IEC 61158-4-7

Edition 1.0 2007-12

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Industrial communication networks – Fieldbus specifications – Part 4-7: Data-link layer protocol specification – Type 7 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-7: Spécification de protocole de la couche de liaison de données – Éléments de Type 7

IEC 61158-4-7:2007-12(EN-FR)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
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.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in 14 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
More than 55 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

A propos de l'IEC
La Commission Electrotechnique Internationale (IEC) 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 IEC
Le contenu technique des publications IEC est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Catalogue IEC - webstore.iec.ch/catalogue
Application autonome pour consulter tous les renseignements bibliographiques sur les Normes internationales, Spécifications techniques, Rapports techniques et autres documents de l'IEC. Disponible pour PC, Mac OS, tablettes Android et iPad.

Recherche de publications IEC - www.iec.ch/searchpub
La recherche avancée permet de trouver des publications IEC en utilisant différents critères (numéro de référence, texte, comité d'études,…). Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

IEC Just Published - webstore.iec.ch/justpublished
Restez informé sur les nouvelles publications IEC. Just Published détaille les nouvelles publications parues. Disponible en ligne et aussi une fois par mois par email.

Electropedia - www.electropedia.org
Le premier dictionnaire en ligne de termes électroniques et électriques. Il contient plus de 30 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans 14 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary
Plus de 55 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

Service Clients - webstore.iec.ch/csc
Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: csc@iec.ch.

IEC 61158-4-7

Edition 1.0 2007-12

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Industrial communication networks – Fieldbus specifications – Part 4-7: Data-link layer protocol specification – Type 7 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-7: Spécification de protocole de la couche de liaison de données – Éléments de Type 7

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

PRICE CODE
CODE PRIX XE

ICS 25.040.40; 35.100.20

ISBN 978-2-8322-1020-8

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 4-7: Data-link layer protocol specification – Type 7 elements

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 provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
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.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

NOTE Use of some of the associated protocol types is restricted by their intellectual-property-right holders. In all cases, the commitment to limited release of intellectual-property-rights made by the holders of those rights permits a particular data-link layer protocol type to be used with physical layer and application layer protocols in Type combinations as specified explicitly in the IEC 61784 series. Use of the various protocol types in other combinations may require permission from their respective intellectual-property-right holders.

International Standard IEC 61158-4-7 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This first edition and its companion parts of the IEC 61158-4 subseries cancel and replace IEC 61158-4:2003. This edition of this part constitutes an editorial revision.

This edition of IEC 61158-4 includes the following significant changes from the previous edition:

a) deletion of the former Type 6 fieldbus, and the placeholder for a Type 5 fieldbus data link layer, for lack of market relevance;
b) addition of new types of fieldbuses;
c) division of this part into multiple parts numbered -4-1, -4-2, ..., -4-19.

This bilingual version (2013-09) corresponds to the monolingual English version, published in 2007-12.

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

FDIS	Report on voting
65C/474/FDIS	65C/485/RVD

Full information on the voting for the approval of this 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 maintenance result 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.

NOTE The revision of this standard will be synchronized with the other parts of the IEC 61158 series.

The list of all the parts of the IEC 61158 series, under the general title *Industrial communication networks – Fieldbus specifications*, can be found on the IEC web site.

The contents of the corrigendum of January 2014 have been included in this copy.

INTRODUCTION

This part of IEC 61158 is one of a series produced to facilitate the interconnection of automation system components. It is related to other standards in the set as defined by the "three-layer" fieldbus reference model described in IEC/TR 61158-1.

The data-link protocol provides the data-link service by making use of the services available from the physical layer. The primary aim of this standard is to provide a set of rules for communication expressed in terms of the procedures to be carried out by peer data-link entities (DLEs) at the time of communication. These rules for communication are intended to provide a sound basis for development in order to serve a variety of purposes:

a) as a guide for implementors and designers;

b) for use in the testing and procurement of equipment;

c) as part of an agreement for the admittance of systems into the open systems environment;

d) as a refinement to the understanding of time-critical communications within OSI.

This standard is concerned, in particular, with the communication and interworking of sensors, effectors and other automation devices. By using this standard together with other standards positioned within the OSI or fieldbus reference models, otherwise incompatible systems may work together in any combination.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 4-7: Data-link layer protocol specification – Type 7 elements

1 Scope

1.1 General

The data-link layer provides basic time-critical messaging communications between devices in an automation environment.

This protocol provides communication opportunities to all participating data-link entities

a) in a synchronously-starting cyclic manner, according to a pre-established schedule, and

b) in a cyclic or acyclic asynchronous manner, as requested each cycle by each of those data-link entities.

Thus this protocol can be characterized as one which provides cyclic and acyclic access asynchronously but with a synchronous restart of each cycle.

1.2 Specifications

This standard specifies

a) procedures for the timely transfer of data and control information from one data-link user entity to a peer user entity, and among the data-link entities forming the distributed data-link service provider;

b) the structure of the fieldbus DLPDUs used for the transfer of data and control information by the protocol of this standard, and their representation as physical interface data units.

1.3 Procedures

The procedures are defined in terms of

a) the interactions between peer DL-entities (DLEs) through the exchange of fieldbus DLPDUs;

b) the interactions between a DL-service (DLS) provider and a DLS-user in the same system through the exchange of DLS primitives;

c) the interactions between a DLS-provider and a Ph-service provider in the same system through the exchange of Ph-service primitives.

1.4 Applicability

These procedures are applicable to instances of communication between systems which support time-critical communications services within the data-link layer of the OSI or fieldbus reference models, and which require the ability to interconnect in an open systems interconnection environment.

Profiles provide a simple multi-attribute means of summarizing an implementation's capabilities, and thus its applicability to various time-critical communications needs.

1.5 Conformance

This standard also specifies conformance requirements for systems implementing these procedures. This part of this standard does not contain tests to demonstrate compliance with such requirements.

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 61158-2 (Ed.4.0), *Industrial communication networks – Fieldbus specifications – Part 2: Physical layer specification and service definition*

IEC 61158-3-7, *Industrial communication networks – Fieldbus specifications – Part 3-7: Data link service definition – Type 7 elements*

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

ISO/IEC 7498-3, *Information technology – Open Systems Interconnection – Basic Reference Model: Naming and addressing*

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

3 Terms, definitions, symbols and abbreviations

For the purposes of this document, the following terms, definitions, symbols and abbreviations 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.

3.1.1 correspondent (N)-entities [7498-1]
correspondent DL-entities (N=2)
correspondent Ph-entities (N=1)

3.1.2 DL-address [7498-3]

3.1.3 DL-connection [7498-1]

3.1.4 DL-connection-end-point [7498-1]

3.1.5 DL-connection-end-point-identifier [7498-1]

3.1.6 DL-name [7498-3]

3.1.7 DL-protocol [7498-1]

3.1.8 DL-protocol-connection-identifier [7498-1]

3.1.9 DL-protocol-control-information [7498-1]

3.1.10 DL-protocol-data-unit [7498-1]

3.1.11 DL-relay [7498-1]

3.1.12 DL-service-connection-identifier [7498-1]

3.1.13 DL-service-data-unit [7498-1]

3.1.14 DL-user-data [7498-1]

3.1.15 flow control [7498-1]

3.1.16 (N)-entity [7498-1]
DL-entity
Ph-entity

3.1.17 (N)-interface-data-unit [7498-1]
DL-service-data-unit (N=2)
Ph-interface-data-unit (N=1)

3.1.18 (N)-layer [7498-1]
DL-layer (N=2)
Ph-layer (N=1)

3.1.19 (N)-service [7498-1]
DL-service (N=2)
Ph-service (N=1)

3.1.20 (N)-service-access-point [7498-1]
DL-service-access-point (N=2)
Ph-service-access-point (N=1)

3.1.21 (N)-service-access-point-address [7498-1]
DL-service-access-point-address (N=2)
Ph-service-access-point-address (N=1)

3.1.22 peer-entities [7498-1]

3.1.23 Ph-interface-control-information [7498-1]

3.1.24 Ph-interface-data [7498-1]

3.1.25 primitive name [7498-3]

3.1.26 reset [7498-1]

3.1.27 responding-DL-address [7498-3]

3.1.28 routing [7498-1]

3.1.29 segmenting [7498-1]

3.1.30 sequencing [7498-1]

3.1.31 system management [7498-1]
systems-management

3.2 Service convention terms and definitions

This standard also makes use of the following terms defined in ISO/IEC 10731 as they apply to the data-link layer:

3.2.1 confirm (primitive);
requestor.deliver (primitive)

3.2.2 deliver (primitive)

3.2.3 DL-service-primitive;
primitive

3.2.4 DL-service-provider

3.2.5 DL-service-user

3.2.6 DL-user-optional-facility

3.2.7 indication (primitive)
acceptor.deliver (primitive)

3.2.8 multi-peer

3.2.9 request (primitive);
requestor.submit (primitive)

3.2.10 requestor

3.2.11 response (primitive);
acceptor.submit (primitive)

3.2.12 submit (primitive)

3.3 Other terms and definitions

NOTE Many definitions are common to more than one protocol Type; they are not necessarily used by all protocol Types.

