 doivent également être renvoyés.

8.2.3.24 Read device variable information FAL PDU

8.2.3.24.1 Primitive Request

La valeur de la Commande est quarante-quatre (54); la valeur du champ "Octet count" est de un (1). Le champ "Value" est tel que montré au Tableau 88.

Tableau 88 – Champ "Value" de "Read device variable information"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Variable code	Variable pour laquelle des informations sont demandées

8.2.3.24.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 89. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 90.

Tableau 89 – Champ "Value" de réponse "Read device variable information"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Variable code	Code reçu dans la demande dans le champ "Value"
1	Unsigned24	Transducer serial number	Value de "Transducer Serial Number" (numéro de série du transducteur)
4	Enumeration	Transducer unit	Code pour les unités des limites de transducteur (Transducer Limits) et de l'étendue minimale (Minimum Span)
5	Float32	Upper transducer limit	Value de la variable "Upper Transducer Limit" (Limite supérieure de transducteur)
9	Float32	Lower transducer limit	Valeur de la variable "Lower Transducer Limit" (Limite inférieure de transducteur)
13	Float32	Damping value	Constante de registre pour la valeur du transducteur
17	Float32	Minimum span	Valeur de la variable "Minimum Span" (étendue minimale)
21	Enumeration	Device variable classification	Fonction accomplie par la variable Device comme spécifié au Tableau F.5
22	Enumeration	Device variable family	La famille de dispositifs telle que spécifiée au Tableau F.4
23	Time	Update period	La période maximale entre mises à jour de la variable Device

Tableau 90 – Codes de réponse spécifiques à la commande "Read device variable information"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
16	Error	Accès limité
32	Error	Occupé

8.2.3.24.3 Procédure au niveau du dispositif répondeur

S'il n'y a pas de transducteur pour la variable Device sélectionnée, le dispositif doit mettre le numéro de série du transducteur à zéro, mettre les autres paramètres Float32 à "0x7F, 0xA0, 0x00, 0x00" et les paramètres d'unités de transducteur à "250" (non utilisé) lorsqu'ils ne s'appliquent pas.

Si la classification de variable Device n'est pas prise en charge par la variable Device, le dispositif doit retourner "0" (non mis en œuvre). Si la famille de variables Device n'est pas prise en charge par la variable Device, le dispositif doit retourner "250" (non utilisé).

Le dispositif doit mettre à jour la variable Device au moins une fois dans l'intervalle indiqué par le champ "Update period" (période de mise à jour).

8.2.3.25 Write device variable FAL PDU**8.2.3.25.1 Primitive Request**

La valeur de la Commande est soixante-dix-neuf (79); la valeur du champ "Octet count" est de huit (8) et le champ "Value" est tel que montré au Tableau 91.

Tableau 91 – "Champ "Value" de "Write device variable"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Variable code	est la Variable pour laquelle des valeurs doivent être écrites
1	Enumeration	Device variable command	est la commande devant être exécutée telle que spécifiée au Tableau F.3
2	Engineering unit	Device variable unit	Unité de mesure de la variable
3	Float32	Device variable value	Valeur devant être écrite pour la variable
7	Bit Field	Device variable status	Statut devant être écrit pour la variable

8.2.3.25.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 91. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 92.

Tableau 92 – Codes de réponse spécifiques à la commande "Write device variable"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Bit de statut de "Device family" non mis
10	Error	Code non valide pour la variable Device
16	Error	Accès limité
17	Error	Indice non valide pour la variable Device; La variable Device n'existe pas dans le dispositif
18	Error	Code d'unités non valide
19	Error	Indice de variable Device non autorisé pour cette commande
32	Error	"Busy" (Occupé) – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.3.25.3 Procédure au niveau du dispositif répondeur

La commande "Write Device variable" contrôle le fonctionnement du dispositif soit en forçant la valeur d'étude, soit en autorisant le fonctionnement normal. Pour les variables Device calculées automatiquement par le dispositif de terrain, le fonctionnement normal doit reprendre lorsque l'énergie est retirée du dispositif et de nouveau appliquée à celui-ci ou suite à l'achèvement d'une réinitialisation du dispositif.

Un dispositif de terrain doit lancer une DR (réponse différée) si la variation de valeur de la variable Device dépasse la limite du taux de variation. Cette DR doit continuer jusqu'à ce qu'une nouvelle valeur puisse être obtenue. Pendant que la variable Device reste forcée, le code de réponse 8 – "Update failure" (échec de mise à jour) doit être retourné chaque fois que la valeur forcée est lue par un client. Cela inique au maître de détecter cet état. Le fait de

forcer la valeur d'une variable Device peut altérer les bits dans l'octet "Application process status".

Il convient que l'application d'hôte n'utilise pas les bits de statut "Device family" (famille de dispositifs) pour des variables Device ne prenant pas en charge une famille de dispositifs (c'est-à-dire que le dispositif de terrain réinitialise toujours ces bits dans ce cas).

Le code d'unité de variable Device reçu avec la commande n'altère pas les unités de variable Device du dispositif de terrain. La valeur de variable Device doit être retournée dans les mêmes unités que celles reçues.

8.2.3.26 Read real-time clock FAL PDU

8.2.3.26.1 Primitive Request

La valeur de la Commande est quatre-vingt-dix (90); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.3.26.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 93. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 94.

Tableau 93 – Champ "Value" de "Read real-time clock"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Date	Date	Date courante
3	Time	Current time	Heure du jour courante
7	Date	Date last set	Dernier réglage de date
10	Time	Time last set	Dernier réglage de l'heure
14	Bit Field	RTC flags	Flags RTC (Real-Time Clock, horloges temps réel) tels que spécifiés au Tableau F.18

Tableau 94 – Codes de réponse spécifiques à la commande "Read real-time clock"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif

8.2.3.26.3 Procédure au niveau du dispositif répondeur

Le dispositif doit répondre avec l'heure interne à laquelle la demande a été reçue. Si l'horloge n'a pas été réglée, la dernière heure réglée doit être initialisée au 01 janvier 1900 minuit (00:00).

8.2.3.27 Write publish data period FAL PDU

8.2.3.27.1 Primitive Request

La valeur de la Commande est de cent trois (103); la valeur du champ "Octet count" est de neuf (9) et le champ "Value" est tel que montré au Tableau 95.

Tableau 95 – Champ "Value" de "Write publish data period"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Publish data message ref	Référence à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Time	Update period	Durée entre deux publications successives des données spécifiées, si les conditions de déclenchement dans la commande 104 sont remplies
5	Time	Maximum update period	Durée entre deux publications successives des données spécifiées, si les conditions de déclenchement dans la commande 104 ne sont pas remplies
NOTE La valeur retournée dans les octets de données de réponse reflète la valeur effectivement utilisée par le dispositif.			

8.2.3.27.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 95. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 96.

Tableau 96 – Codes de réponse spécifiques à la commande "Write publish data period"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données recus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Temps de mise à jour ajustés
9	Error	Code non valide pour la variable Device
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.3.27.3 Procédure au niveau du dispositif répondeur

Cette demande sélectionne la période de mise à jour minimale et maximale d'un message Publish data. Le temps minimal doit être inférieur ou égal au temps maximal. Seules sont permises les valeurs suivantes du paramètre "Update period":

0,100 s,
0,250 s,
0,500 s,
1 s,
2 s,
4 s,
8 s,
16 s,
32 s, et

n'importe quelle valeur de 60 s à 3 600 s.

Les dispositifs de terrain doivent corriger à la valeur permise la plus proche les valeurs de réglage qui diffèrent de celles-ci et indiquer "Update times adjusted" (Temps de mise à jour ajustés). La valeur par défaut du paramètre "Update period" est de 60 s.

Si plusieurs messages Publish data sont activés et coïncident tous dans la limite du même temps, ils sont agrégés à l'aide de la Commande 78 pour former une seule APDU. Les messages Publish data ayant la plus haute priorité seront concaténés les premiers jusqu'au nombre maximal d'octets qu'autorise la couche de transport respective. Le message Publish data ayant le numéro de référence le plus bas a une plus haute priorité que le message Publish data ayant le numéro de référence le plus élevé. L'événement "Application process status" a une plus faible priorité que les messages Publish data et a une plus haute priorité que l'événement de Commande 48.

8.2.3.28 Write publish data trigger FAL PDU

8.2.3.28.1 Primitive Request

La valeur de la Commande est de cent quatre (104); la valeur du champ "Octet count" est de huit (8) et le champ "Value" est tel que montré au Tableau 97.

Tableau 97 – Champ "Write publish data trigger"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Publish data message ref	Référence à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Enumeration	Trigger mode	Code de sélection du mode Trigger (déclenchement) tel que spécifié au Tableau F.12
2	Enumeration	Device variable classification	Fonction accomplie par la variable Device comme spécifié au Tableau F.5
3	Engineering unit	Device variable unit	Unité de mesure de la variable
4	Float32	Trigger level	Fenêtre ou niveau utilisé(e) pour déclencher l'édition
NOTE La valeur retournée dans les octets de données de réponse reflète la valeur effectivement utilisée par le dispositif.			

8.2.3.28.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 97. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 98.

Tableau 98 – Codes de réponse spécifiques à la commande "Write publish data trigger"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
3	Error	La valeur du paramètre devant être écrite est plus élevée que celle permise
4	Error	La valeur du paramètre devant être écrite est plus faible que celle permise
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture

Valeur	Classe	Description
9	Error	"Publish data message ref" non valide
11	Error	Classification non valide pour la variable Device
12	Error	Code d'unités non valide
13	Error	Code de sélection non valide pour le mode déclenchement de Publish data
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.3.28.3 Procédure au niveau du dispositif répondeur

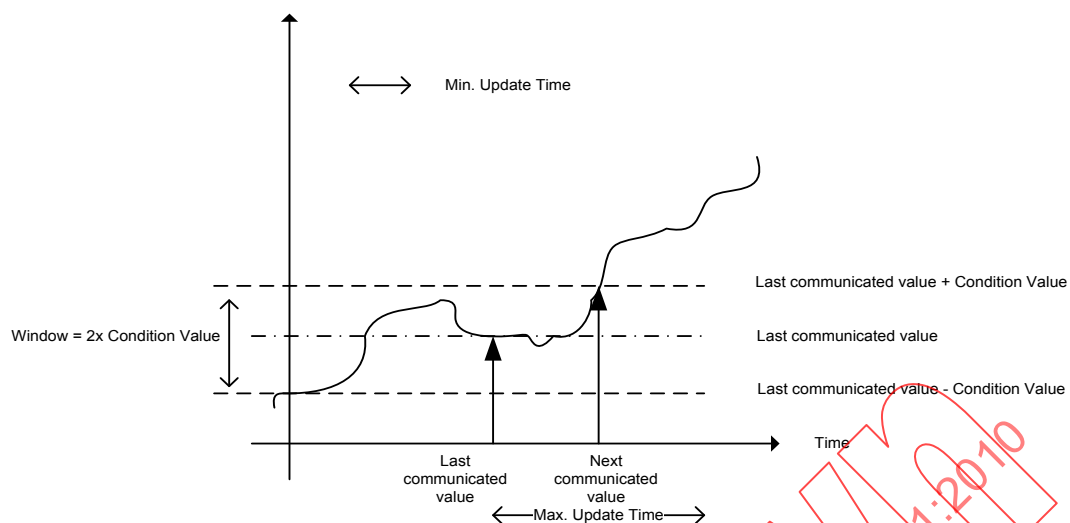
Sauf configuration contraire par cette demande, le message Publish data doit supposer que le mode Trigger est "Continuous" et que la valeur Trigger est de 0,00. De surcroît, si une commande spécifique au dispositif est spécifiée comme la commande Publish, cette demande doit retourner "Access restricted" (accès limité) et doit utiliser la valeur par défaut de "Continuous" pour le mode Trigger.

Les modes de déclenchement permettent de configurer le dispositif pour différer l'édition du message Publish data au-delà de la période de mise à jour spécifiée dans la Commande 103 (voir Tableau 95). Dans tous les cas, le message Publish data doit être déclenché en cas de dépassement du temps maximal de mise à jour indiqué dans la Commande 103 (voir Tableau 95). La source de déclenchement est spécifiée et est normalement la première valeur de processus retournée comme réponse à la commande comme montré au Tableau 99. Le numéro de commande pour le message Publish data est spécifié en 8.2.3.32.

Tableau 99 – Source de déclenchement de message Publish data

Command	Référence	Source de déclenchement
1	8.2.3.2	Valeur de Primary Variable
2	8.2.3.3	Primary Variable en pourcent de la plage
3	8.2.3.4	Primary Variable
9	8.2.3.8	Variable Device pour intervalle de temps 0

Dans le mode "Windowed" (fenêtrage), la valeur de Trigger doit être un nombre positif et est la fenêtre symétrique autour de la dernière valeur communiquée. La Figure 45 illustre l'édition du message Publish data après dépassement de la fenêtre.