For the purpose of this part of IEC 61158, the following definitions also apply:

3.3.1
acknowledgement response DLPDU
information that the recipient of an acknowledged message emits in order to signal either the proper reception of the message or the lack of available resources to store the message, received by the DLE on the local link that emitted the message which requested the acknowledgement

3.3.2
basic cycle
sequence of scanning by the bus-arbitrator of:

a) a set of DLCEP-identifiers for variables, requests, and cyclical application messages,

b) plus the window provided for aperiodic exchanges,

c) plus the window provided for message services,

d) plus the window provided for synchronization

3.3.3
basic transaction
succession of DLPDUs related to a single DL-service instance

3.3.4
bus-arbitrator (BA)
DLE that controls each data producer's right to access the medium

NOTE At any given instant one and only one bus-arbitrator is active in each DL-segment of a DL-subnetwork.

3.3.5
consumed identified variable
identified variable that corresponds to a DLCEP-identifier for which the entity in question receives data

3.3.6
control field
portion of an emitted or received DLPDU that gives the nature of the data exchanged and the type of exchange

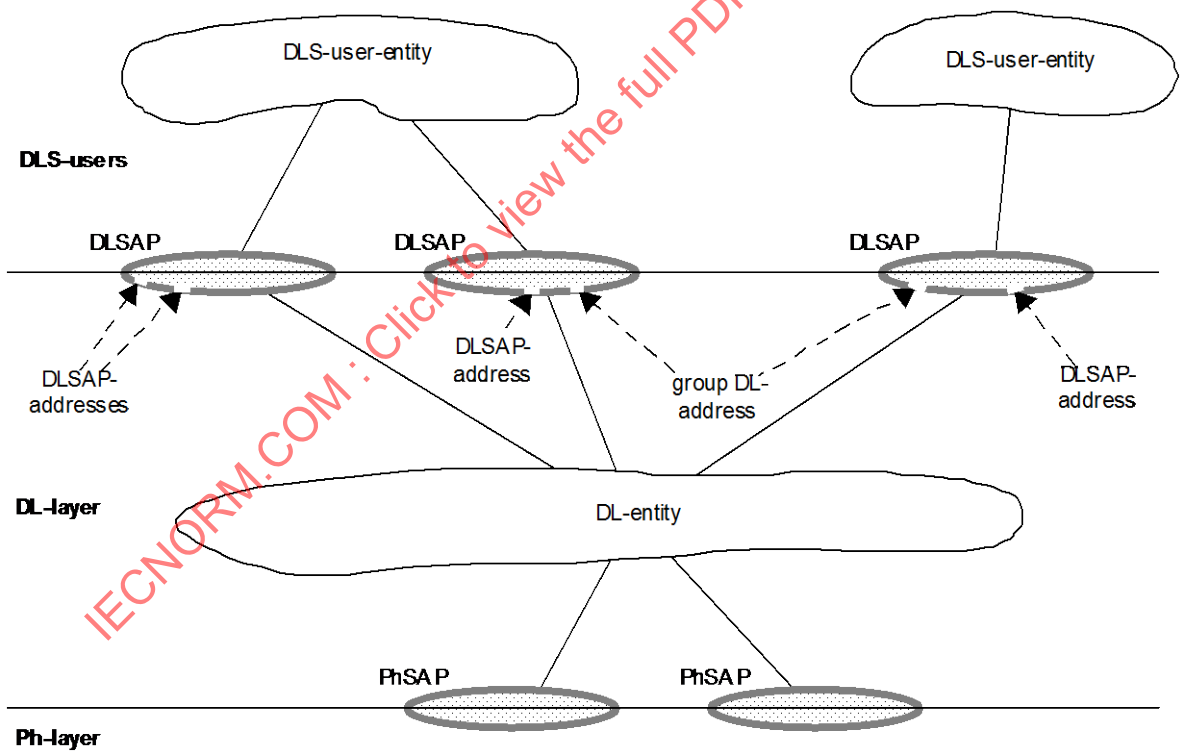
3.3.7
destination address
three octets specifying the DL-segment of the DLE to whom the message is sent, and the destination DLSAP's sub-address within the local link DL-segment

3.3.8
DLCEP-address
information that the bus-arbitrator emits to allocate the medium to a data producer for the purpose of exchanging a variable

3.3.9
DL-segment, link, local link
single DL-subnetwork in which any of the connected DLEs may communicate directly, without any intervening DL-relaying, whenever all of those DLEs that are participating in an instance of communication are simultaneously attentive to the DL-subnetwork during the period(s) of attempted communication

3.3.10
DLSAP
distinctive point at which DL-services are provided by a single DL-entity to a single higher-layer entity.

NOTE This definition, derived from ISO/IEC 7498-1, is repeated here to facilitate understanding of the critical distinction between DLSAPs and their DL-addresses. (See Figure 1.)



NOTE 1 DLSAPs and PhSAPs are depicted as ovals spanning the boundary between two adjacent layers.

NOTE 2 DL-addresses are depicted as designating small gaps (points of access) in the DLL portion of a DLSAP.

NOTE 3 A single DL-entity may have multiple DLSAP-addresses and group DL-addresses associated with a single DLSAP.

Figure 1 – Relationships of DLSAPs, DLSAP-addresses and group DL-addresses

3.3.11
DL(SAP)-address
either an individual DLSAP-address, designating a single DLSAP of a single DLS-user, or a group DL-address potentially designating multiple DLSAPs, each of a single DLS-user.

NOTE This terminology is chosen because ISO/IEC 7498-3 does not permit the use of the term DLSAP-address to designate more than a single DLSAP at a single DLS-user

3.3.12
(individual) DLSAP-address
DL-address that designates only one DLSAP within the extended link

NOTE A single DL-entity may have multiple DLSAP-addresses associated with a single DLSAP.

3.3.13
end of message transaction indication DLPDU
information that the source entity of a message emits in order to return link access control to the bus-arbitrator at the end of a message transaction

3.3.14
extended link
DL-subnetwork, consisting of the maximal set of links interconnected by DL-relays, sharing a single DL-name (DL-address) space, in which any of the connected DL-entities may communicate, one with another, either directly or with the assistance of one or more of those intervening DL-relay entities

NOTE An extended link may be composed of just a single link.

3.3.15
frame
denigrated synonym for DLPDU

3.3.16
group DL-address
DL-address that potentially designates more than one DLSAP within the extended link. A single DL-entity may have multiple group DL-addresses associated with a single DLSAP. A single DL-entity also may have a single group DL-address associated with more than one DLSAP

3.3.17
identified variable (or simply "variable")
DLL variable (buffer) for which an associated DLCEP-identifier has been defined

3.3.18
identifier
16-bit word associated with a system variable. A DLCEP-identifier uniquely designates a single variable within the DL-subnetwork

3.3.19
invalid DLCEP-identifier
identifier not recognized locally

3.3.20
local link
set of devices that respect the DL-protocol and that are interconnected through a medium . Only one bus-arbitrator is active on a single local link

3.3.21
macrocycle
set of basic cycles needed for all cyclical DLCEP-identifiers to be scanned

3.3.22
message DL-address
information that the bus-arbitrator emits to allocate the medium to a source entity for a message transfer

3.3.23
message response DLPDU
information that a data producer emits in response to a message DLCEP-identifier DLPDU

NOTE The desired destination entity or entities pick up this information.

3.3.24
node
single DL-entity as it appears on one local link

3.3.25
periodic scanning of variables
action by the bus-arbitrator that guarantees the cyclical exchange of variables

NOTE This is the basic principle of the Type 7 DL-service and protocol.

3.3.26
produced identified variable
identified variable that corresponds to a DLCEP-identifier for which the DLE emits data

3.3.27
receiving DLS-user
DL-service user that acts as a recipient of DL-user-data

NOTE A DL-service user can be concurrently both a sending and receiving DLS-user.

3.3.28
request DLCEP-address
information that the bus-arbitrator emits to allocate the medium to the initiator of an explicit request for a variable exchange

3.3.29
request response DLPDU
information that the initiator of an explicit request for a variable exchange emits in response to a request DLCEP-identifier DLPDU, and to whose transmittal the bus-arbitrator also responds

3.3.30
sending DLS-user
DL-service user that acts as a source of DL-user-data

3.3.31
source address
24-bit word including the DL-segment number of the entity sending the message, and the entity's sub-address within the DL-segment

3.3.32
timers
see 5.6

3.3.33
triggered message scanning
function of a bus-arbitrator that makes it possible to transfer messages

3.3.34
triggered scanning of variables
function of a bus-arbitrator that makes possible the non-cyclical exchange of variables

3.3.35
triggered periodic scanning of messages
function of a bus-arbitrator that makes it possible to request triggered message exchanges cyclically

3.3.36
triggered periodic scanning of variables
function of a bus-arbitrator that makes it possible to request triggered variable transfers cyclically

3.3.37
turnaround time
time interval between reception or emission of the last MAC symbol of a DLPDU, signaled by a SILENCE indication from the PhL, and the reception or emission of the first MAC symbol of the subsequent DLPDU, signaled by an ACTIVITY indication from the PhL, both as measured in a given station

3.3.38
variable response DLPDU
information that a data producer emits in response to a DLCEP-identifier DLPDU, which also alerts data consumers to the relevance of the immediately time-proximate DLPDU

3.4 Symbols and abbreviations

3.4.1 BA	bus-arbitrator
3.4.2 B_Dat_Cons	buffer which contains the value of the data consumed
3.4.3 B_Dat_Prod	buffer which contains the value of the data produced
3.4.4 B_Req1/2	buffer containing the list of DLCEP-identifiers that are the objet of a specified explicit request for a transfer at the priority 1 (urgent) or 2 (normal)
3.4.5 ID_DAT	DLPDU used to allocate the medium to a buffer transfer
3.4.6 ID_MSG	DLPDU used to allocate the medium to a message exchange
3.4.7 ID_RQ1/2	DLPDU used to allocate the medium to a request for a buffer transfer
3.4.8 PRT	physical reaction time
3.4.9 Q_IDMSG	queue of requested DLCEP-identifiers received by the BA for message transfer
3.4.10 Q_IDRQ1/2	queue of requested DLCEP-identifiers received by the BA at the priority 1 (urgent) or 2 (normal)
3.4.11 Q_Msg_Cyc	queue which contains messages to be emitted that are associated with cyclical exchanges
3.4.12 Q_Msg_Aper	queue which contains messages to be emitted that are associated with aperiodic exchanges
3.4.13 Q_Req1/2	queue containing the list of DLCEP-identifiers that are the objet of a free explicit request for a transfer at the priority 1 (urgent) or 2 (normal)
3.4.14 Q_RPRQ	queue for aperiodic transfers in progress
3.4.15 RQ_Inhibit	Indicator used to manage explicit request for variable

exchanges

3.4.16 RP_ACK DLPDU used to transfer an acknowledgement of message exchange, Transfer OK

3.4.17 RP_DAT DLPDU used to carry the value of the identified variable previously requested.

3.4.18 RP_DAT_MSG DLPDU used to carry the value of the identified variable previously requested with a request for a message exchange.

3.4.19 RP_DAT_RQ1/2 DLPDU used to carry the value of the identified variable previously requested with a request for an explicit transfer of variables.

3.4.20 RP_DAT_RQ1/2_MSG DLPDU used to carry the value of the identified variable previously requested with a request for an explicit transfer of variables and a request for a message transfer.

3.4.21 RP_MSG_ACK DLPDU used to carry the message to be exchanged with a request of acknowledgement of this exchange.

3.4.22 RP_MSG_NOACK DLPDU used to carry the message to be exchanged without a request of acknowledgement of this exchange.

3.4.23 RP_NAK DLPDU used to transfer an acknowledgement of message exchange, that the message was received correctly, but was not stored by the DLE.

3.4.24 RP_END DLPDU used to terminate a message exchange.

3.4.25 STT station turnaround time

3.4.26 TI latency time

4 Overview of the DL-protocol

4.1 Overall description of medium allocation

An element known as the **bus-arbitrator (BA)** controls the right of each data producer to access the medium by emitting a DLPDU containing a DLCEP-identifier. At any given instant there should be only one active bus-arbitrator in each local link.

NOTE The term "data producer" designates the sole station connected to the local link that is recognized as having the responsibility of emitting the data associated with the identifier DLPDU circulating on the medium.

Each transaction belongs to one of the three medium allocation classes defined below:

— cyclical exchange of variables, requests, or messages,

— explicit request for variable exchange,

— explicit request for message transfer.

For cyclical variable exchanges a basic transaction is made up of the following phases. The bus-arbitrator broadcasts a variable identifier DLPDU. The sole producer of the information required then broadcasts a variable response DLPDU. During this phase consumers take the information from the local link. Figure 2 shows the various phases of a variable exchange transaction. When one transaction has been completed the bus-arbitrator begins the following transaction according to guidelines defined when the system is configured.

During a cyclical variable exchange the information producer can, using the response DLPDU, transmit to the BA an explicit request for the exchange of variables or messages.

For an explicit request for a variable exchange, a basic transaction is made up of the following phases: The bus-arbitrator broadcasts a request identifier DLPDU. Then the initiator of the request broadcasts a request response DLPDU. One or more transactions identical to the cyclical variables exchange transaction then follow.

For message transfers a basic transaction consists of the following phases: The bus-arbitrator broadcasts a message identifier DLPDU. Then the message response DLPDU is exchanged between the communicating entities. This DLPDU may or may not be followed by an acknowledgement DLPDU. The source entity then transmits an end of transaction indication DLPDU to the bus-arbitrator.

The time interval separating the reception or emission of the end of one DLPDU and the emission or reception of the following DLPDU is known as the station's turnaround time, whether the station's function be that of a producer/consumer or bus-arbitrator.

A more detailed definition of turnaround time is given in 3.3.37. The impact of turnaround time on data-link timers is described in 5.6.2.

The role of the bus-arbitrator is to "give the floor" to each data producer, taking into account the services required for the type of data (according to the three medium allocation classes defined above).

The bus-arbitrator thus has three types of functions:

— periodic scanning of variables or periodic triggering of variable and message exchanges,

— triggered scanning of variables,

— triggered scanning of messages.

In addition, the bus-arbitrator can provide a synchronization function in order to guarantee the constant length of scanning cycles.

Each type of scanning takes place in a specific window, that is, respectively in a periodic window, an aperiodic variables window, an aperiodic messages window, or a synchronization window, as shown in Figure 2. These four windows constitute a basic scanning cycle.

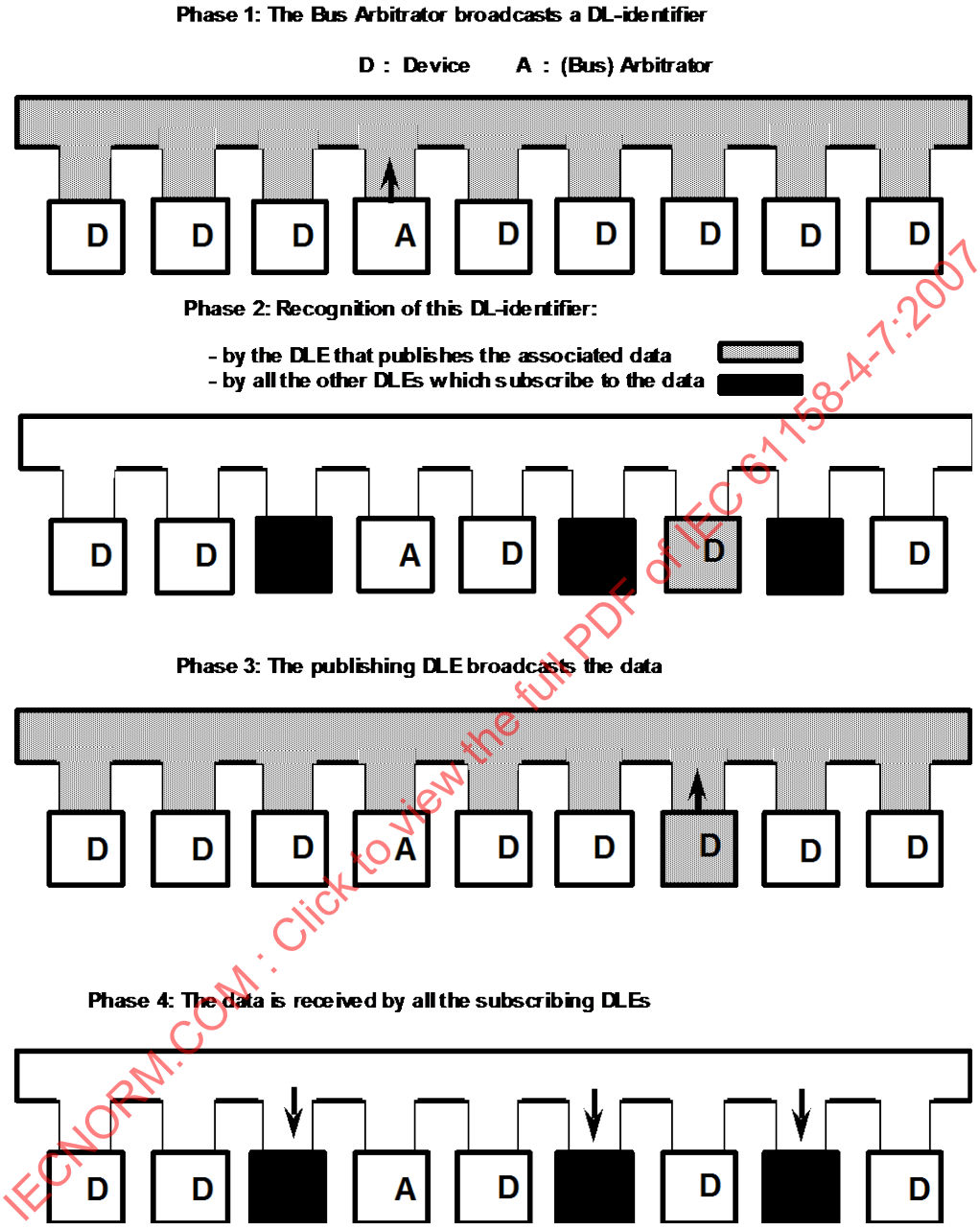


Figure 2 – General description of medium allocation

The medium access technique has the following characteristics:

— broadcasting of identified variables,

— increased efficiency in cyclical variable exchanges,

— parameters for medium sharing can be set by the user when the system is configured,

— guaranteed access time for cyclical variable exchanges, under all circumstances and regardless of the number of requests for triggered variable and/or message exchanges.

— possibility of triggering a transaction in accordance with a global clock, that is a clock that indicates the same time for all stations.

In addition, the medium access technique makes it possible to:

— give cyclical exchanges highest priority,

— respect the scanning period associated with each variable,

— give different priorities to triggered messages transfers and variable exchanges. These transactions are triggered in adjustable windows: the lengths of the "aperiodic variable" and "aperiodic message" windows are defined in terms of maximum limits set by the user when the system is configured.

— change the priority of aperiodic transactions by inserting them in the periodic window.

4.2 Types of entities

4.2.1 Producer/consumer entity

DLL services use various types of DLPDUs. Each DLPDU type is defined in the control field of the DLPDU. Interactions between a specific service and DLPDU types will be described in the detailed specifications of each service.