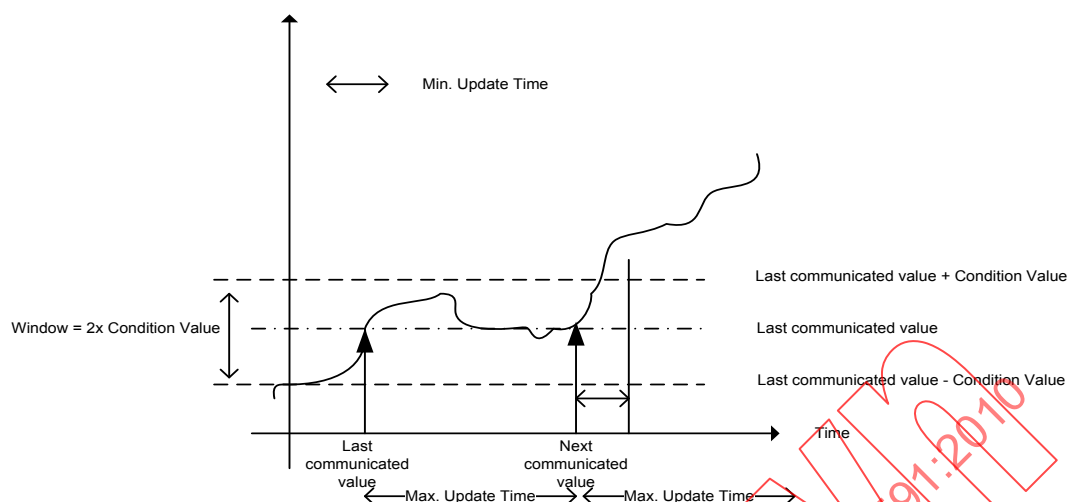


Légende

Anglais	Français
Min .Update Time	Temps de mise à jour minimum
Window=2X Condition Value	Fenêtre=2 fois la valeur de condition
Last communicated value	Dernière valeur communiquée
Next communicated value	Prochaine valeur communiquée
Max update time	Temps de mise à jour maximum
Last communicated value+condition value	Dernière valeur communiquée +valeur de condition

Figure 45 – Mode de déclenchement à fenêtre

La Figure 46 montre que même si la valeur ne dévie pas au-delà de la fenêtre, un "Publish Data Message" est toujours édité en cas de dépassement de la période "Max Update Period".

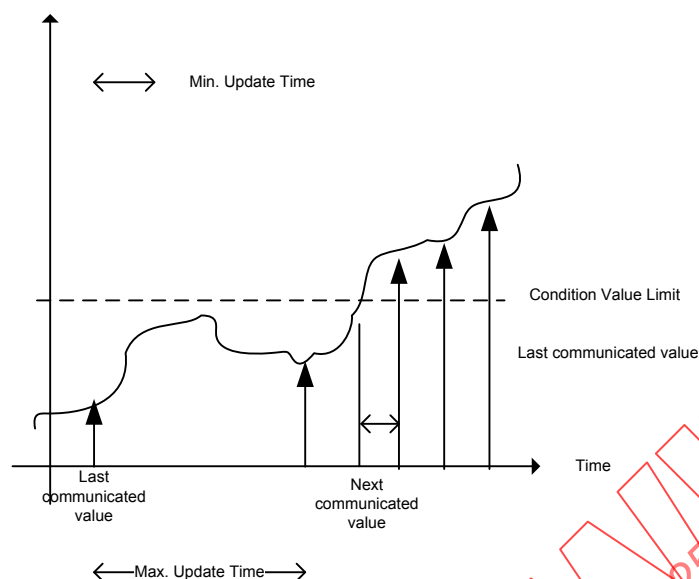


Légende

Anglais	Français
Min .Update Time	Temps de mise à jour minimum
Window=2X Condition Value	Fenêtre=2 fois (la valeur de condition)
Last communicated value	Dernière valeur communiquée
Next communicated value	Prochaine valeur communiquée
Max update time	Temps de mise à jour maximum
Last comunicated value+condition value	Dernière valeur communiquée +valeur de la condition
Last comunicated value-condition value	Dernière valeur communiquée -valeur de la condition

Figure 46 – Mode de déclenchement à fenêtre avec expiration du temps maximal de mise à jour

Dans le mode "Rising" (montée), le Publish Data Message doit être édité lorsque la valeur source se situe au-dessus du seuil établi par la valeur de déclenchement. La Figure 47 montre que le temps de mise à jour varie une fois la limite dépassée. Les messages Publish Data sont édités à la fréquence indiquée par la période mise à jour (Update period) tant que la valeur source reste au-dessus de la valeur de déclenchement. Si la valeur passe en dessous de la valeur de déclenchement, le temps de mise à jour doit être la période maximale de mise à jour.



Légende

Anglais	Français
Min. Update Time	Temps de mise à jour minimum
Condition Value Limit	Limite de la valeur de condition
Last communicated value	Dernière valeur communiquée
Next communicated value	Prochaine valeur communiquée
Time	Temps
Max update time	Temps de mise à jour maximum

Figure 47 – Mode de déclenchement à niveau

Dans le mode "Falling" (descente) qui est similaire au mode "Rising", le message "Publish Data Message" doit être édité lorsque la valeur source passe en dessous du seuil établi par la valeur de déclenchement.

8.2.3.29 Read publish data mode configuration FAL PDU

8.2.3.29.1 Primitive Request

La valeur est de cent cinq (105); La valeur du champ "Octet count "est de zéro (0) ou un (1). Si la demande à un octet, le champ "Value" de la demande est montré au Tableau 100.

Tableau 100 – Champ "Value" de la demande "Read publish data mode configuration"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Publish data message ref	Ceci est utilisé pour se référer à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question.

8.2.3.29.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 101. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 102.

Tableau 101 – Champ "Value" de la réponse "Read publish data mode configuration"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Publish data mode control	Code de commande de mode de données Publish tel que spécifié au Tableau F.2
1	Unsigned8	Command number expansion	31 (0x1F) ou LSB du numéro de commande de données Publish – voir Note
2	Unsigned8	Device variable code 0	Code de variable Device attribué à l'intervalle de temps Slot 0
3	Unsigned8	Device variable code 1	Code de variable Device attribué à l'intervalle de temps Slot 1
4	Unsigned8	Device variable code 2	Code de variable Device attribué à l'intervalle de temps Slot 2
5	Unsigned8	Device variable code 3	Code de variable Device attribué à l'intervalle de temps Slot 3
6	Unsigned8	Device variable code 4	Code de variable Device attribué à l'intervalle de temps Slot 4
7	Unsigned8	Device variable code 5	Code de variable Device attribué à l'intervalle de temps Slot 5
8	Unsigned8	Device variable code 6	Code de variable Device attribué à l'intervalle de temps Slot 6
9	Unsigned8	Device variable code 7	Code de variable Device attribué à l'intervalle de temps Slot 7
10	Unsigned8	Publish data message ref	Référence à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
11	Unsigned8	Max publish count	Nombre maximal de messages Publish data pris en charge par le dispositif
12	Unsigned16	Extended command number	Nom de commande étendu (voir 8.2.2.2)
14	Time	Update period	Durée entre deux publications successives des données spécifiées, si les conditions de déclenchement dans la commande 104 sont remplies
18	Time	Maximum update period	Durée entre deux publications successives des données spécifiées, si les conditions de déclenchement dans la commande 104 ne sont pas remplies
22	Enumeration	Trigger mode	Code de sélection du mode Trigger (déclenchement) tel que spécifié au Tableau F.12
23	Enumeration	Device variable classification	Fonction accomplie par la variable Device comme spécifié au Tableau F.5
24	Engineering unit	Device variable unit	Unité de mesure de la variable
25	Float32	Trigger level	Fenêtre ou niveau utilisé(e) pour déclencher l'édition
NOTE Si la demande a un octet, ce paramètre est mis à 31.			

Tableau 102 – Codes de réponse spécifiques à la commande "Read publish data mode configuration"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
9	Error	"Publish data message ref" non valide
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.3.29.3 Procédure au niveau du dispositif répondeur

Si le dispositif reçoit une demande sans octets de données, il doit retourner la configuration de Publish data message '0' et le LSB du numéro de commande Publish doit être retourné dans un octet de donnée de réponse un (1).

Si un intervalle de temps n'est pas configuré pour émettre une variable Device, l'intervalle de temps en question doit retourner "250" (non utilisé) comme code de la variable Device. Si la commande 9 est à éditer, ce code de variable Dispositif pour l'intervalle de temps soit satisfaire aux exigences selon la commande 9 (voir 8.2.3.8).

Si un dispositif met en œuvre ce mode Publish data, il doit prendre en charge au moins trois jeux de Publish data, à savoir la valeur minimale de "Max publish count" doit être de trois (3).

8.2.3.30 Flush delayed responses FAL PDU

8.2.3.30.1 Primitive Request

La valeur de la Commande est de cent six (106); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.3.30.2 Primitive Response

Le champ "Value" de la réponse est null (vide). Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 103.

Tableau 103 – Codes de réponses spécifiques à la commande "Flush delayed responses"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Toutes les réponses purgées sauf les réponses différées en cours
16	Error	Accès limité
32	Error	Occupé une réponse ultérieure ne peut être lancée

8.2.3.30.3 Procédure au niveau du dispositif répondeur

Cette commande efface toutes les réponses différées en attente pour le Maître qui a émis la commande. Les réponses différées actuellement en cours qui ne peuvent pas être interrompues ou abandonnées peuvent être achevées. Si tel est le cas, le Slave (esclave) doit répondre par "All but running delayed responses flushed" (« Toutes les réponses purgées sauf les réponses différées en cours »).

Il convient que les dispositifs prennent en charge un minimum de deux tampons de DR (un pour chaque Master). Si un seul tampon de réponses DR est pris en charge, cette commande doit purger le tampon de DR même s'il est utilisé par l'autre Master qui n'a pas envoyé la demande de purge.

8.2.3.31 Write publish data device variables FAL PDU

8.2.3.31.1 Primitive Request

La valeur est de cent sept (107); la valeur du champ "Octet count " est soit (5), soit (9) et le champ "Value" est tel que montré au Tableau 104.

Tableau 104 – Champ "Value" de "Write publish data mode configuration"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Device variable code 0	Code de variable Device attribué à l'intervalle de temps Slot 0
1	Unsigned8	Device variable code 1	Code de variable Device attribué à l'intervalle de temps Slot 1
2	Unsigned8	Device variable code 2	Code de variable Device attribué à l'intervalle de temps Slot 2
3	Unsigned8	Device variable code 3	Code de variable Device attribué à l'intervalle de temps Slot 3
4	Unsigned8	Device variable code 4	Code de variable Device attribué à l'intervalle de temps Slot 4
5	Unsigned8	Device variable code 5	Code de variable Device attribué à l'intervalle de temps Slot 5
6	Unsigned8	Device variable code 6	Code de variable Device attribué à l'intervalle de temps Slot 6
7	Unsigned8	Device variable code 7	Code de variable Device attribué à l'intervalle de temps Slot 7
8	Unsigned8	Publish data message ref	Référence à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
NOTE La valeur retournée dans les octets de données de réponse reflète la valeur effectivement utilisée par le dispositif.			

8.2.3.31.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 104. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 105.

Tableau 105 – Codes de réponse spécifiques à la commande "Write publish data device variables"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Conflit d'états de Publish data
9	Error	"Publish data message ref" non valide

8.2.3.31.3 Procédure au niveau du dispositif répondeur

Les codes des variables Device reçus dans la demande doivent être utilisés par un dispositif d'édition pour une commande 9 ou 33 dans le mode Publish data. Le maître doit toujours envoyer soit cinq (5), soit neuf (9) octets de données dans la demande. Si l'un quelconque des codes de variables Device débutant par l'intervalle de temps 1 est '250' (non utilisé), la réponse de commande 9 ou commande 33 doit être tronquée à partir du premier intervalle de temps qui a une valeur '250' (non utilisé). Pour la commande 33, seuls les quatre premiers intervalles de temps doivent être utilisés. Si la demande a seulement cinq (5) octets de données, le code de variables Device pour les intervalles de temps 4 à 7 doit être mis à '250' (non utilisé).

Si le mode Trigger (voir Tableau 97) pour le "Publish data message ref" dans cette demande est différent de zéro et la classification de variables Device de la source de déclenchement ne concorde pas avec le nouveau code de variable Device attribué à l'intervalle de temps Slot 0, la valeur provenant de cette demande doit être acceptée et le code de réponse "Publish data condition conflict" (Conflit d'états de Publish data) doit être retourné; le dispositif de terrain doit corriger la classification, les codes d'unité, régler le mode Trigger pour éditer en continu à la période de mise à jour jusqu'à ce qu'il reçoive une autre commande 104.

8.2.3.32 Write publish data mode command number FAL PDU

8.2.3.32.1 Primitive Request

La valeur est de cent huit (108); la valeur du champ "Octet count" est soit trois (3) comme montré au Tableau 106, soit un (1) comme montré au Tableau 107.