NOTE 5.5 describes all types of DLPDUs and also explains the use of various fields.

The functioning of an entity belonging to the highest conformance class requires:

— a mechanism for analyzing DLPDU type (use of the control field),

— a table of identifiers recognized upon emission,

— a table of identifiers recognized upon reception (both tables are defined at the interface between the DLL and system management),

a mechanism that provides read/write access to variables and messages.

The conceptual model of a data-link producer/consumer entity includes various buffers: queues needed to provide the services offered by the DLL, as shown in .Figure 3.

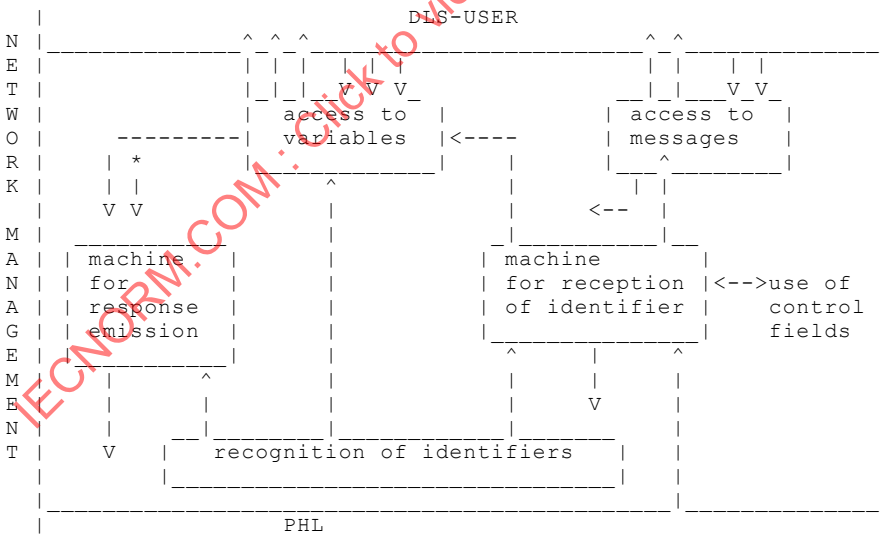


Figure 3 – Internal structure of a producer/consumer entity

The following are associated with each identifier produced:

— a buffer called B_DATprod, which contains the value of the variable produced,

— a buffer called B_REQ, containing the list of identifiers that are the object of an explicit request for a buffer transfer, if the identifier has been reserved for the explicit request for buffer transfer service,

— a queue called Q_MSGcyc, which contains messages to be emitted that are associated with cyclical message transfer, if the identifier has been reserved for cyclical message transfer,

— an indicator of the validity of the identifier produced (system management),
— for each of the B_DATprod and B_REQ buffers, an availability indicator,
— an indicator associated with Q_MSGcyc: queue filled or not,
— indicators used to manage explicit requests for variable exchange and message transfer requests:
— explicit request for buffer transfer in progress (RQ),
— priority associated with an explicit request (PR),
— scope of the explicit request (RQ_INHIBIT)
— message transfer request in progress (MSG) if the identifier has been reserved for aperiodic message transfer,
— reference to the queue of messages to be emitted associated with aperiodic message transfer.

The following are associated with each identifier consumed:

— a buffer called B_DATcons, which contains the value of the variable consumed,
— an indicator of the validity of the identifier consumed (system management),
— an indicator linked to the management of B_DATcons: availability of the buffer (access conflict),

NOTE The buffer availability indicator provides information concerning the integrity of data manipulated by service primitives.

Each entity that supports a message transfer request service also uses:

— a queue called Q_MSGaper, which is associated with aperiodic message transfer and contains messages to be emitted,
— a queue called Q_MSGrec that contains messages received.
— an indicator of queue status (full or not) is associated with the queue reserved for aperiodic message transfer.

In addition, each entity that supports acknowledged message service has the following characteristics:

— value of the source entity's even/odd bit,
— maximum value of the restart counter,
— current value of the restart counter.

The following are associated with the queue Q_MSGrec:

— an indicator of queue status (full or not),
— value of the destination entity's even/odd bit,
— value of the stored source address.

If the entity also supports the free explicit request for buffer transfer service, there are two global queues for holding free explicit requests for buffer transfer:

— Q_REQ1 is associated with urgent requests,

Q_REQ2 is associated with normal requests.

A status indicator (full or not) is associated with each queue.

Figure 4 shows many of these associated buffers and queues.

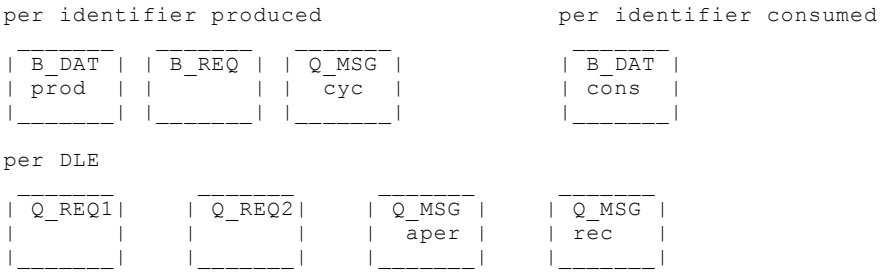


Figure 4 – Associated buffers and queues

Timers associated with the data-link protocol are also managed by the producer/consumer entity (see 5.6).

Annex B presents an object model that defines the attributes of a DLCEP-identifier.

4.2.2 Bus arbitrator entity

The function provided by the bus-arbitrator consists of "giving permission to speak" to each data producer. This permission is granted for three types of scanning:

— periodic or triggered periodic scanning of variables and messages,

— triggered scanning of variables,

— triggered scanning of messages.

The bus-arbitrator also provides the following functions:

— chaining of basic transactions,

— chaining of the various types of scanning,

— analysis of DLPDU control fields (the control field carries aperiodic variable and message requests),

— filling of aperiodic windows in accordance with these requests.

NOTE A basic transaction is made up of the succession of DLPDUs related to a single service.

In addition, the bus-arbitrator can provide a synchronization function to guarantee the constant length of a basic scanning cycle.

Each type of scanning takes place in a special window: respectively in a periodic window, an aperiodic variables window, an aperiodic messages window, or a synchronization window.

The four windows constitute a basic scanning cycle.

Figure 5 provides an overview of the bus-arbitrator structure. A more complete description of the bus-arbitrator is given in 7.4.

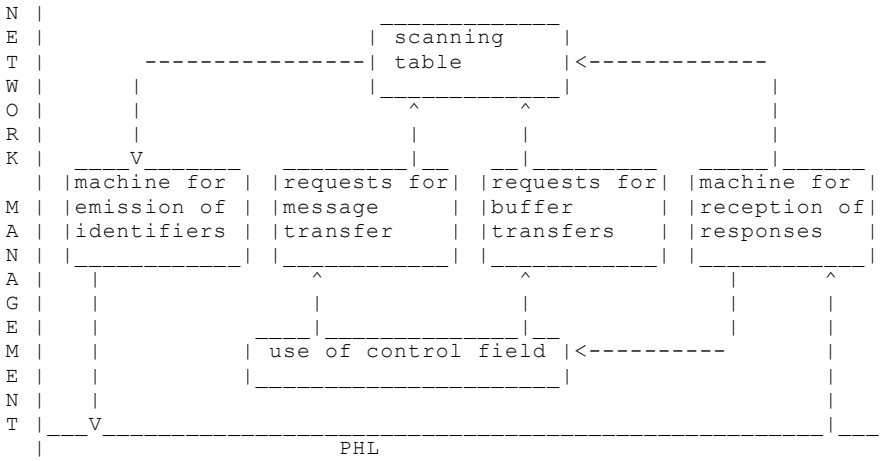


Figure 5 – Internal structure of a bus arbitrator

The conceptual model of the bus-arbitrator details the various tables and queues needed to provide the services offered by the DLL.

The bus-arbitrator manages according to system configuration, as shown in Figure 6:

— a scanning table with static portions and portions that are modified dynamically during scanning,

— a recovery buffer for the triggered periodic scanning of variables,

— a queue called Q_IDRQ1 for urgent explicit requests for buffer transfer,

— a queue called Q_IDRQ2 for normal explicit requests for buffer transfer,

— a queue for aperiodic transfers in progress (Q_RPRQ),

— a queue called Q_IDMSG for requests for message transfer,

— a basic padding sequence used to fill the synchronization window.

NOTE The management algorithm and/or calculation of these tables should provide equal access rights for the various entities, that is, starving certain entities should be avoided.

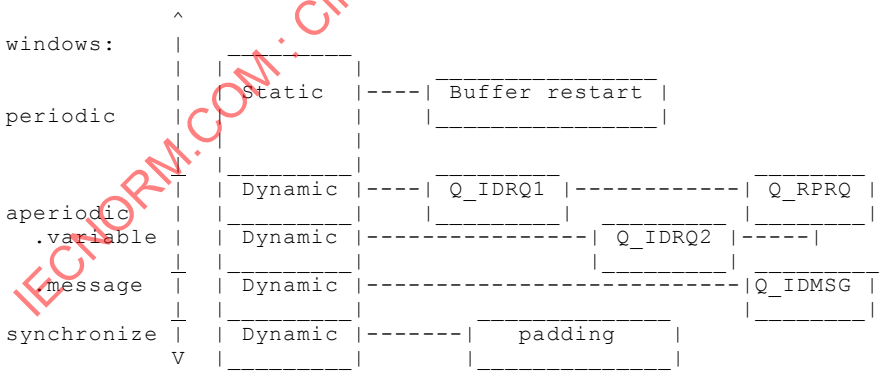


Figure 6 – Polling BA Table

4.3 Addressing

4.3.1 DLSAP-user relationship

The role of the DLL is to transmit information reliably and in accordance with a transmission protocol.