Tableau 106 – Champ "Value" de "Tableau"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Publish command number	Numéro de commande 16 bits du message de réponse devant être transmis dans le mode Publish data
2	Unsigned8	Publish data message ref	Référence à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
NOTE La valeur retournée dans les octets de données de réponse reflète la valeur effectivement utilisée par le dispositif.			

Tableau 107 – Champ "Value" de "Write publish data mode command number"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Publish command number	Numéro de commande 8 bits du message de réponse devant être transmis dans le mode Publish data
NOTE La valeur retournée dans les octets de données de réponse reflète la valeur effectivement utilisée par le dispositif.			

8.2.3.32.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 106 et au Tableau 107. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 108.

Tableau 108 – Codes de réponse spécifiques à la commande "Write publish data mode command number"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Conflit d'états de Publish data
9	Error	"Publish data message ref" non valide

8.2.3.32.3 Procédure au niveau du dispositif répondeur

Les valeurs permises pour le numéro de commande Publish sont {1, 2, 3, 9, 33, 48}. Tous les dispositifs doivent prendre en charge toutes ces valeurs sauf 33. Si le dispositif met en œuvre la commande facultative 33, il doit aussi prendre en charge cette valeur de numéro de commande Publish. Si un dispositif de terrain reçoit un seul octet de données dans la demande, il doit supposer pour Publish data message ref une valeur de '0'; et il doit traiter la demande comme un numéro de commande à un seul octet et répondre avec un seul octet dans la réponse.

Si le mode Trigger (voir Tableau 97) pour le "Publish data message ref" dans cette demande est différent de zéro et la classification de variables Device de la source de déclenchement ne concorde pas avec le numéro de commande indiquée dans la demande, la valeur provenant de cette demande doit être acceptée et le code de réponse "Publish data condition conflict" (« Conflit d'états de Publish data ») doit être retourné; le dispositif de terrain doit corriger la classification, les codes d'unité, régler le mode Trigger pour éditer en continu à la période de mise à jour jusqu'à ce qu'il reçoive une autre commande 104.

Après l'activation de l'édition (voir 8.2.3.33), le dispositif doit éditer la réponse à "Publish command number" en utilisant le service "Information report" (rapport d'informations)

8.2.3.33 Write publish data mode control FAL PDU

8.2.3.33.1 Primitive Request

La valeur est de cent neuf (109); la valeur du champ "Octet count" est soit de un (1) – le premier champ, soit de deux (2) – les deux champs, comme montré au Tableau 109.

Tableau 109 – Champ "Value" de "Write publish data mode control"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Publish data mode control	Code de commande de mode de données Publish tel que spécifié au Tableau F.2
1	Unsigned8	Publish data message ref	Référence à un jeu de données Publish dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question

8.2.3.33.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 109. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 110.

Tableau 110 – Codes de réponse spécifiques à la commande "Write publish data mode control"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Période de mise à jour augmentée
9	Error	Largeur de bande insuffisante
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée.

Valeur	Classe	Description
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.3.33.3 Procédure au niveau du dispositif répondeur

Cette commande est utilisée pour activer ou désactiver le mode Publish data pour les dispositifs. Cette commande a une incidence sur le fonctionnement de la couche liaison de données. Toutes les exigences de couche liaison de données relatives à l'entrée et à la sortie du mode Publish Data doivent être satisfaites.

Si la demande concerne l'activation de l'édition, le dispositif répondeur doit exécuter la procédure ci-après.

- Si le dispositif répondeur n'a pas la capacité actuellement programmée pour éditer le jeu de données spécifié pour "Publish data message ref", il doit répondre par "DR initiated" (réponse différée déclenchée).
- Le dispositif doit envoyer une demande "Request timetable FAL PDU" (voir 8.2.4.31.1) au gestionnaire de réseau pour éditer les données. Le gestionnaire de réseau est censé envoyer une réponse à "Request timetable FAL PDU" avec "DR initiated" et commencer à traiter la demande. Alors, le gestionnaire de réseau:
 - alloue les ressources réseau pour le service Publish;
 - si requis, envoie la demande "Write superframe FAL PDU" (voir 8.2.6.6.1) pour établir une supertrame; et
 - envoie la demande "Write link FAL PDU" (voir 8.2.6.8.1) pour établir une liaison (Link) au sein de la supertrame.

NOTE 1 Le gestionnaire de réseau envoie aussi une demande aux autres dispositifs de réseau pour ajouter une supertrame et une liaison (Link) en vue de l'édition.

- Dans la plupart des cas, le gestionnaire de réseau établit des liaisons dans une supertrame existante pour satisfaire à la demande. Une fois le réseau configuré, le gestionnaire de réseau retourne une réponse à "Request timetable FAL PDU" indiquant que la demande a été parachevée.
- Après réception de la réponse, le dispositif doit envoyer une réponse indiquant le succès.
- Si la réponse du gestionnaire de réseau à "Request timetable FAL PDU" indique que la demande ne peut pas être satisfaite, le dispositif doit répondre en indiquant "Insufficient bandwidth" (largeur de bande insuffisante).
- Si le dispositif répondeur a une capacité suffisante actuellement programmée pour éditer le jeu de données spécifié dans "Publish data message" le dispositif doit envoyer une réponse indiquant le succès.

Si la demande concerne la désactivation de l'édition, le dispositif répondeur doit exécuter la procédure ci-après.

- Le dispositif répondeur doit répondre par "Success".
- Le dispositif doit envoyer une demande "Delete timetable FAL PDU" (voir 8.2.4.33.1) au gestionnaire de réseau. Le gestionnaire de réseau est censé démarrer le traitement de la demande. Alors, le gestionnaire de réseau:
 - désalloue les ressources réseau pour le service Publish;
 - envoie la demande "Delete link FAL PDU" (voir 8.2.6.9.1) pour retirer la liaison (Link) de la supertrame; et
 - si requis, envoie la demande "Delete superframe FAL PDU" (voir 8.2.6.7.1) pour retirer la supertrame.

NOTE 2 Le gestionnaire de réseau envoie aussi une demande aux autres dispositifs de réseau pour supprimer la supertrame et la liaison (Link) qui avait été utilisée pour l'édition.

- Une fois les ressources réseau désallouées, le gestionnaire de réseau retourne une réponse à "Request timetable FAL PDU" indiquant que la demande a été parachevée.

Si le dispositif n'a pas la capacité réseau requise pour l'édition, il doit mettre le bit "Capacity denied" dans le statut "Standardized" (voir 8.2.3.22.2 – commande 48).

S'il n'y a qu'un seul octet dans la demande, alors:

- Si le dispositif est construit conformément à la présente norme et la valeur de ce champ n'est ni '0' ni '1', le dispositif doit répondre par "Trop peu d'octets de données reçus";
- Sinon, le dispositif doit supposer que le message Publish data message '0' est en cours d'activation ou de désactivation pour l'édition.

8.2.3.34 Read event notification summary FAL PDU

8.2.3.34.1 Primitive Request

La valeur est de cent quinze (115); La valeur du champ "Octet count" est de un (1) comme montré au Tableau 111.

Tableau 111 – Champ "Value" de la demande "Read event notification summary"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Event number ref	Ceci est utilisé pour se référer à un jeu de données d'événement dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question.

8.2.3.34.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 112. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 113.

Tableau 112 – Champ "Value" de la réponse "Read event notification summary"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Event number ref	Référence à un jeu de données d'événement dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Unsigned8	Maximum event count	Nombre maximal de messages de données d'événement pris en charge par le dispositif
2, bit 7	Bit Field	Event status	Statut d'événement tel que spécifié au Tableau F.14 – bits 7 à 4 au décalage d'octet 2
2, bit 3	Bit Field	Event notification control	Code de contrôle de notification d'événement tel que spécifié au Tableau F.13 – bits 3 à 0 au décalage d'octet 2
3	Time	Event occurrence time	Heure à laquelle le premier événement non acquitté a été déclenché; 0xFFFFFFFF si aucun événement en attente.
7	Time	Event notification retry time	Heure de nouvelle tentative de notification d'événement
11	Time	Maximum update time	Temps maximal de mise à jour
15	Time	Event de-bounce interval	Intervalle anti-rebond d'événement
19	Array of Bit Field (1 à 26)	Event mask	La concaténation du masque pour déclenchement sur les bits mis à '1' dans "Application process status" et la réponse de commande 48

**Tableau 113 – Codes de réponse spécifiques à la commande
"Read event notification summary"**

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif

8.2.3.34.3 Procédure au niveau du dispositif répondeur

Cette commande lit la configuration de l'Event Notification (notification d'événement). Si le dispositif prend en charge moins de champs de statut que le nombre maximal de champs de statut décrits dans les réponses de commande 48 (voir Tableau 84), la réponse à cette demande doit inclure "Event mask " (masque d'événements) pour les champs qui sont pris en charge par le dispositif.

8.2.3.35 Write event notification bit mask FAL PDU

8.2.3.35.1 Primitive Request

La valeur est de cent seize (116); la valeur du champ "Octet count" est de deux (2) à vingt-huit (28) comme montré au Tableau 114.

Tableau 114 – Champ "Value" de "Write event notification bit mask"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Event number ref	Référence à un jeu de données d'événement dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Array of Bit Field (1 à 26)	Event mask	La concaténation du masque pour déclenchement sur les bits mis à '1' dans "Application process status" et la réponse de commande 48

8.2.3.35.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 114. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 115.

**Tableau 115 – Codes de réponse spécifiques à la commande
"Write event notification bit mask"**

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
16	Error	Accès limité

8.2.3.35.3 Procédure au niveau du dispositif répondeur

Cette commande établit le masque d'événement qui est utilisé pour déclencher une notification d'événement. Pour chaque bit mis à '1' dans "Event mask", toute modification de la valeur correspondante de "Application process status" ou de la réponse de Command 48

déclenche une notification d'événement. Si le nombre de champs "Event mask" dans la demande est inférieur à celui des champs "Event mask" pris en charge par le dispositif, les bits de masque non spécifiés doivent être considérés comme étant '0'.

Le dispositif doit conserver le masque d'événement par le biais d'une réinitialisation du dispositif, d'un auto-test ou par le biais de la coupure puis de la remise de l'alimentation en énergie.

8.2.3.36 Write event notification timing FAL PDU

8.2.3.36.1 Primitive Request

La valeur de la Commande est de cent dix-sept (117); la valeur du champ "Octet count" est de treize (13) et le champ "Value" est tel que montré au Tableau 116.

Tableau 116 – Champ "Value" de "Write event notification timing"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Event number ref	Référence à un jeu de données d'événement dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Time	Event notification retry time	Heure de nouvelle tentative de notification d'événement
5	Time	Maximum update time	Temps maximal de mise à jour
9	Time	Event de-bounce interval	Intervalle anti-rebond d'événement
NOTE La valeur retournée dans les octets de données de réponse reflète la valeur effectivement utilisée par le dispositif.			

8.2.3.36.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 116. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 117.

Tableau 117 – Codes de réponse spécifiques à la commande "Write event notification timing"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Temps de mise à jour ou intervalle antirebond ajusté
9	Error	Demande non valide de temporisation de notification d'événement
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée.
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.3.36.3 Procédure au niveau du dispositif répondeur

Cette commande fixe les paramètres de temporisation pour la notification d'événement. Si le temps de nouvelle tentative de notification d'événement n'est pas inférieur ou égal au temps maximal de mise à jour, le dispositif doit le réajuster. S'il y a lieu, les dispositifs doivent corriger les valeurs de temporisation reçue dans la demande et répondre par "Update time or de-bounce interval adjusted" (« temps de mise à jour ou intervalle antirebond ajusté »).

L'intervalle antirebond d'événement établit la durée minimale sur laquelle le bit de statut doit rester changé (à savoir, le temps pendant lequel l'événement doit persister) dans un dispositif avant qu'une notification d'événement ne soit envoyée. Le dispositif doit effectuer un balayage pour rechercher de possibles événements au moins deux fois au cours d'un même intervalle antirebond d'événement.

Si aucun événement n'a été généré pendant un temps maximal de mise à jour, le dispositif doit éditer la réponse de Commande 119 une fois à l'attention de la passerelle en utilisant la demande "Information report".

Le dispositif doit conserver les paramètres de temporisation de la notification d'événement par le biais d'une réinitialisation du dispositif, d'un autotest ou par le biais de la coupure puis de la remise de l'alimentation en énergie.

8.2.3.37 Write event notification control FAL PDU

8.2.3.37.1 Primitive Request

La valeur est de cent dix-huit (118); La valeur du champ "Octet count" est de deux (2) comme montré au Tableau 118.

Tableau 118 – Champ "Value" de "Write event notification control"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Event number ref	Référence à un jeu de données d'événement dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Bit Field	Event notification control	Code de contrôle de notification d'événement tel que spécifié au Tableau F.13

8.2.3.37.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 118. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 119.

Tableau 119 – Codes de réponse spécifiques à la commande "Write event notification control"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Temps de mise à jour ajustés
14	Warning	Temps de mise à jour incertain – autorisé uniquement en cas de non-connexion au réseau.