Within a single physical station several user entities have access to DLL services. Each of these entities uses a service access point (SAP) belonging to the DLL, in conformance with the ISO's static model in which each entity (N+1) is recognized by the address of the service access point (N) to which the entity is statically attached.

Several associations between user entities are possible on the level of access points to DLL services.

Thus, in accordance with the ISO model, a user entity (N+1) requests the establishment of a connection (N) in order to communicate with another user entity (N+1). The (N) entities, each of which is linked to one of the two (N+1) entities, create and manage this connection after negotiating it.

All exchanges between the entities (N+1) thus associated take place using this connection. The connection is identified locally at each SAP by a specific connection end point identifier (CEPi).

The number of DLSAPs corresponds to the number of entities in a station that are potential users of DLL services.

The number of CEPi per DLSAP corresponds to the number of potential communication links allocated to the (N+1) entity linked to the DLSAP.

4.3.2 OSI model relationship

This DLL follows the same rules as ISO/IEC 7498. Each potential user entity calls upon the services of the DLL through a DLSAP.

The DLL places two distinct addressing spaces as the disposal of the DLS-user:

— the addressing space concerning buffer transfer, each identifier being coded in 16 bits,

— the addressing space concerning message exchanges, which enables addressing messaging DLS-users, each address being coded in 24 bits.

The identifiers, encoded in 16 bits, allow addressing buffers within a local link, whereas the messaging addresses, encoded in 24 bits are meant to identify all the messaging DLS-users of the DL-subnetwork, that is those of all the DL-segments of the DL-subnetwork. Consequently, a communication system composed of n local links will have a single messaging address space but will have n identifier spaces to address the buffers.

Figure 7 illustrates this usage.

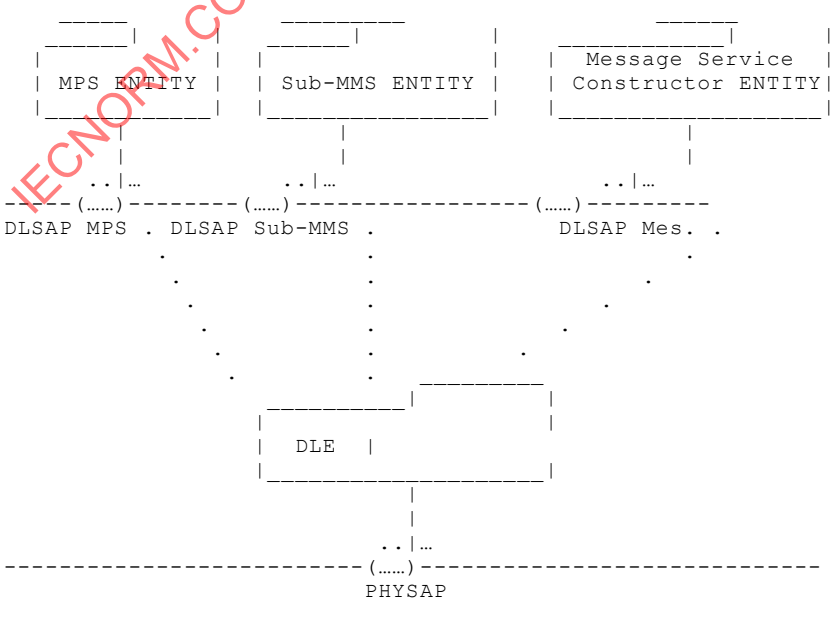


Figure 7 – Addressing scheme

4.3.2.1 Buffer transfer addressing

For reasons relating to performance and determinism, all associations between the various DLL user entities are known and defined when the system is configured.

Thus upon configuration the number of DLSAP is known, as well as the number of connections within the DLSAP.

This DLL uses broadcasting as its mode of transmission over the physical support. There is one producer and one or more consumers of each piece of information transmitted.

The connections within each DLSAP are multipoint connections providing unilateral communication (connections have more than two extremities, and over these connections data communication occurs in a single direction set in advance: from the producer to the consumers).

Connection end point identifiers (CEPi) that are known as identifiers and are encoded using 16 bits identify associations between user entities.

Thus CEPis are not recognized locally within a DLSAP, but globally throughout the DL-subnetwork, and the role of the DLSAP address is less important in the addressing model.

Thus the concept of an DLSAP is not the same as in the general ISO model. This DLL represents a group of between 1 and n identifiers employed by a user entity attached to the DLL. On the other hand, a DLSAP does not address the Application entity directly, but through the use of identifiers.

Each connection (identifier) can be used differently (according to the configuration and periodic or aperiodic services chosen) to communicate with other stations (variable transfers) or with the bus-arbitrator (requests for aperiodic transfers).

4.3.2.2 Message exchange addressing

4.3.2.2.1 Addressing requirements

The addressing of the messaging DLS-users must allow meeting the usual communication requirement such as indicated in the ISO and which are as follows:

— point-to-point communication between two DLS-users, whether or not located in the same local link,

— multipoint communication between an entity and all the entities of a group, the latter being located in the same equipment or the local link or the extended link,

— communication by distribution between a DLS-user and all the other DLS-users of a DLE, a local link or the extended link.

NOTE This corresponds to the reservation of a particular group containing all the entities of a device, a local link or the extended link. The addressing of the messaging DLS-users must distinguish between two types of addresses:

— the physical address, in order to identify an application entity with respect to its physical location, by means of the DL-segment number and the number of the station where it is located,

— the logic address, in order to identify an application entity whose physical location will not be required.

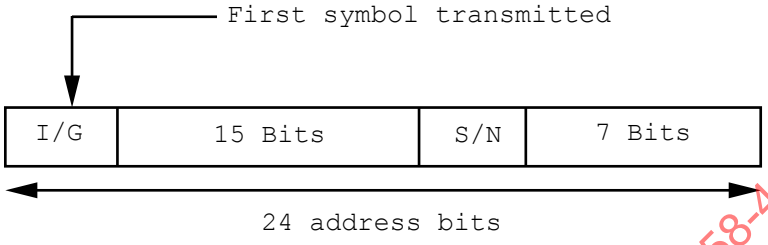
The notions of restricted groups on the DL-subnetwork with respect to groups of DLS-users in a device or in a DL-segment, have been defined with an aim to optimize the flow on the bus. Thus when addressing a group, it is thus necessary to have the possibility of a filter on the level of each bridge. The function of this filter is to prevent distribution on all the DL-segments of the extended link when this is not necessary.

The addressing of the buffers limits the number of devices on a DL-segment to 256. This must be taken into account in the messaging addressing mode.

4.3.2.2.2 Encoding of the addresses

The source and addressee DLS-users exchanging messages are identified in a DL-subnetwork with the help of addresses which are found in the link protocol data units (RP_MSG_NOACK and RP_MSG_ACK).

In order to meet the individual addressing requirements of the DLS-users in a device, in a group of devices of a local link or a group of devices of the extended link, the addressing space offered by the 24 bits is divided as shown in Figure 8.



I/G designates an individual DLS-user (I)or a group of DLS-users (G)

S/N designates a group of DLS-users on a DL-segment (S) or on the DL-subnetwork (N)

Figure 8 – Address partitioning

The encoding of the individual addresses and the groups of addresses is shown in Table 1.

Table 1 – Individual and group address encoding

I/G	S/N	Type of addressing
0	0	Individual address
1	0	Address of a group in a DL-segment
not significant	1	Address of a group in the DL-subnetwork

The link layer thus offers an addressing space which is divided into two sub-spaces:

— one containing link addresses meant to identify DLS-users individually,

— the other meant to identify DLS-user groups.

The group addresses are themselves divided into two sub-spaces:

— the sub-space defining addresses of restricted groups, thus enabling identification of the DLS-users which belong to the same local link,

— the other space defining the addresses of generalized groups, thus enabling identification of the application identities which all belong globally to the DL-subnetwork.

Each of these previously defined sub-spaces (individual addresses and addresses of restricted groups) are divided into:

— physical addresses,

— logical addresses.

4.3.2.2.2.1 Individual addressing

The individual addresses enabling a DLS-user to correspond with one and only one other DLS-user are structured as shown in Figure 9 and Figure 10:

a) Individual physical addresses:

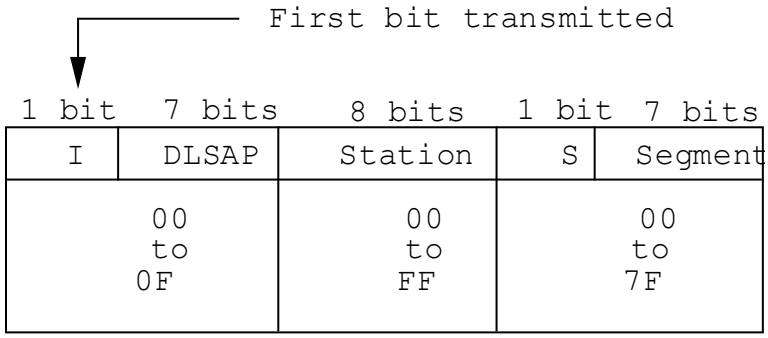


Figure 9 – Structure of an individual physical address

This structure allows identifying 16 physical DLSAPs per device.

b) Individual logical addresses:

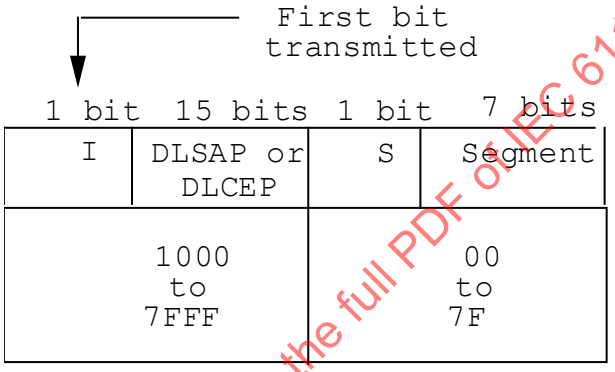


Figure 10 – Structure of an individual logical address

The addressing capacity is a total of 28K logical DLSAPs and DLCEPs per DL-segment.