Valeur	Classe	Description
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée.
33	Error	Réponse différée (DR) déclenchée – autorisé uniquement lors de l'activation d'événements dans un dispositif connecté à un réseau conforme à la présente norme
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.3.37.3 Procédure au niveau du dispositif répondeur

Cette commande est utilisée pour activer ou désactiver la notification d'événement. Si un dispositif de terrain conforme au Type 20 rejoint le réseau et il lui est refusé la capacité requise pour la notification d'événement, il doit mettre le bit 'Capacity denied' (capacité refusée) dans le "Standardized status 3" (c'est-à-dire le statut normalisé 3), voir Tableau F.11. La capacité peut être refusée en raison de demandes de données Publish ou de notification d'événement.

Le dispositif doit conserver le contrôle du mode de notification d'événement par le biais d'une réinitialisation du dispositif, d'un autotest ou par le biais de la coupure puis de la remise de l'alimentation en énergie.

8.2.3.38 Write event notification acknowledgement FAL PDU

8.2.3.38.1 Primitive Request

La valeur est de cent dix-neuf (119); la valeur du champ "Octet count" est soit de un (1), soit de huit (8) trente-trois (33) comme montré au Tableau 120. S'il n'y a qu'un seul octet, celui-ci doit être "Event number ref".

Tableau 120 – Champ "Value" de "Write event notification acknowledgement"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Event number ref	Référence à un jeu de données d'événement dans toutes les demandes et réponses read (de lecture) et write (d'écriture) pour le jeu en question
1	Time	Event occurrence time	Heure à laquelle le premier événement non acquitté a été déclenché; '-1' lorsque aucun événement n'est en attente
5	Unsigned16	Configuration change counter	Configuration change counter
7	Bit Field	Application process status	Statut de processus d'application (voir 8.2.1.4)
8	Array of Bit Field (1 à 25)	Additional device status	Données de commande 48, voir Note
NOTE Contient des données verrouillées et peut ne pas concorder avec les données de commande 48 courantes.			

8.2.3.38.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 120. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 121.

**Tableau 121 – Codes de réponse spécifiques à la commande
"Write event notification acknowledgement"**

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus; généré uniquement si le numéro d'événement ne figure pas dans les données de la demande
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Tous les événements n'ont pas été effacés

8.2.3.38.3 Procédure au niveau du dispositif répondeur

Cette commande est utilisée pour acquitter l'Event Notification (notification d'événement). Le dispositif doit comparer les valeurs de paramètres dans l'acquiescement reçu aux valeurs courantes pour les mêmes paramètres dans le dispositif – "Event occurrence time" (heure de survenue de l'événement), "Configuration change counter" (compteur de changements de configuration), "Application process status" (statut de processus d'application) et "Additional device status" (statut de dispositif supplémentaire). Si toutes les valeurs de paramètres sont égales, le dispositif doit répondre par "Success" et supprimer les bits de statut d'événement Event Status (voir Tableau 112) à '0'. Sachant qu'il peut y avoir plusieurs les transitions de bits horodatées, la commande doit seulement effacer la plus ancienne. Si tous les bits ne sont pas effacés à '0', le dispositif doit par "Not all events cleared" (Tous les événements n'ont pas été effacés.). Il s'agit certes d'une commande d'écriture, mais elle ne doit toutefois être altérée par aucun mécanisme de protection en écriture. Il est essentiel que les notifications d'événements puissent être effacées à tout moment.

La commande est tronquée après le dernier octet reçu lors de l'établissement du masque d'événements (voir la Commande 116 en 8.2.3.35). Si des octets au-delà de ceux fournis dans la Commande 116 sont reçus dans cette demande de commande, ils doivent être ignorés et ne doivent pas être retournés dans la réponse de commande. Les plus récentes valeurs pour les octets "Application process status", "Configuration change counter" et les octets de réponse de Commande 48 sont toujours incluses dans la Commande 119.

Cette commande peut être émise avec uniquement l'"Event number ref" dans les données de demande. Si cela se produit, le dispositif doit retourner les données d'événement verrouillées courantes comme montré au Tableau 120 et ne doit pas effacer les bits "Event Status" à '0'.

Le dispositif doit stocker au moins trois (3) messages de réponse (l'événement courant actuellement édité en utilisant la Commande 119 et deux de plus qui sont en tampon). Il est fortement conseillé que cinq (5) transitions soient en tampon ("bufferisées"). Il n'est pas nécessaire de les sauvegarder à travers les cycles d'alimentation en énergie.

8.2.4 Structures des APDU spécifiques aux services de gestion de couche

8.2.4.1 Write Join_key FAL PDU

8.2.4.1.1 Primitive Request

La valeur de la Commande est de sept cent soixante-huit (768); la valeur du champ "Octet count" est de un (1) et le champ "Value" est tel que montré au Tableau 122.

Tableau 122 – Champ "Value" de "Write Join_key"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Security key	Join_key	Il s'agit d'une clé réseau de 128 bits utilisée pour la sécurité.

8.2.4.1.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 122. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 123.

Tableau 123 – Codes de réponse spécifiques à la commande "Write Join_key"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit
65	Error	Échec du changement de clé

8.2.4.1.3 Procédure au niveau du dispositif répondeur

Cette commande permet à un outil de maintenance local (par exemple un dispositif tenu à la main) de modifier une Join_key de dispositif de réseau. Si un dispositif est connecté au réseau et reçoit cette commande via le port de maintenance, le dispositif doit répondre par "Access Restricted" (accès limité).

8.2.4.2 Read join status FAL PDU

8.2.4.2.1 Primitive Request

La valeur de la Commande est de sept cent soixante-neuf (769); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.4.2.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 124. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 125.

Tableau 124 – Champ "Value" de "Read join status"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Wireless mode	Mode sans fil tel que spécifié au Tableau F.27; les quatre bits de poids fort doivent être mis à zéro.
1	Bit Field	Join status	Cette énumération de 2 octets indiquent le statut de rattachement tel que spécifié au Tableau F.28
3	Unsigned8	Neighbors	Nombre de voisins disponibles
4	Unsigned8	Advertising packets	Nombre de paquets d'annonce reçus
5	Unsigned8	Max_join_retries	Nombre de tentatives de rattachement; Un trop grand nombre de tentatives de rattachement amène le dispositif à considérer que le rattachement a échoué.
6	Time	Join_rsp_time	Durée depuis que la dernière demande de rattachement a été envoyée

Décalage d'octet	Type de donnée	Nom de paramètre	Description
10	Time	Search time	Durée d'écoute pour détecter la première annonce

Tableau 125 – Codes de réponse spécifiques à la commande "Read join status"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.4.2.3 Procédure au niveau du dispositif répondeur

Cette commande permet à un système hôte ou à un dispositif tenu à la main de surveiller un dispositif de terrain à mesure qu'il effectue des transitions à travers le processus de rattachement conformément à l'Article A.2. Si le dispositif ne peut pas rejoindre le réseau, la réponse à cette commande fournit une assistance de diagnostic à un technicien instruments ou à une personne d'entretien.

8.2.4.3 Write active_search_shed_time FAL PDU

8.2.4.3.1 Primitive Request

La valeur de la Commande est de sept cent soixante-dix (770); la valeur du champ "Octet count" est de quatre (4) et le champ "Value" est tel que montré au Tableau 126.

Tableau 126 – Champ "Value" de demande "Write Active_search_shed_time"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Time	Active_search_shed_time	Durée pendant laquelle la recherche active est activée au cours de la procédure de rattachement

8.2.4.3.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 127. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 128.

Tableau 127 – Champ "Value" de réponse de "Write Active_search_shed_time"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Time	Active_search_shed_time	Durée pendant laquelle la recherche active est activée au cours de la procédure de rattachement
4	Time	Advertise_time	Intervalle auquel les DLPDU Advertise sont émises
8	Unsigned8	Neighbors	Nombre de voisins effectuant des annonces en plus de ce dispositif

Tableau 128 – Codes de réponse spécifiques à la commande "Write Active_search_shed_time"

Valeur	Classe	Description
0	Success	Aucune erreur
3	Error	Paramètre passé plus grand que permis
4	Error	Paramètre passé plus petit que permis

Valeur	Classe	Description
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Mis à la valeur la plus proche possible
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit
65	Error	Demande rejetée (par exemple opérateur neutralisé)

8.2.4.3.3 Procédure au niveau du dispositif répondeur

Cette commande demande une recherche active et rapide – voir A.2.3.2. En général, elle est reçue localement par un dispositif de réseau à partir d'un outil de maintenance local (par exemple, un dispositif tenu à la main) et est propagée vers le gestionnaire de réseau. Le gestionnaire de réseau est censé configurer ou activer une rapide annonce dans le dispositif et un ou plusieurs de ses voisins.

8.2.4.4 Write join mode configuration FAL PDU

8.2.4.4.1 Primitive Request

La valeur de la Commande est de sept cent soixante et onze (771); la valeur du champ "Octet count" est de cinq (5) et le champ "Value" est tel que montré au Tableau 129.

Tableau 129 – Champ "Value" de "Write Join mode configuration"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Join mode	Le mode de rattachement tel que spécifié au Tableau F.36
1	Time	Active_search_shed_time	Durée pendant laquelle la recherche active est activée au cours de la procédure de rattachement

8.2.4.4.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 129. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 130.

Tableau 130 – Codes de réponse spécifiques à la commande "Write Join mode configuration"

Valeur	Classe	Description
0	Success	Aucune erreur
3	Error	Paramètre passé plus grand que permis
4	Error	Paramètre passé plus petit que permis
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Mis à la valeur la plus proche possible
16	Error	Accès limité

Valeur	Classe	Description
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit
65	Error	Demande rejetée

8.2.4.4.3 Procédure au niveau du dispositif répondeur

Cette commande permet à un système hôte ou dispositif tenu à la main de forcer un dispositif de terrain à entrer dans le mode active join (rattachement actif). Le dispositif doit rester dans le mode "active search" (recherche active) pendant au moins le temps "Active_search_shed_time" (voir A.2.3.2). Ce temps écoulé, le dispositif peut entrer en mode Deep Sleep (sommeil profond) ou Ultra Low Power (puissance ultra faible).

8.2.4.5 Read Join mode configuration FAL PDU

8.2.4.5.1 Primitive Request

La valeur de la Commande est de sept cent soixante-douze (772); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.4.5.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 131. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 132.

Tableau 131 – Champ "Value" de la réponse "Read Join mode configuration"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Join mode	Le mode de rattachement tel que spécifié au Tableau F.36
1	Time	Active_search_shed_time	Durée pendant laquelle la recherche active est activée au cours de la procédure de rattachement

Tableau 132 – Codes de réponse spécifiques à la commande "Read Join mode configuration"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.4.5.3 Procédure au niveau du dispositif répondeur

Cette commande lit le mode "Join" et le temps "Active_search_shed_time".

8.2.4.6 Write network_ID FAL PDU

8.2.4.6.1 Primitive Request

La valeur de la Commande est de sept cent soixante-treize (773); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est tel que montré au Tableau 133.

Tableau 133 – Champ "Value" de "Write network_ID"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	network_ID	Identificateur de réseau tel que défini en 3.3.65

8.2.4.6.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 133. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 134.

Tableau 134 – Codes de réponse spécifiques à la commande "Write network_ID"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Changement de Network_ID change en attente; une nouvelle valeur doit être utilisée pour le prochain rattachement.
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit
65	Error	Network_ID non valide

8.2.4.6.3 Procédure au niveau du dispositif répondeur

Cette commande configure le dispositif pour reconnaître le network_ID correct. Si cette commande n'a pas été émise par le gestionnaire de réseau ou via le port de maintenance, le dispositif doit répondre par "Access Restricted" (accès limité).

Si le dispositif n'est pas connecté au réseau, le network_ID est immédiatement changé. Il convient que le dispositif entre dans le mode d'économie d'énergie et attende la commande de configuration du mode de rattachement avant de commencer à rechercher activement le réseau.

Sinon, le nouveau network_ID doit être utilisé pour le prochain processus de rattachement.

8.2.4.7 Read network_ID FAL PDU**8.2.4.7.1 Primitive Request**

La valeur de la Commande est de sept cent soixante-quatorze (774); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.4.7.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 135. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 136.

Tableau 135 – Champ "Value" de "Read network_ID"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Network_ID	Identificateur de réseau tel que défini en 3.3.65

Tableau 136 – Codes de réponse spécifiques à la commande "Read network_ID"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.4.7.3 Procédure au niveau du dispositif répondeur

Cette commande lit la valeur de réglage courante du network_ID dans le dispositif.

8.2.4.8 Write network_tag FAL PDU

8.2.4.8.1 Primitive Request

La valeur de la Commande est de sept cent soixante-quinze (775), la valeur du champ "Octet count" est de trente-deux (32) et le champ "Value" est tel que montré au Tableau 137.

Tableau 137 – Champ "Value" de "Write Network_tag"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0 – 31	Latin-1	Network_tag	Marqueur long (long_tag) devant être stocké dans le dispositif

8.2.4.8.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 137. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 138.

Tableau 138 – Codes de réponse spécifiques à la commande "Write Network_tag"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.4.8.3 Procédure au niveau du dispositif répondeur

Cette commande lit le marqueur pour le réseau. Le Network_tag est un proxy pour le network_ID et, de ce fait, le réseau peut être identifié sous une forme textuelle au niveau de l'application.

8.2.4.9 Read network_tag FAL PDU

8.2.4.9.1 Primitive Request

La valeur de la Commande est de sept cent soixante-seize (776); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.4.9.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 139. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 140.

Tableau 139 – Champ "Value" de réponse "Read Network_tag"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0 – 31	Latin-1	Long_tag	Marqueur long stocké dans le dispositif

Tableau 140 – Codes de réponse spécifiques à la commande "Read Network_tag"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé

8.2.4.10 Read wireless device capabilities FAL PDU

8.2.4.10.1 Primitive Request

La valeur de la Commande est de sept cent soixante-dix-sept (777); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.4.10.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 141. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 142.

Tableau 141 – Champ "Value" de "Read wireless device capabilities"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Power source	Source d'énergie telle que spécifiée au Tableau F.20
1	Float32	Peak packet rate	Valeur de crête du nombre de paquets par seconde
5	Time	Load duration	Durée à la charge de paquets de crête avant la consommation de l'énergie
9	Time	Recovery time	Temps pour récupérer après consommation de l'énergie
13	Integer8	Good RSL	Seuil de RSL (niveau de signal reçu en dB) pour une connexion "good" (correcte)
14	Time	Min Keep_alive_time	Temps d'entretien (Keep_alive_time) minimal requis
18	Unsigned16	Neighbor count	Nombre maximal de voisins

Décalage d'octet	Type de donnée	Nom de paramètre	Description
20	Unsigned16	Packet buffer	Nombre maximal de tampons de paquets

Tableau 142 – Codes de réponse spécifiques à la commande "Read wireless device capabilities"

Valeur	Classe	Description
0	Success	Aucune erreur

8.2.4.10.3 Procédure au niveau du dispositif répondeur

Cette PDU est utilisée par le gestionnaire de réseau pour déterminer des caractéristiques opérationnelles d'un dispositif de réseau. Le gestionnaire de réseau est censé utiliser le débit de paquets de crête (Peak) et le temps de Recovery (récupération) pour prendre en charge des dispositifs rechargeables ou récupérateurs d'énergie.

Les dispositifs alimentés par batterie d'accumulateurs doivent pouvoir supporter la fréquence de crête des paquets pendant la durée de vie spécifiée pour le dispositif. La durée de charge doit être d'une heure au moins. Si ce paramètre n'est pas applicable, sa valeur doit être mise à vingt-quatre heures.

Alors que le dispositif récupère après une consommation d'énergie, il doit pouvoir acheminer un paquet par seconde. Si ce paramètre n'est pas applicable, sa valeur doit être mise à zéro.

8.2.4.11 Read battery life FAL PDU

8.2.4.11.1 Primitive Request

La valeur de la Commande est de sept cent soixante-dix-huit (778); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est null (vide).

8.2.4.11.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 143. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 144.

Tableau 143 – Champ "Value" de réponse "Read battery life"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Battery life	Durée de vie restante de la batterie en jours

Tableau 144 – Codes de réponse spécifiques à la commande "Read battery life"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé

8.2.4.11.3 Procédure au niveau du dispositif répondeur

Cette PDU est utilisée par l'état courant de la batterie dans un dispositif de réseau. Si le dispositif n'a ni batterie ni autre composant d'accumulation d'énergie, le dispositif peut retourner 0xFFFF comme étant la valeur de la durée de vie de la batterie.

NOTE Une analyse de conception méticuleuse de la performance à long terme des est recommandée pour mettre correctement en œuvre cette commande.

8.2.4.12 Report device health FAL PDU

8.2.4.12.1 Primitive Request

Il n'y a aucune primitive "request" pour cette commande.

8.2.4.12.2 Primitive Response

La valeur de la commande est de sept cent soixante-dix-neuf (779); le champ "Value" de la réponse est montré au Tableau 145. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 146.

Tableau 145 – Champ "Value" de réponse de "Report device health"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Packets generated	Nombre de paquets générés par ce dispositif
2	Unsigned16	Packets terminated	Nombre de paquets terminés par ce dispositif
4	Unsigned8	DLL MIC failures	Nombre de défaillances détectées relatives au MIC MAC de la DLL.
5	Unsigned8	NL MIC failures	Nombre de défaillances détectées relatives au MIC (de session) de la couche NL
6	Enumeration	Power status	Statut de puissance tel que spécifié au Tableau F.33
7	Unsigned8	CRC errors	Nombre d'erreurs de CRC détectées
8	Unsigned8	Packets missed	Nombre de paquets détectés comme absents

Tableau 146 – Codes de réponse spécifiques à la commande "Report device health"

Valeur	Classe	Description
0	Success	Aucune erreur

8.2.4.12.3 Procédure au niveau du dispositif répondeur

Cette PDU est utilisée par le dispositif de terrain pour éditer des compteurs relatifs aux communications comme "Information report" (rapport d'informations). Ces compteurs doivent être réinitialisés seulement juste avant que le dispositif ne rejoigne le réseau. Ensuite, les compteurs doivent être incrémentés et repasser par zéro après dépassement de capacité. La valeur de "Packets missed" est incrémentée pour chaque zéro dans l'historique du compteur de Nonces qui est en soupassement de capacité.

8.2.4.13 Read neighbor health list FAL PDU

8.2.4.13.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingts (780); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est tel que montré au Tableau 147.

Tableau 147 – Champ "Value" de demande "Read neighbor health list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau "Neighbor" (voisins)
1	Unsigned8	Entry count	Nombre d'entrées de voisins devant être lues

8.2.4.13.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 148. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 149.

Tableau 148 – Champ "Value" de réponse de "Read neighbor health list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau "Neighbor" (voisins)
1	Unsigned8	Entry count	Nombre d'entrées de voisins contenues dans la réponse
2	Unsigned8	Neighbor count	Nombre total de voisins
3	Unsigned16	Neighbor_nickname	Surnom du voisin comme défini en 3.3.67
5	Bit Field	Neighbor flags	Fanions Neighbor (voisins) tels que spécifiés au Tableau F.34
6	Integer8	Average_RSL	RSL (niveau de signal reçu en dBm) moyen depuis le dernier rapport
7	Unsigned16	Packets_transmitted	Nombre de paquets émis vers ce voisin
9	Unsigned16	Missed_ACK_packets	émission échouée vers ce voisin – nombre de paquets attendant un ACK et aucun reçu
11	Unsigned16	Packets_received	Nombre de paquets reçus en provenance de ce voisin
13..			

^a Les champs dans les octets 3 à 12 sont répétés autant de fois que la valeur de "Neighbor count".

Tableau 149 – Codes de réponse spécifiques à la commande "Read neighbor health list"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.13.3 Procédure au niveau du dispositif répondeur

Cette commande lit les entrées du tableau Neighbor à partir du dispositif de réseau. Les voisins avec des liaisons au dispositif sont attendus au début de la liste et ceux sans liaisons le sont à la fin. Le tableau de voisins est une liste dont des entrées sont supprimées lorsqu'un voisin arrête la communication ou sont ajoutées lorsqu'un nouveau voisin est détecté. L'indice de tableau (Table index) et le compte d'entrées (Entry count) changent à la suite de l'ajout ou de la suppression d'un voisin. Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (mis à la valeur possible la plus proche)

Le dispositif doit envoyer cette réponse à une demande de lecture. En outre, le dispositif doit éditer la liste de santé des voisins (Neighbor health list) en utilisant la priorité de niveau "Process Data" comme "Information report". La période d'édition est fixée en configurant l'heure de "Health report" en utilisant la Commande 795 (voir 8.2.4.27).

Les compteurs de paquets doivent être réinitialisés seulement juste avant que le dispositif ne rejoigne le réseau. Ensuite, les compteurs doivent être incrémentés et repasser par zéro après dépassement de capacité.

Cette PDU est également utilisée par le dispositif de terrain pour éditer la santé des voisins comme "Information report" (rapport d'informations).

8.2.4.14 Read device nickname FAL PDU

8.2.4.14.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-un (781); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.14.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 150. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 151.

Tableau 150 – Champ "Value" de réponse de "Read device nickname"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Device nickname	Surnom du dispositif comme défini en 3.3.67

Tableau 151 – Codes de réponse spécifiques à la commande "Read device nickname"

Valeur	Classe	Description
0	Success	Aucune erreur

8.2.4.14.3 Procédure au niveau du dispositif répondeur

Cette commande lit un surnom de dispositif de réseau (adresse courte).

8.2.4.15 Read session list FAL PDU

8.2.4.15.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-deux (782); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est montré au Tableau 152.

Tableau 152 – Champ "Value" de demande "Read session list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau "Session"
1	Unsigned8	Entry count	Nombre d'entrées de sessions devant être lues

8.2.4.15.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 153. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 154.

Tableau 153 – Champ "Value" de réponse de "Read session list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau "Session"
1	Unsigned8	Entry count	Nombre d'entrées de sessions contenues dans la réponse
2	Unsigned8	Session count	Nombre total de sessions actives
3	Enumeration	Session_type	Session_type comme spécifié au Tableau F.24
4	Unsigned16	Peer_nickname	Surnom du dispositif de correspondant

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
6	Unique ID	Peer_unique_ID	ID propre au dispositif de correspondant tel que défini en 3.3.91
11	Unsigned32	Peer_nonce_counter	Valeur la plus élevée du compteur de nonces reçue en provenance du dispositif de correspondant.
15	Unsigned32	My_nonce_counter	Compteur de nonces pour les paquets fournis par le dispositifs
19..			

^a Les champs dans les octets 3 à 18 sont répétés autant de fois que la valeur de "Session count".

Tableau 154 – Codes de réponse spécifiques à la commande "Read session list"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.15.3 Procédure au niveau du dispositif répondeur

Cette commande lit les entrées du tableau Session à partir du dispositif de réseau. Le Tableau Session est une liste dont les entrées sont supprimées ou ajoutées. L'indice de session (Session index) et le compte d'entrées (Entry count) changent à la suite de l'addition ou de la suppression d'une session. Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (mis à la valeur possible la plus proche)

8.2.4.16 Read superframe list FAL PDU

8.2.4.16.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-trois (783); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est montré au Tableau 155.

Tableau 155 – Champ "Value" de demande "Read superframe list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau Superframe
1	Unsigned8	Entry count	Nombre d'entrées de superframes devant être lues

8.2.4.16.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 156. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 157.

Tableau 156 – Champ "Value" de réponse de "Read superframe list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau Superframe
1	Unsigned8	Entry count	Nombre d'entrées de superframes contenues dans la réponse
2	Unsigned8	Superframe count	Nombre total de superframes actives

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
3	Unsigned8	Superframe_ID	Numéro d'identification attribué à la supertrame
4	Unsigned16	Num_Slots	Nombre d'intervalles de temps dans cette supertrame, c'est-à-dire taille de la supertrame.
6	Bit Field	Superframe mode	Fanions de mode Superframe tels que spécifiés au Tableau F.23
7..			
^a Les champs dans les octets 3 à 6 sont répétés autant de fois que la valeur de "Superframe count".			

Tableau 157 – Codes de réponse spécifiques à la commande "Read superframe list"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.16.3 Procédure au niveau du dispositif répondeur

Cette commande lit les entrées du tableau Superframe à partir du dispositif de réseau. Le Tableau Superframe est une liste dont les entrées sont supprimées ou ajoutées. L'indice de supertrame (Superframe index) et le compte d'entrées (Entry count) changent à la suite de l'addition ou de la suppression d'une supertrame. Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche).

8.2.4.17 Read link list FAL PDU**8.2.4.17.1 Primitive Request**

La valeur de la Commande est de sept cent quatre-vingt-quatre (784); la valeur du champ "Octet count" est de trois (3) et le champ "Value" est montré au Tableau 158.

Tableau 158 – Champ "Value" de demande "Read link list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Table index	Indice du Tableau Link
2	Unsigned8	Entry count	Nombre d'entrées de liaisons devant être lues

8.2.4.17.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 159. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 160.

Tableau 159 – Champ "Value" de réponse de "Read link list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Table index	Indice du Tableau Link
2	Unsigned8	Entry count	Nombre d'entrées de liaisons contenues dans la réponse
3	Unsigned16	Link count	Nombre total de liaisons actives
5	Unsigned8	Link_ID	Numéro d'identification attribué à cette liaison

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
6	Unsigned16	Slot	Nombre d'intervalles de temps dans la supertrame attribué à cette liaison
8	Unsigned8	Channel_Offset	Décalage initial de la voie de saut de fréquence
9	Unsigned16	Neighbor_nickname	Surnom du voisin pour cette liaison, ou bien 0xFFFF si liaison de diffusion
11	Bit Field	Link options	Options de liaisons telles que spécifiées au Tableau F.22
12	Enumeration	Link_type	Link_type comme spécifié au Tableau F.21
13..			