4.3.2.2.2.2 Addressing of restricted groups

The restricted group addresses which enable a DLS-user to correspond with a group of DLS-users of the same device or of the same DL-segment are represented by their codes as shown in Figure 11 and Figure 12:

a) Addresses of restricted physical groups:

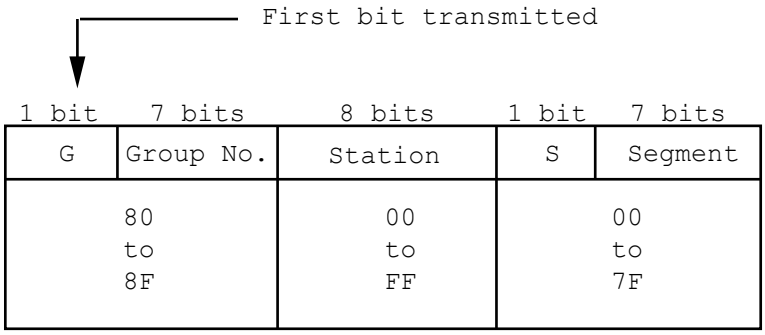


Figure 11 – Structure of restricted physical group address

The addressing capacity is then 16 numbers of physical restricted groups in a DLE. The distribution in a device is ensured by the specific group number, 8F.

b) Addresses of restricted logical groups:

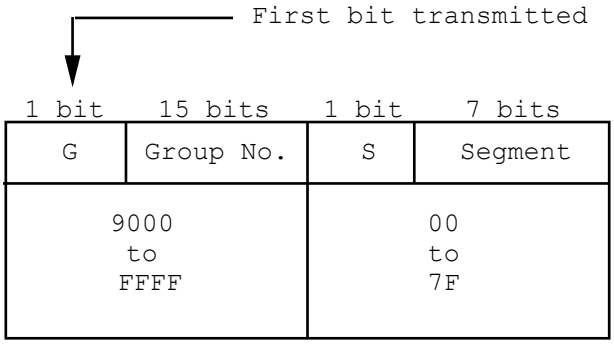


Figure 12 – Structure of a restricted logical group address

This structure allows addressing 28K numbers of restricted logical groups in a local link. FFFF corresponds to the distribution group in a DL-segment.

4.3.2.2.2.3 Addressing of generalized groups

The addresses of generalized groups allow a DLS-user to correspond with a group of DLS-users on the extended link. These addresses are divided into three sub-areas:

— an addressing sub-area which can be used by all the equipment of the extended link,

— a reserved sub-area,

— a vendor addressing sub-area.

The encoding structure of these addresses is as shown in Figure 13:

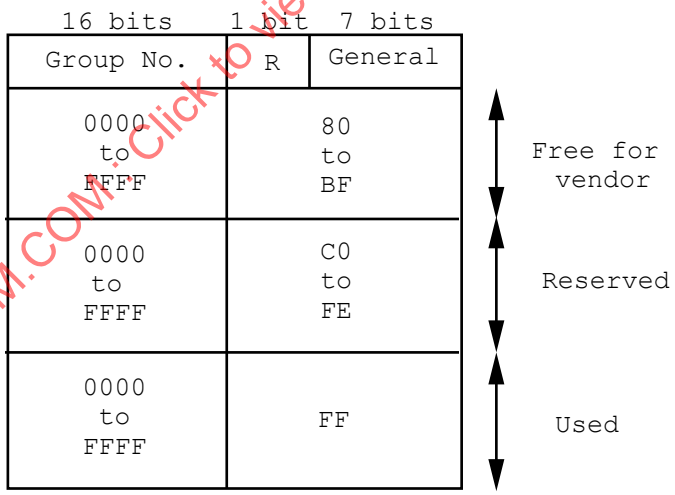


Figure 13 – Structure of a generalized group address

This addressing area allows identifying generalized 64K groups by sub-area element.

The distributionto the entire extended link is ensured by means of the particular group number FFFF.

4.3.2.3 Addressing capacity

The encoding rules offer the following possibilities for DL-subnetwork in terms of maximum capacity:

— 126 DL-segments in a DL-subnetwork;
— the "00" segment number is the DL-segment number by default;
— 256 DLEs per DL-segment in a DL-subnetwork;
— 16 physical DLSAPs in a DLE;
— 16 physical restricted group DL-addresses in a DLE;
— 28K logical DLSAPs in a local link;
— 28K logical restricted group DL-addresses in a local link;
— 64K generalized groups DL-addresses in the DL-subnetwork in the used sub-area.

The exact structure of the encoding of the beginning of an RP_MSG_XX message response DLPDU is as shown in Figure 14:

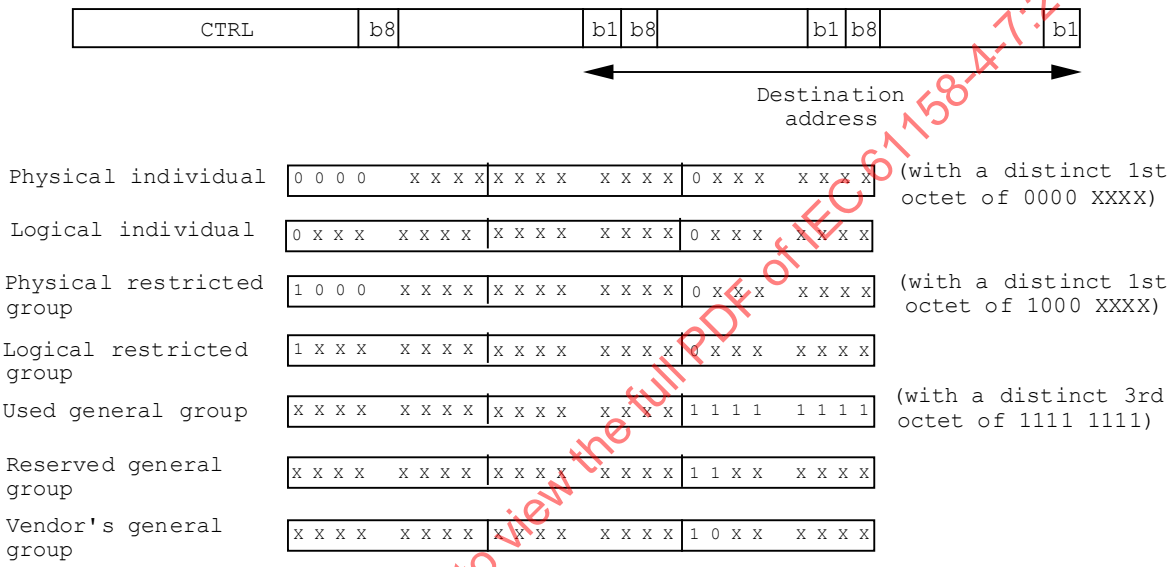


Figure 14 – Summary of address structure

The source address follows the destination address according to the same encoding.

4.4 Flow control

4.4.1 Variable exchanges

For the cyclical transfer of identified variables the principle of periodic scanning implies the replacement of an old variable value with a new variable independent of any reading access by the data consumer.

Management of flow control for these exchanges is definitively settled upon configuration:

— the bus-arbitrator manages flow control by respecting the scanning period associated with the variable,
— stations manage flow control by associating a buffer with each identified variable.

This principle also applies to triggered variable transfers since the user is only interested in the most recent validated value, and since the user associates a buffer with each identified variable. In addition, upon configuration of a DLL system a time delimiter is defined for the aperiodic window for triggered variable transfers for each of the bus-arbitrator's basic scanning cycles.

4.4.2 Message transfers

Flow control for message transfers is handled by the acknowledgement mechanism and by the queues for messages to be emitted and messages received.

When a DLL system is configured a time delimiter is defined for the aperiodic window for triggered message transfers for each of the bus-arbitrator's basic scanning cycles.

For acknowledged message transfers a portion of flow control is provided upon emission since the DLL only accepts requests for acknowledged message transfers if there is available space in the queue for messages to be emitted that is specified in the request. With this type of transfer there is also flow control upon reception by the queue of messages received and through the transmission of an acknowledgement DLPDU.

4.4.3 Detection of DLPDU duplication/loss

Detection mechanisms provided apply to errors caused by communication problems (cut-off DLPDUs, FCS error) or by out-of-service stations.

These errors include:

— loss of a DLCEP-identifier DLPDU,

— lack of a response DLPDU (DLPDU lost or station absent),

— loss of a request response DLPDU,

— loss of a message response DLPDU,

— loss of a message acknowledgement DLPDU,

— loss of a DLPDU indicating the end of a message transaction.

and can cause either the loss or the duplication of data.

DLPDU losses are taken into account in the graphs of final state machines defining data-link protocols.