^a Les champs dans les octets 5 à 12 sont répétés autant de fois que la valeur de "Link count".

Tableau 160 – Codes de réponse spécifiques à la commande "Read link list"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.17.3 Procédure au niveau du dispositif répondeur

Cette commande lit les entrées du tableau Link à partir du dispositif de réseau. Le Tableau Link est une liste dont les entrées sont supprimées ou ajoutées. L'indice de liaison (Link index) et le compte d'entrées (Entry count) changent à la suite de l'addition ou de la suppression d'une liaison. Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche).

8.2.4.18 Read graph list FAL PDU

8.2.4.18.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-cinq (785); la valeur du champ "Octet count" est de un (1) et le champ "Value" est montré au Tableau 161.

Tableau 161 – Champ "Value" de demande "Read graph list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau Graph

8.2.4.18.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 162. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 163.

Tableau 162 – Champ "Value" de réponse de "Read graph list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau Graph
1	Unsigned8	Graph count	Nombre total de graphes
2	Unsigned16	Graph_ID	Numéro d'identification attribué à ce graphe

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
4	Unsigned8	Neighbor count	Nombre de voisins
5	Unsigned16	Neighbor_nickname	Surnom de voisin pour ce graphe
7..			

^a Les champs dans les octets 5 à 6 sont répétés autant de fois que la valeur de "Neighbor count".

Tableau 163 – Codes de réponse spécifiques à la commande "Read graph list"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.18.3 Procédure au niveau du dispositif répondeur

Cette commande lit un élément dans le tableau Graph (graphes) à partir du dispositif de réseau. L'indice sert uniquement de référence et le graphe référencé peut changer si un graphe placé avant lui sur la liste est supprimé. Par conséquent, en cas de non-validité du paramètre dans la demande, le dispositif doit répondre avec la plus proche valeur valide du paramètre et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche)

8.2.4.19 Read neighbor attribute flag FAL PDU**8.2.4.19.1 Primitive Request**

La valeur de la Commande est de sept cent quatre-vingt-six (786); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est montré au Tableau 164.

Tableau 164 – Champ "Value" de demande "Read neighbor attribute flag"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Neighbor_nickname	Surnom du voisin comme défini en 3.3.67

8.2.4.19.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 165. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 166.

Tableau 165 – Champ "Value" de réponse de "Read neighbor attribute flag"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Neighbor_nickname	Surnom du voisin comme défini en 3.3.67
2	Bit Field	Neighbor flags	Attributs de Neighbor (voisins) tels que spécifiés au Tableau F.34

Tableau 166 – Codes de réponse spécifiques à la commande "Read neighbor attribute flag"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
65	Error	Neighbor_nickname inconnu

8.2.4.19.3 Procédure au niveau du dispositif répondeur

Cette commande lit les attributs d'un voisin à partir du dispositif de réseau.

8.2.4.20 Read neighbor signal level FAL PDU

8.2.4.20.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-sept (787); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est montré au Tableau 167.

Tableau 167 – Champ "Value" de demande "Read neighbor signal level"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau "Neighbor" (voisins)
1	Unsigned8	Entry count	Nombre d'entrées de voisins devant être lues

8.2.4.20.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 168. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 169.

Tableau 168 – Champ "Value" de réponse de "Read neighbor signal level"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau "Neighbor" (voisins)
1	Unsigned8	Entry count	Nombre d'entrées de voisins contenues dans la réponse
2	Unsigned8	Neighbor count	Nombre total de voisins
3	Unsigned16	Neighbor_nickname	Surnom du voisin comme défini en 3.3.67
5	Integer8	Average_RSL	RSL (niveau de signal reçu en dBm) moyen de voisin
6..			

^a Les champs dans les octets 3 à 5 sont répétés autant de fois que la valeur de "Neighbor count".

Tableau 169 – Codes de réponse spécifiques à la commande "Read neighbor signal level"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.20.3 Procédure au niveau du dispositif répondeur

Cette commande lit le niveau de signal du Tableau Neighbor (voisins) à partir du dispositif de réseau. Le tableau de voisins est une liste dont des entrées sont supprimées lorsqu'un voisin arrête la communication ou sont ajoutées lorsqu'un nouveau voisin est détecté. L'indice de tableau (Table index) et le compte d'entrées (Entry count) changent à la suite de l'addition ou de la suppression d'un voisin. Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche)

Le dispositif doit envoyer cette réponse à une demande de lecture. En outre, le dispositif doit éditer le niveau de signal de voisin (Neighbor signal level) en utilisant la priorité de niveau "Process Data" comme "Information report". La période d'édition est fixée en configurant l'heure de "Health report" en utilisant la Commande 795 (voir 8.2.4.27).

8.2.4.21 Report path down alarm FAL PDU

8.2.4.21.1 Primitive Request

Il n'y a aucune primitive "request" pour ce service. La valeur de la Commande est de sept cent quatre-vingt-huit (788).

8.2.4.21.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 170. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 171.

Tableau 170 – Champ "Value" de réponse de "Report path down alarm"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Neighbor_nickname	Surnom du voisin comme défini en 3.3.67

Tableau 171 – Codes de réponse spécifiques à la commande "Report path down alarm"

Valeur	Classe	Description
0	Success	Aucune erreur
16	Error	Accès limité

8.2.4.21.3 Procédure au niveau du dispositif répondeur

Si le dispositif ne peut pas communiquer avec l'un de ses voisins pendant un temps dépassant Path_fail_time, il doit éditer la réponse d'alarme "Path down" (trajet défaillant) en utilisant la priorité de niveau "Alarm" comme "Information report" (rapport d'informations). Le temps Path_fail_time est configuré par la Commande 795 (voir 8.2.4.27).

8.2.4.22 Report source route failed alarm FAL PDU

8.2.4.22.1 Primitive Request

Il n'y a aucune primitive "request" pour ce service. La valeur de la Commande est de sept cent quatre-vingt-neuf (789).

8.2.4.22.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 172. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 173.

Tableau 172 – Champ "Value" de réponse de "Report source route failed alarm"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Unreachable neighbor	Surnom de voisin injoignable dans le chemin de source
2	Unsigned32	NL MIC	MIC de la couche réseau issu de la NPDU qui a échoué à l'acheminement

Tableau 173 – Codes de réponse spécifiques à la commande "Report source route failed alarm"

Valeur	Classe	Description
0	Success	Aucune erreur
16	Error	Accès limité

8.2.4.22.3 Procédure au niveau du dispositif répondeur

Si le dispositif détecte une défaillance de chemin de source, il doit éditer la réponse d'alarme "Source route failed" (chemin de source défaillant) en utilisant la priorité de niveau "Alarm" comme "Information report" (rapport d'informations).

8.2.4.23 Report graph route failed alarm FAL PDU

8.2.4.23.1 Primitive Request

Il n'y a aucune primitive "request" pour ce service. La valeur de la Commande est de sept cent quatre-vingt-dix (790).

8.2.4.23.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 174. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 175.

Tableau 174 – Champ "Value" de réponse de "Report graph route failed alarm"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Graph_ID	Graph_ID du chemin défaillant

Tableau 175 – Codes de réponse spécifiques à la commande "Report graph route failed alarm"

Valeur	Classe	Description
0	Success	Aucune erreur
16	Error	Accès limité

8.2.4.23.3 Procédure au niveau du dispositif répondeur

Si le dispositif détecte une défaillance de chemin de graphe, il doit éditer la réponse d'alarme "Graph route failed" (chemin de graphe défaillant) en utilisant la priorité de niveau "Alarm" comme "Information report" (rapport d'informations).

8.2.4.24 Report transport layer failed alarm FAL PDU

8.2.4.24.1 Primitive Request

Il n'y a aucune primitive "request" pour ce service. La valeur de la Commande est de sept cent quatre-vingt-onze (791).

8.2.4.24.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 176. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 177.

Tableau 176 – Champ "Value" de réponse de "Report transport layer failed alarm"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned16	Unreachable peer	Surnom d'un homologue injoignable dans la couche de transport de bout en bout

Tableau 177 – Codes de réponse spécifiques à la commande "Report transport layer failed alarm"

Valeur	Classe	Description
0	Success	Aucune erreur
16	Error	Accès limité

8.2.4.24.3 Procédure au niveau du dispositif répondeur

Si le dispositif détecte une défaillance de session, il doit éditer la réponse d'alarme "Transport layer connection failed" (connexion de couche transport défaillante) en utilisant la priorité de niveau "Alarm" comme "Information report" (rapport d'informations).

8.2.4.25 Write UTC time mapping FAL PDU

8.2.4.25.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-treize (793); la valeur du champ "Octet count" est de sept (7) et le champ "Value" est tel que montré au Tableau 178.

Tableau 178 – Champ "Value" de "Write UTC time mapping"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Date	Date	Date
3	Time	Time of day	Heure du jour

8.2.4.25.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 178. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 179.

Tableau 179 – Codes de réponse spécifiques à la commande "Write UTC time mapping"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
16	Error	Accès limité

8.2.4.25.3 Procédure au niveau du dispositif répondeur

Cette commande établit la date et l'heure reçues dans la demande comme instance dans le temps auquel l'ASN zéro a débuté. La date et l'heure dans la demande peuvent être le temps TUC ou le temps local ou n'importe quel autre temps utilisé par le demandeur. Si cette commande n'a pas été émise par le gestionnaire de réseau, le dispositif doit répondre par "Access Restricted" (accès limité).

8.2.4.26 Read UTC time mapping FAL PDU

8.2.4.26.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-quatorze (794); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.26.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 180. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 181.

Tableau 180 – Champ "Value" de réponse de "Read UTC time mapping"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Date	Date	Date
3	Time	Time of day	Heure du jour

Tableau 181 – Codes de réponse spécifiques à la commande "Read UTC time mapping"

Valeur	Classe	Description
0	Success	Aucune erreur

8.2.4.26.3 Procédure au niveau du dispositif répondeur

Cette commande lit le temps auquel l'ASN zéro a débuté, comme réglé dans le dispositif de réseau.

8.2.4.27 Write timer interval FAL PDU

8.2.4.27.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-quinze (795); la valeur du champ "Octet count" est de cinq (5) et le champ "Value" est tel que montré au Tableau 182.

Tableau 182 – Champ "Value" de "Write timer interval"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Énumération	Timer_type	(Type de compteur comme spécifié au Tableau F.19
1	Unsigned32	Intervalle	Intervalle de temporisateur en ms

8.2.4.27.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 182. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 183.

Tableau 183 – Codes de réponse spécifiques à la commande "Write timer interval"

Valeur	Classe	Description
0	Success	Aucune erreur
3	Error	Paramètre passé plus grand que permis par le dispositif
4	Error	Paramètre passé plus petit que permis par le dispositif
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Mis à la valeur la plus proche possible
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit
65	Error	Timer_type non valide
66	Error	Intervalle de temporisateur non valide

8.2.4.27.3 Procédure au niveau du dispositif répondeur

Cette commande établit la durée pour le temporisateur spécifié par le Timer_type (type de temporisateur) dans un dispositif de réseau.

8.2.4.28 Read timer interval FAL PDU**8.2.4.28.1 Primitive Request**

La valeur de la Commande est de sept cent quatre-vingt-seize (796); la valeur du champ "Octet count" est de un (1) et le champ "Value" est montré au Tableau 184.

Tableau 184 – Champ "Value" de demande "Read timer interval"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Timer_type	Type de compteur comme spécifié au Tableau F.19

8.2.4.28.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 185. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 186.

Tableau 185 – Champ "Value" de réponse de "Read timer interval"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Timer_type	Type de compteur comme spécifié au tableau F 19
1	Unsigned32	Intervalle	Intervalle de temporisateur en ms

Tableau 186 – Codes de réponse spécifiques à la commande "Read timer interval"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée
65	Error	Timer_type non valide

8.2.4.28.3 Procédure au niveau du dispositif répondeur

Cette commande lit la durée pour le temporisateur spécifié par le Timer_type (type de temporisateur) dans un dispositif de réseau.

8.2.4.29 Write radio power output FAL PDU

8.2.4.29.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-dix-sept (797); la valeur du champ "Octet count" est de un (1) et le champ "Value" est tel que montré au Tableau 187.

Tableau 187 – Champ "Value" de "Write radio power output"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Integer8	Output power	Puissance de sortie en dBm

8.2.4.29.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 187. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 188.

Tableau 188 – Codes de réponse spécifiques à la commande "Write radio power output"

Valeur	Classe	Description
0	Success	Aucune erreur
3	Error	Paramètre passé plus grand que permis
4	Error	Paramètre passé plus petit que permis
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture

Valeur	Classe	Description
8	Warning	Mis à la valeur la plus proche possible
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.4.29.3 Procédure au niveau du dispositif répondeur

Cette commande établit la puissance de sortie radio pour un dispositif ou une passerelle. Si le dispositif ne peut pas accepter la valeur exacte envoyée depuis l'application, il doit l'établir à la valeur la plus proche qu'il peut et doit répondre par "Set to the nearest possible value" (c'est-à-dire mise à la valeur possible la plus proche). Chaque dispositif doit supporter au moins les valeurs suivantes -10 dBm, 0 dBm et +10 dBm. Si le dispositif répondeur est connecté au réseau et si cette commande n'a pas été émise par le gestionnaire de réseau, le dispositif doit répondre par "Access Restricted" (accès limité). Sinon, si le dispositif répondeur n'est pas connecté au réseau et si cette commande n'a pas été reçue via le port de maintenance, le dispositif doit répondre par "Access Restricted" (accès limité).

8.2.4.30 Read radio power output FAL PDU

8.2.4.30.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-dix-huit (798); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.30.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 189. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 190.

Tableau 189 – Champ "Value" de réponse de "Read radio power output"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Integer8	Output power	Puissance de sortie en dBm

Tableau 190 – Codes de réponse spécifiques à la commande "Read radio power output"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.4.30.3 Procédure au niveau du dispositif répondeur

Cette commande lit la puissance de sortie radio dans un dispositif ou une passerelle.

8.2.4.31 Request timetable FAL PDU

8.2.4.31.1 Primitive Request

La valeur de la Commande est de sept cent quatre-vingt-dix-neuf (799); la valeur du champ "Octet count" est de dix (10) et le champ "Value" est tel que montré au Tableau 191.

Tableau 191 – Champ "Value" de "Request TimeTable"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Timetable_ID	Identificateur d'indicateur horaire fourni par un dispositif demandeur auquel il est spécifique; la valeur doit être située dans la plage de 0x00 à 0x7F.
1	Bit Field	Timetable_type	Fanions de demande d'indicateur horaire comme spécifiés au Tableau F.15
2	Enumeration	Application_domain	Domaine d'application tel que spécifié au Tableau F.16
3	Unsigned16	Peer_nickname	Surnom de l'homologue avec lequel le Timetable (indicateur horaire) est demandé
5	Time	Service_duration	Si service intermittent, alors latence, sinon période (communications cycliques)
9	Unsigned8	Route_ID	Identificateur de chemin pour ce Timetable

8.2.4.31.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 191. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 192.

Tableau 192 – Codes de réponse spécifiques à la commande "Request TimeTable"

Valeur	Classe	Description
0	Success	Aucune erreur
4	Error	Paramètre passé plus petit que permis
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Mis à la valeur la plus proche possible
16	Error	Accès limité
14	Warning	Largeur de bande de communications fournie considérablement inférieure à celle demandée (réduite plus que le réglage à la valeur la plus proche)
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit
65	Error	Demande d'indicateur horaire refusée
66	Error	Timetable_type inconnu
67	Error	Application_domain inconnu
68	Error	Peer_nickname inconnu
69	Error	Timetable_ID non valide

8.2.4.31.3 Procédure au niveau du dispositif répondeur

Cette commande doit être utilisée par un dispositif sans fil pour ajouter un indicateur horaire à un autre dispositif sans fil avec des caractéristiques spécifiées de largeur de bande et de latence. Une réponse sans erreurs à cette commande indique que le gestionnaire de réseau a accepté la demande et tentera de traiter la demande. Le Timetable_ID est unique seulement au sein du dispositif demandeur.

Sur un réseau intensément utilisé, le gestionnaire de réseau peut refuser des demandes de largeur de bande de communications ou fournir une largeur de bande considérablement moindre que celle demandée. Lorsque cela se produit, le dispositif doit mettre l'état "Capacity Denied" (capacité refusée) dans "Standardized status 3" (voir Tableau F.11) et le bit "More status available" (plus de statuts disponibles) dans l'octet "Application process status" (Statut de processus d'application).

Si le dispositif reçoit une commande "Write Timetable" (voir Commande 973) sur le même Timetable_ID alors qu'il attend une réponse à cette commande, le dispositif doit rejeter la nouvelle commande "Write Timetable".

8.2.4.32 Read timetable list FAL PDU

8.2.4.32.1 Primitive Request

La valeur de la Commande est de huit cents (800); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est montré au Tableau 193.

Tableau 193 – Champ "Value" de demande "Read TimeTable list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau Timetable (indicateur horaire)
1	Unsigned8	Entry count	Nombre d'entrées de Timetable devant être lues

8.2.4.32.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 194. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 195.

Tableau 194 – Champ "Value" de réponse de "Read TimeTable list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice du Tableau Timetable (indicateur horaire)
1	Unsigned8	Entry count	Nombre d'entrées de Timetable contenues dans la réponse
2	Unsigned8	Timetable count	Nombre total de Timetables (indicateurs horaires) actifs
3	Unsigned8	Timetable_ID	Identificateur de Timetable de l'entrée
4	Bit Field	Timetable_type	Fanions de demande d'indicateur horaire comme spécifiés au Tableau F.15
5	Enumeration	Application_domain	Domaine d'application tel que spécifié au Tableau F.16
6	Unsigned16	Peer_nickname	Surnom de l'homologue avec lequel le service est établi
8	Time	Service_duration	Si service intermittent, alors latence, sinon période (communications cycliques)
12	Unsigned8	Route_ID	Identificateur de chemin pour ce Timetable
13..			

^a Les champs dans les octets 3 à 12 sont répétés autant de fois que la valeur de "Timetable count".

Tableau 195 – Codes de réponse spécifiques à la commande "Read TimeTable list"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
8	Warning	Mis à la valeur la plus proche possible

8.2.4.32.3 Procédure au niveau du dispositif répondeur

Cette commande lit les entrées du tableau Timetable à partir du dispositif de réseau. Le Tableau Timetable est une liste dont les entrées sont supprimées ou ajoutées. L'indice d'indicateur horaire (Timetable index) et le compte d'entrées (Entry count) changent à la suite de l'addition ou de la suppression d'un Timetable. Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche). Si l'indice "Timetable index" est supérieur au nombre maximal d'indicateurs horaires (Timetable) actuellement contenus dans le dispositif, le dispositif doit retourner la dernière entrée et mettre le code de réponse à "Set to nearest possible value" (mis à la valeur la plus proche). Le compte 'Entry count' doit être mis à une valeur valide et le code de réponse "Set to nearest possible value" retourné.

8.2.4.33 Delete timetable FAL PDU

8.2.4.33.1 Primitive Request

La valeur de la Commande est de huit cent un (801); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est tel que montré au Tableau 196.

Tableau 196 – Champ "Value" de demande "Delete TimeTable"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Timetable_ID	Identificateur de Timetable spécifique au dispositif
1	Enumeration	Reason	Raison de la suppression de Timetable telle que spécifiée au Tableau F.25

8.2.4.33.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 197. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 198.

Tableau 197 – Champ "Value" de réponse de "Delete TimeTable"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Timetable_ID	Identificateur de Timetable spécifique au dispositif
1	Enumeration	Reason	Raison de la suppression de Timetable telle que spécifiée au Tableau F.25
2	Unsigned8	Timetable count	Nombre total de Timetables (indicateurs horaires) restants

Tableau 198 – Codes de réponse spécifiques à la commande "Delete TimeTable"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
16	Error	Accès limité
65	Error	Entrée introuvable
66	Error	Code de raison non valide
67	Error	Code de raison rejeté, Timetable pas supprimé
68	Error	Suppression interdite - le demandeur n'est pas le propriétaire du Timetable

8.2.4.33.3 Procédure au niveau du dispositif répondeur

Cette commande notifie la suppression de Timetable à un dispositif. L'opération suppression peut être provoquée par la demande d'un homologue ou par décision du gestionnaire de réseau. Seul le créateur de la Timetable peut supprimer une Timetable. Si le propriétaire du Timetable ne génère pas cette commande, le destinataire doit rejeter la commande et retourner le code de réponse "Delete not allowed" (suppression interdite).

8.2.4.34 Read route list FAL PDU**8.2.4.34.1 Primitive Request**

La valeur de la Commande est de huit cent deux (802); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est montré au Tableau 199.

Tableau 199 – Champ "Value" de demande "Read route list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice de Tableau Route (chemins)
1	Unsigned8	Entry count	Nombre d'entrées de chemins devant être lues

8.2.4.34.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 200. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 201.

Tableau 200 – Champ "Value" de réponse de "Read route list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Table index	Indice de Tableau Route (chemins)
1	Unsigned8	Entry count	Nombre d'entrées de chemins contenues dans la réponse
2	Unsigned8	Route count	Nombre total de chemins actifs
3	Unsigned8	Routes remaining	Nombre d'entrées de chemins non utilisées dans le dispositif
4	Unsigned8	Route_ID	Identificateur de chemin pour cette entrée
5	Unsigned16	Destination nickname	Surnom du dispositif de destination du chemin
7	Unsigned16	Graph_ID	Graph_ID de ce chemin
9	Unsigned8	Source route flag	Chemin de source attaché à ce chemin – 1: Oui, 0: Non
10..			

^a Les champs dans les octets 3 à 9 sont répétés autant de fois que la valeur de " Route count".

Tableau 201 – Codes de réponse spécifiques à la commande "Read route list"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible

8.2.4.34.3 Procédure au niveau du dispositif répondeur

Cette commande lit les entrées du tableau Route à partir du dispositif de réseau. Le Tableau Route est une liste dont les entrées sont supprimées ou ajoutées. L'indice de chemin (Route index) et le compte d'entrées (Entry count) changent à la suite de l'addition ou de la suppression d'un "Route". Par conséquent, en cas de non-validité de paramètre(s) dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche).

8.2.4.35 Read source route FAL PDU

8.2.4.35.1 Primitive Request

La valeur de la Commande est de huit cent trois (803); la valeur du champ "Octet count" est de un (1) et le champ "Value" est montré au Tableau 202.

Tableau 202 – Champ "Value" de demande "Read source route"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Route_ID	Identificateur de chemin pour lequel le chemin de source doit être lu

8.2.4.35.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 203. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 204.

Tableau 203 – Champ "Value" de réponse de "Read source route"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Route_ID	Identificateur de chemin pour lequel le chemin de source doit être lu
1	Unsigned8	Hop count	Nombre de sauts dans ce chemin
2	Unsigned16	Hop	Surnom de l'entrée de saut 0
4..			

^a Les champs dans les octets 2 à 3 sont répétés autant de fois que la valeur de " Hop count"

Tableau 204 – Codes de réponse spécifiques à la commande "Read source route"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
65	Error	Entrée introuvable

8.2.4.35.3 Procédure au niveau du dispositif répondeur

Cette commande lit le contenu d'un chemin Source particulier. Les adresses de diffusion ne sont pas des valeurs d'adresses légitimes dans les chemins Source et ne doivent être incluses dans aucune réponse pour cette commande.

8.2.4.36 Read CCA mode FAL PDU**8.2.4.36.1 Primitive Request**

La valeur de la Commande est de huit cent quatre (804); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.36.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 205. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 206.

Tableau 205 – Champ "Value" de réponse de "Read CCA mode"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	CCA Mode	Mode d'évaluation de voie libre – 1: activé, 0: désactivé

Tableau 206 – Codes de réponse spécifiques à la commande "Read CCA mode"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.4.36.3 Procédure au niveau du dispositif répondeur

Cette commande lit le mode Évaluation de voie libre (CCA) dans un dispositif de réseau.

8.2.4.37 Write CCA mode FAL PDU**8.2.4.37.1 Primitive Request**

La valeur de la Commande est de huit cent cinq (805); la valeur du champ "Octet count" est de un (1) et le champ "Value" est tel que montré au Tableau 207.

Tableau 207 – Champ "Value" de "Write CCA mode"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	CCA Mode	Mode d'évaluation de voie libre – 1: activé, 0: désactivé

8.2.4.37.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 207. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 208.

Tableau 208 – Codes de réponse spécifiques à la commande "Write CCA mode"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide (par exemple, mode CCA non valide)
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée.
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte
36	Error	Réponse différée (DR) en conflit

8.2.4.37.3 Procédure au niveau du dispositif répondeur

Cette commande écrit le mode Évaluation de voie libre (CCA) dans un dispositif de réseau. Si la commande est reçue en provenance d'une autre source que le gestionnaire de réseau, le dispositif doit rejeter cette demande avec le code de réponse "Access Restricted" (accès limité).

8.2.4.38 Read handheld superframe FAL PDU

8.2.4.38.1 Primitive Request

La valeur de la Commande est de huit cent six (806); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.38.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 209. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 210.

Tableau 209 – Champ "Value" de réponse de "Read handheld superframe"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Superframe_ID	Numéro d'identification attribué à la supertrame
1	Unsigned16	Num_Slots	Nombre d'intervalles de temps dans cette supertrame, taille de la supertrame.
3	Bit Field	Superframe mode	Fanions de mode Superframe tels que spécifiés au Tableau F.23

**Tableau 210 – Codes de réponse spécifiques à la commande
"Read handheld superframe"**

Valeur	Classe	Description
0	Success	Aucune erreur
9	Error	Absence de super trame contrnue

8.2.4.38.3 Procédure au niveau du dispositif répondeur

Cette commande est utilisée pour déterminer si une super trame contenue dans un dispositif de réseau est activé.