Loss or duplication of DLPDUs can have the following impact on services:

— loss of a DLCEP-identifier DLPDU or a response concerning periodically exchanged variables is not usually detrimental since the value of the variable will be broadcast again during the next local link cycle;

— loss of an unacknowledged message DLPDU or of a DLPDU associated with a triggered variable exchange will only be handled by the upper layers;

— loss of an acknowledgement DLPDU results in a negative confirmation being sent to the initiating DLS-user;

— the concept of DLPDU duplication does not apply to periodically exchanged variables since the principle of cyclical variable refreshing implies that the same variable will be sent repeatedly;

— acknowledged message DLPDUs are protected from duplication by an even/odd numbering mechanism for messages;

— the duplication of a DLPDU associated with an explicit request for a buffer transfer will result in an additional overriding of the variable. This duplication is not detrimental since the concept of exchanging variables cyclically or upon explicit request is based on an asynchronous refreshing that is handled by the local link, and that the consumer cannot control. The semantics of data transmitted must be compatible with this working principle, that is, the variables exchange service is designed for the transmission of statuses, and not for the transmission of sequences of events.

4.5 Graphical representation

To be helpful for the reader some mechanisms are explained by a graphical representation. These representations are based on evaluation network built on an oriented graph with three types of nodes:

— the states, represented by a circle,

— the requests represented by a rectangle,

— the transitions, represented by a horizontal line.

The evaluation network is built according to the following principles, as illustrated in Figure 15:

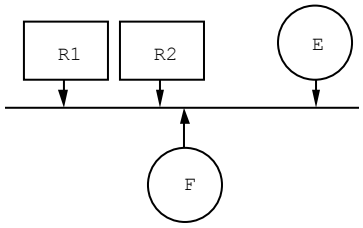


Figure 15 – Example of an evaluation net

When the described system is in state E, the arrival of request R1 or R2 results in a transition which switches it state F.

On the other hand, crossing a transition takes place, by convention, in zero time.

4.5.1 Simple actions

Simple actions can be associated with each transition, corresponding either to the transmission of requests or to the execution of local actions.

They are materially represented by a triangle, labeled with the name of the request or of the description of the executed action.

4.5.2 Conditions

The crossing of a transition can concern a condition. In this case, the oriented graph associated with the representation, shows a condition associated with the arc which precedes the transition.

A corollary of this representation allows defining choices for crossing a transition from the same initial state.

In the same way as for a simple transition, the transmission of an action can be associated with the crossing of a conditional transition.

5 General structure and encoding of PhIDUs and DLPDUs and related elements of procedure

5.1 DLPDU formats and components

This subclause describes the structure of DLPDUs and the use of the various control fields within the DLPDU. The term "DLPDU" refers to the protocol data units exchanged by data-link entities.

The basic structure of a DLPDU is as shown in Figure 16. The transmission and reception order is shown in Figure 17:

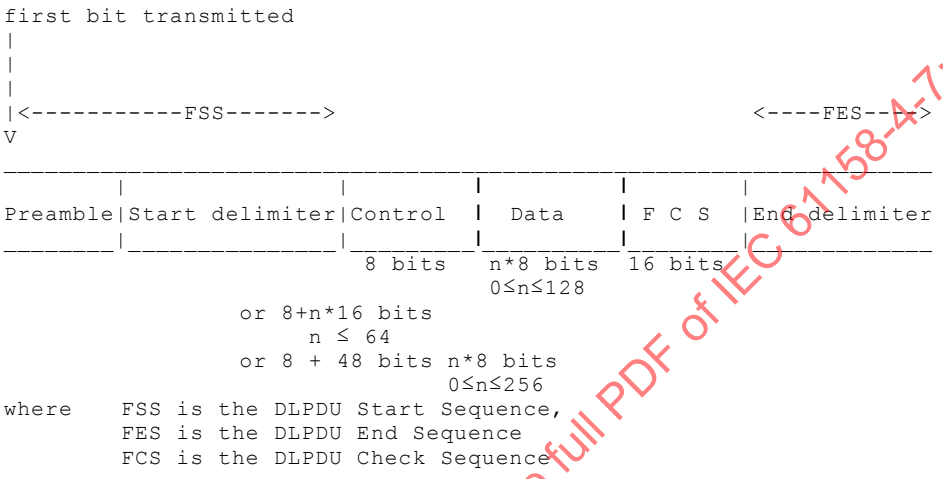


Figure 16 – Basic DLPDU structure

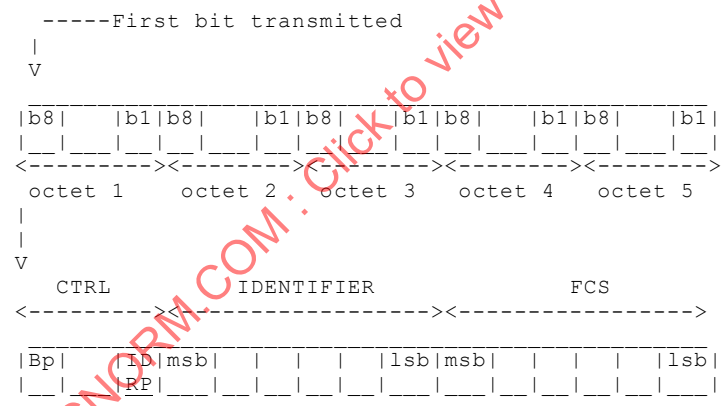


Figure 17 – DLPDU transmission / reception order

Octets are transmitted in the same order in which they are received (from the DLS-user or the PhL).

5.2 Description of each DLPDU component

5.2.1 Preamble

(See IEC 61158-2)

5.2.2 DLPDU delimiters

(See IEC 61158-2)

5.2.3 Control and addressing field

The control and addressing field specifies the type of DLPDU being exchanged:

— bit 1 of the control field indicates whether the DLPDU is a DLCEP-identifier DLPDU (bit 1 = 1) or a response DLPDU (bit 1 = 0)

— bits 2 through 7 of the control field, where each bit has a specific function, as shown in Table 2:

  – if bit 2 is set at 1 the DLPDU is related to a buffer transfer

  – if bit 3 is set at 1 the DLPDU is related to a message transfer

  – if bit 4 is set at 1 the DLPDU is related to an explicit request for a buffer transfer

  – if bit 5 is set at 1 the DLPDU is related to an acknowledgement

  – bit 6 indicates priority or the result of the acknowledgement

  – bit 7 set at 1 is used only to indicate an end of message transaction DLPDU;

— bit 8 of the control field (the first bit transmitted) is used only for a message response DLPDU or an acknowledgement response DLPDU. Bit 8 provides for even/odd numbering of messages to support duplicate message detection.

For a DLCEP-identifier DLPDU these bits specify the type of exchange in progress. For a response DLPDU these bits indicate the type of exchange in progress as well as requests for message transfer and explicit requests for variable exchanges, with their priorities.

Table 2 – DLPDU control-field coding

Bits 2 3 4 5 6 7	Identifier DLPDU
1 0 0 0 0 x	allocation for identified variable
0 1 0 0 0 x	allocation for message
0 0 1 0 1 x	allocation for urgent explicit request
0 0 1 0 0 x	allocation for normal explicit request
	Response DLPDU
1 0 0 0 0 x	identified variable response
1 1 0 0 0 x	identified variable response + message request
1 0 1 0 1 x	identified variable response + explicit urgent request
1 0 1 0 0 x	identified variable response + explicit normal request
1 1 1 0 1 x	identified variable response + explicit urgent request + message request
1 1 1 0 0 x	identified variable response + explicit normal request + message request
0 1 0 1 0 x	acknowledged message
0 1 0 0 0 x	unacknowledged message
0 0 0 1 1 x	positive acknowledgement
0 0 0 1 0 x	negative acknowledgement
0 0 1 0 1 x	list of identifiers for an urgent request
0 0 1 0 0 x	list of identifiers for a normal request
0 0 0 0 1 x	end of message transaction

For identifier DLPDUs the control and addressing field is extended to 8 + 16 bits. In this case it includes a single DL-identifier that represents a local-link DL-address encoded in 16 bits.

For request response DLPDUs the control and addressing field is extended to 8 + × times 16 bits. In this case it includes the list of DL-identifiers associated with the variables requested.

For message response DLPDUs the control and addressing field is extended to 8 + 24 + 24 bits. In this case it includes two extended-link DL-addresses — the destination and source, respectively, of the message.

Table 3 gives the correspondence between the name and coding of the control field's bits.

Table 3 – Correspondence between name and coding of 8 bits in the control field

Bits	Associated name
1 2 3 4 5 6 7 8	
1 1 0 0 0 0 × x	ID_DAT
1 0 1 0 0 0 × x	ID_MSG
1 0 0 1 0 1 × x	ID_RQ1
1 0 0 1 0 0 × x	ID_RQ2
0 1 0 0 0 0 × x	RP_DAT
0 1 1 0 0 0 × x	RP_DAT_MSG
0 1 0 1 0 1 × x	RP_DAT_RQ1
0 1 0 1 0 0 × x	RP_DAT_RQ2
0 1 1 1 0 1 × x	RP_DAT_RQ1_MSG
0 1 1 1 0 0 × x	RP_DAT_RQ2_MSG
0 0 1 0 1 0 × N	RP_MSG_ACK
0 0 1 0 0 0 × x	RP_MSG_NOACK
0 0 0 0 1 1 × N	RP_ACK+
0 0 0 0 1 0 × N	RP_ACK-
0 0 0 1 0 1 × x	RP_RQ1
0 0 0 1 0 0 × x	RP_RQ2
0 0 0 0 0 0 1 x	RP_END
NOTE N takes the value 0 or 1; x values are ignored but should be 0	

5.2.4 DLS-user-data field

Only response DLPDUs include a DLS-user-data field. The DLS-user-data field contains:

— either the value of a variable,

— or a message.