8.2.4.39 Write handheld superframe FAL PDU

8.2.4.39.1 Primitive Request

La valeur de la Commande est de huit cent sept (807); la valeur du champ "Octet count" est de deux (2) et le champ "Value" est tel que montré au Tableau 211.

Tableau 211 – Champ "Value" de "Write handheld superframe"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Superframe_ID	Numéro d'identification attribué à la supertrame
1	Bit Field	Superframe mode	Fanions de mode Superframe tels que spécifiés au Tableau F.23

8.2.4.39.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 211. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 212.

Tableau 212 – Codes de réponse spécifiques à la commande "Write handheld superframe"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide (par exemple, mode supertrame non valide)
5	Error	Trop peu d'octets de données reçus
16	Error	Accès limité

8.2.4.39.3 Procédure au niveau du dispositif répondeur

Cette commande permet à un dispositif tenu à la main de demander que le dispositif de réseau active la supetrame contenue afin que la maintenance puisse être effectuée. La supertrame tenue à la main est utilisée entre un dispositif de réseau et un dispositif tenu à la main essentiellement à des fins de maintenance.

8.2.4.40 Lecture du paquet "hop-to-live FAL PDU"

8.2.4.40.1 Primitive Request

La valeur de la Commande est de huit cent huit (808); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.40.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 213. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 214.

Tableau 213 – Champ "Value" de réponse de "Read packet hop-to-live"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Hop-to-live	Valeur actuellement configurée de Hop-to-live

Tableau 214 – Codes de réponse spécifiques à la commande "Read packet hop-to-live"

Valeur	Classe	Description
0	Success	Aucune erreur
6	Error	Erreur de commande spécifique au dispositif
32	Error	Occupé – une réponse différée n'a pas pu être lancée

8.2.4.40.3 Procédure au niveau du dispositif répondeur

Cette commande lit la valeur courante du Hop-to-live configuré pour un dispositif de réseau.

8.2.4.41 Write packet hop-to-live FAL PDU

8.2.4.41.1 Primitive Request

La valeur de la Commande est de huit cent neuf (809); la valeur du champ "Octet count" est de un (1) et le champ "Value" est tel que montré au Tableau 215.

Tableau 215 – Champ "Value" de "Write packet hop-to-live"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Hop-to-live	Valeur de Hop-to-live

8.2.4.41.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 215. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 216.

Tableau 216 – Codes de réponse spécifiques à la commande "Write packet hop-to-live"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
6	Error	Erreur de commande spécifique au dispositif
7	Error	Dans le mode protection en écriture
8	Warning	Mis à la valeur la plus proche
16	Error	Accès limité
32	Error	Occupé – une réponse différée n'a pas pu être lancée
33	Error	Réponse différée (DR) déclenchée
34	Error	Réponse différée (DR) en cours
35	Error	Réponse différée (DR) morte

Valeur	Classe	Description
36	Error	Réponse différée (DR) en conflit

8.2.4.41.3 Procédure au niveau du dispositif répondeur

Cette commande écrit la valeur du Hop-to-live dans un dispositif de réseau. Si la valeur de Hop-to-live reçue dans la demande est inférieure à huit (8), cette valeur doit être passée à huit (8) et le code de réponse "Set to the nearest value" (mise à la valeur la plus proche) doit être retourné. Si la commande est reçue en provenance d'une autre source que le gestionnaire de réseau, le dispositif doit rejeter cette demande avec le code de réponse "Access Restricted" (accès limité).

8.2.4.42 Read Join_priority FAL PDU

8.2.4.42.1 Primitive Request

La valeur de la Commande est de huit cent dix (810); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.42.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 217. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 218.

Tableau 217 – Champ "Value" de réponse de "Read Join_priority"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Join_priority	Join_priority à laquelle le dispositif doit faire l'annonce; un nombre inférieur indique un meilleur choix pour le rattachement

Tableau 218 – Codes de réponse spécifiques à la commande "Read Join_priority"

Valeur	Classe	Description
0	Success	Aucune erreur

8.2.4.42.3 Procédure au niveau du dispositif répondeur

Cette commande lit la valeur courante de Join_priority dans un dispositif de réseau.

8.2.4.43 Write Join_priority FAL PDU

8.2.4.43.1 Primitive Request

La valeur de la Commande est de huit cent onze (811); la valeur du champ "Octet count" est de un (1) et le champ "Value" est tel que montré au Tableau 219.

Tableau 219 – Champ "Value" de "Write Join_priority"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Join_priority	Join_priority à laquelle le dispositif doit faire l'annonce; un nombre inférieur indique un meilleur choix pour le rattachement

8.2.4.43.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 219. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 220.

Tableau 220 – Codes de réponse spécifiques à la commande "Write Join_priority"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche
16	Error	Accès limité

8.2.4.43.3 Procédure au niveau du dispositif répondeur

Cette commande écrit la valeur de Join_priority dans un dispositif de réseau. Si la valeur de Join_priority reçue dans la demande est supérieure à quinze (15), cette valeur doit être passée à quinze (15) et le code de réponse "Set to the nearest value" (mise à la valeur la plus proche) doit être retourné. Si la commande est reçue en provenance d'une autre source que le gestionnaire de réseau, le dispositif doit rejeter cette demande avec le code de réponse "Access Restricted" (accès limité).

8.2.4.44 Read receive Priority_threshold FAL PDU

8.2.4.44.1 Primitive Request

La valeur de la Commande est de huit cent douze (812); la valeur du champ "Octet count" est de zéro (0) et le champ "Value" est "null" (vide).

8.2.4.44.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 221. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 222.

Tableau 221 – Champ "Value" de réponse de "Read receive Priority_threshold"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Priority_threshold	La DLPDU ayant la priorité la plus faible devant être acceptée en provenance d'un autre dispositif.

Tableau 222 – Codes de réponse spécifiques à la commande "Read receive Priority_threshold"

Valeur	Classe	Description
0	Success	Aucune erreur

8.2.4.44.3 Procédure au niveau du dispositif répondeur

Cette commande lit la valeur courante de Receive Priority_threshold (seuil de priorité de réception) dans un dispositif de réseau.

8.2.4.45 Write receive Priority_threshold FAL PDU

8.2.4.45.1 Primitive Request

La valeur de la Commande est de huit cent treize (813); la valeur du champ "Octet count" est de un (1) et le champ "Value" est tel que montré au Tableau 223.

Tableau 223 – Champ "Value" de "Write receive Priority_threshold"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Unsigned8	Priority_threshold	La DLPDU ayant la priorité la plus faible devant être acceptée en provenance d'un autre dispositif.

8.2.4.45.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 223. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 224.

Tableau 224 – Codes de réponse spécifiques à la commande "Write receive Priority_threshold"

Valeur	Classe	Description
0	Success	Aucune erreur
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche
16	Error	Accès limité

8.2.4.45.3 Procédure au niveau du dispositif répondeur

Cette commande écrit la valeur de Receive Priority_threshold dans un dispositif de réseau. Les valeurs permises pour ce paramètre sont zéro (0) à trois (3). Si une toute autre valeur est reçue dans la demande, le dispositif doit mettre la valeur du paramètre à '3' et retourner la réponse "Set to the nearest value" (mise à la valeur la proche). Si la commande est reçue en provenance d'un autre source que le gestionnaire de réseau, le dispositif doit rejeter cette demande avec le code de réponse "Access Restricted" (accès limité).

8.2.4.46 Read device list FAL PDU

8.2.4.46.1 Primitive Request

La valeur de la Commande est de huit cent quatorze (814); la valeur du champ "Octet count" est de quatre (4) et le champ "Value" est montré au Tableau 225.

Tableau 225 – Champ "Value" de demande "Read device list"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Device code	Code de liste de dispositif tel que spécifié au Tableau F.30
1	Unsigned8	Entry count	Nombre d'entrées de dispositifs devant être lues
2	Unsigned16	Table index	Indice de début du Tableau de dispositifs

8.2.4.46.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 226. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 227.

Tableau 226 – Champ "Value" de réponse de "Read device list"

Décalage d'octet ^a	Type de donnée	Nom de paramètre	Description
0	Enumeration	Device code	Code de liste de dispositif tel que spécifié au Tableau F.30
1	Unsigned8	Entry count	Nombre d'entrées de dispositifs contenues dans la réponse
2	Unsigned16	Table index	Indice de début du Tableau de dispositifs
4	Unsigned16	Device count	Nombre total d'entrées de dispositifs dans le tableau
6	Unique ID	Unique ID	L'identificateur unique attribué au dispositif tel que défini en 3.3.91 et décrit en 6.2.1.2.4
11..			
^a Les champs dans les octets 6 à 10 sont répétés autant de fois que la valeur de " Device count".			

Tableau 227 – Codes de réponse spécifiques à la commande "Read device list"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide, c'est-à-dire code de liste de dispositifs non pris en charge
5	Error	Trop peu d'octets de données reçus
8	Warning	Mis à la valeur la plus proche possible
64	Error	Commande non mise en œuvre

8.2.4.46.3 Procédure au niveau du dispositif répondeur

Un dispositif de réseau peut maintenir plusieurs Tableaux Device avec une liste de dispositifs. Les Tableaux Device actuellement pris en charge sont – 'Active list' (liste active), 'White list' (liste blanche) et 'Black list' (liste noire). Cette commande lit les entrées dans le Tableau Device pour le type de liste spécifié par le "Device code" dans la demande. Le Tableau Device est une liste de dispositifs dont les entrées sont supprimées ou ajoutées. L'indice de dispositif ("Device index") et le compte d'entrées ("Entry count") changent à la suite de l'addition ou de la suppression d'un dispositif. Par conséquent, en cas de non-validité de paramètre(s) "Entry count" ou "Table index" dans la demande, le dispositif doit répondre avec la plus proche valeur valide du/des paramètre(s) et envoyer le code de réponse "Set to the nearest possible value" (c'est-à-dire mis à la valeur possible la plus proche).

La passerelle et le gestionnaire de réseau doivent maintenir le Tableau Device pour 'Active list' et prendre en charge cette commande. Il convient que le gestionnaire de réseau maintienne le Tableau Device pour 'White list' et 'Black list'. Il convient que tous les dispositifs de réseau maintiennent le Tableau Device pour 'Black list'. Si un dispositif de réseau ne maintient pas le Tableau Device pour 'Black list' et le "device code" dans la demande est pour 'Black list', il doit répondre à cette commande par "Command not implemented" (« commande non mise en œuvre »).

8.2.4.47 Add device list entry FAL PDU

8.2.4.47.1 Primitive Request

La valeur de la Commande est de huit cent quinze (815); la valeur du champ "Octet count" est de six (6) et le champ "Value" est montré au Tableau 228.

Tableau 228 – Champ "Value" de demande "Add device list entry"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Device code	Code de liste de dispositif tel que spécifié au Tableau F.30
1	Unique ID	Unique ID	L'identificateur unique attribué au dispositif tel que défini en 3.3.91 et décrit en 6.2.1.2.4

8.2.4.47.2 Primitive Response

Le champ "Value" de la réponse est montré au Tableau 229. Les valeurs des "Response code" (codes de réponse) sont montrées au Tableau 230.

Tableau 229 – Champ "Value" de réponse de "Add device list entry"

Décalage d'octet	Type de donnée	Nom de paramètre	Description
0	Enumeration	Device code	Code de liste de dispositif tel que spécifié au Tableau F.30
1	Unique ID	Unique ID	L'identificateur unique attribué au dispositif tel que défini en 3.3.91 et décrit en 6.2.1.2.4
6	Unsigned16	Device count	Nombre total d'entrées restantes du Tableau Device

Tableau 230 – Codes de réponse spécifiques à la commande "Add device list entry"

Valeur	Classe	Description
0	Success	Aucune erreur
2	Error	Sélection non valide, c'est-à-dire code de liste de dispositifs non pris en charge
5	Error	Trop peu d'octets de données reçus
8	Warning	Dispositif déjà dans la liste
16	Error	Accès limité
65	Error	Plus d'autres entrées disponibles
66	Error	Conflit de listes de dispositifs

8.2.4.47.3 Procédure au niveau du dispositif répondeur

Cette commande ajoute un dispositif dans le Tableau Device pour le type de liste spécifié par "Device code". Le Tableau Device pour 'Active list' ne peut pas être modifié par cette commande. Si la demande tente d'ajouter à la liste noire ('Black list') un dispositif qui est dans la liste active ('Active list'), le dispositif doit rejeter la demande et répondre par "Device list conflict" (c'est-à-dire conflit de listes de dispositifs). Cette commande peut être utilisée pour déplacer un dispositif entre la 'White list' et la 'Black list'.

8.2.4.48 Delete device list entry FAL PDU**8.2.4.48.1 Primitive Request**

La valeur de la Commande est de huit cent seize (816); la valeur du champ "Octet count" est de six (6) et le champ "Value" est montré au Tableau 231.