5.2.5 DLPDU frame check sequence (FCS) field

Within this subclause, any reference to bit *K* of an octet is a reference to the bit whose weight in a one-octet unsigned integer is 2^K.

NOTE This is sometimes referred to as "little endian" bit numbering.

For the protocol Type of this standard, as in other International Standards (for example, ISO/IEC 3309, ISO/IEC 8802 and ISO/IEC 9314-2), DLPDU-level error detection is provided by calculating and appending a multi-bit frame check sequence (FCS) to the other DLPDU fields during transmission to form a "systematic code word"[1)] of length ***n*** consisting of ***k*** DLPDU message bits followed by ***n - k*** (equal to 16) redundant bits, and by calculating during

1) W. W. Peterson and E. J. Weldon, Jr., *Error Correcting Codes* (2nd edition), MIT Press, Cambridge, 1972.

reception that the message and concatenated FCS form a legal (*n*,*k*) code word. The mechanism for this checking is as follows:

The generic form of the generator polynomial for this FCS construction is specified in equation (6) and the polynomial for the receiver's expected residue is specified in equation (11). The specific polynomials for this standard are specified in Table 4. An exemplary implementation is discussed in Annex A.

Table 4 – FCS length, polynomial and expected residual

Item	Value	
n-k	16	
G(x)	$X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^6 + X^3 + X^2 + X + 1$	(notes 1, 2, 3)
R(x)	$X^{15} + X^{14} + X^{13} + X^9 + X^8 + X^7 + X^4 + X^2$	(note 4)

NOTE 1 Code words D(X) constructed from this G(X) polynomial have Hamming distance 4 for lengths ≤ 344 octets and Hamming distance 5 for lengths ≤ 15 octets.

NOTE 2 This G(X) polynomial is relatively prime to all, and is thus not compromised by any, of the polynomials commonly used in DCEs (modems): the differential encoding polynomial $1 + X^{-1}$ and all primitive scrambling polynomials of the form $1 + X^{-j} + X^{-k}$.

NOTE 3 This G(X) polynomial is the optimal 16-bit polynomial for burst error detection over DLPDUs of 300 octets or less when the statistics of the error burst have a Poisson distribution (as is the usual case).

NOTE 4 The remainder R(x) should be 1110 0011 1001 0100 (X^{15} to X^0, respectively) in the absence of errors.

5.2.5.1 At the sending DLE

The original message (that is, the DLPDU without an FCS), the FCS, and the composite message code word (the concatenated DLPDU and FCS) shall be regarded as vectors M(X), F(X), and D(X), of dimension *k*, *n* - *k*, and *n*, respectively, in an extension field over GF(2). If the message bits are $m_1 \ldots m_k$ and the FCS bits are $f_{n-k-1} \ldots f_0$, where

$m_1 \ldots m_8$ form the first octet sent,

$m_{8N-7} \ldots m_{8N}$ form the *N*th octet sent,

$f_7 \ldots f_0$ form the last octet sent, and

m_1 is sent by the first PhL symbol(s) of the message and f_0 is sent by the last PhL symbol(s) of the message (not counting PhL framing information),

NOTE This "as transmitted" ordering is critical to the error detection properties of the FCS.

then the message vector M(X) shall be regarded to be

$$M(X) = m_1X^{k-1} + m_2X^{k-2} + \ldots + m_{k-1}X^1 + m_k \quad (1)$$

and the FCS vector F(X) shall be regarded to be

$$F(X) = f_{n-k-1}X^{n-k-1} + \ldots + f_0 \quad (2)$$
$$= f_{15}X^{15} + \ldots + f_0$$

The composite vector D(X), for the complete DLPDU, shall be constructed as the concatenation of the message and FCS vectors

$$D(X) = M(X)\,X^{n-k} + F(X) \quad (3)$$
$$= m_1X^{n-1} + m_2X^{n-2} + \ldots + m_kX^{n-k} + f_{n-k-1}X^{n-k-1} + \ldots + f_0$$
$$= m_1X^{n-1} + m_2X^{n-2} + \ldots + m_kX^{16} + f_{15}X^{15} + \ldots + f_0 \quad \text{(for the case of k = 15)}$$

The DLPDU presented to the PhL shall consist of an octet sequence in the specified order.

The redundant check bits f_{n-k-1} ... f_0 of the FCS shall be the coefficients of the remainder F(X), after division by G(X), of $L(X)\ (X^k + 1) + M(X)\ X^{n-k}$

where G(X) is the degree ***n-k*** generator polynomial for the code words

$$G(X) = X^{n-k} + g_{n-k-1}X^{n-k-1} + \ldots + 1 \quad (4)$$

and L(X) is the maximal weight (all ones) polynomial of degree ***n-k-1***

$$L(X) = \frac{X^{n-k}+1}{X+1} = X^{n-k-1} + X^{n-k-2} + \ldots + X + 1 \quad (5)$$

$$= X^{15} + X^{14} + X^{13} + X^{12} + \ldots + X^2 + X + 1 \quad \text{(for the case of k = 15)}$$

That is,

$$F(X) = L(X)\ (X^k + 1) + M(X)\ X^{n-k} \ \text{(modulo}\ G(X)) \quad (6)$$

NOTE 1 The L(X) terms are included in the computation to detect initial or terminal message truncation or extension by adding a length-dependent factor to the FCS.

NOTE 2 As a typical implementation when n-k = 16, the initial remainder of the division is preset to all ones. The transmitted message bit stream is multiplied by X^{n-k} and divided (modulo 2) by the generator polynomial G(X), specified in equation (7). The ones complement of the resulting remainder is transmitted as the (n-k)-bit FCS, with the coefficient of X^{n-k-1} transmitted first.

5.2.5.2 At the receiving DLE

The octet sequence indicated by the PhE shall be concatenated into the received DLPDU and FCS, and regarded as a vector V(X) of dimension ***u***

$$V(X) = v_1X^{u-1} + v_2X^{u-2} + \ldots + v_{u-1}X + v_u \quad (7)$$

NOTE 1 Because of errors **u** can be different than **n**, the dimension of the transmitted code vector.

A remainder R(X) shall be computed for V(X), the received DLPDU and FCS, by a method similar to that used by the sending DLE (see 5.2.5.1) in computing F(X)

$$R(X) = L(X)\ X^u + V(X)\ X^{n-k} \ \text{(modulo}\ G(X)) \quad (8)$$

$$= r_{n-k-1}X^{n-k-1} + \ldots + r_0$$

Define E(X) to be the error code vector of the additive (modulo-2) differences between the transmitted code vector D(X) and the received vector V(X) resulting from errors encountered (in the PhS provider and in bridges) between sending and receiving DLEs.

$$E(X) = D(X) + V(X) \quad (9)$$

If no error has occurred, so that E(X) = 0, then R(X) will equal a non-zero constant remainder polynomial

$$R_{ok}(X) = L(X)\ X^{n-k} \ \text{(modulo}\ G(X)) \quad (10)$$

whose value is independent of D(X). Unfortunately R(X) will also equal $R_{ok}(X)$ in those cases where E(X) is an exact non-zero multiple of G(X), in which case there are “undetectable” errors. In all other cases, R(X) will not equal $R_{ok}(X)$; such DLPDUs are erroneous and shall be discarded without further analysis.

NOTE 2 As a typical implementation, the initial remainder of the division is preset to all ones. The received bit stream is multiplied by X^{n-k} and divided (modulo 2) by the generator polynomial G(X), specified in equation (7).

5.3 PhIDU structure and encoding

Each PhIDU consists of Ph-interface-control-information (PhICI) and in some cases one octet of Ph-interface-data. When the DLE transmits a DLPDU, it computes a DLPDU check sequence for the DLPDU as specified in 5.2.5, concatenates the DLPDU and DLPDU check sequence, and transmits the concatenated pair as a sequence of PhIDUs as follows:

a) The DLE issues a single Ph-DATA request primitive with PhICI specifying start-of-activity, and awaits the consequent Ph-DATA confirm primitive.

b) The DLE issues a sequence of Ph-DATA request primitives with PhICI specifying DATA, each accompanied by one octet of the DLPDU as Ph-interface-data, from first to last octet of the DLPDU, and after each Ph-DATA request primitive awaits the consequent Ph-DATA confirm primitive.

c) The DLE issues a sequence of two Ph-DATA request primitives with PhICI specifying DATA, each accompanied by one octet of the FCS as Ph-interface-data, from first to last octet of the FCS, and after each Ph-DATA request primitive awaits the consequent Ph-DATA confirm primitive.

d) The DLE issues a single Ph-DATA request primitive with PhICI specifying END-OF-DATA-AND-ACTIVITY, and awaits the consequent Ph-DATA confirm primitive.

The DLE forms a received DLPDU by concatenating the sequence of octets received as Ph-interface-control-information of consecutive Ph-DATA indications, computing a DLPDU check sequence for those received octets as specified in 5.2.5, and checks the syndrome of the computed FCS for correctness as follows:

e) The DLE receives a single Ph-DATA indication primitive with PhICI specifying START-OF-ACTIVITY, and initializes its computation of an FCS for the received DLPDU.

f) The DLE receives a sequence of Ph-DATA indication primitives with PhICI specifying DATA, each accompanied by one octet of the received DLPDU as Ph-interface-data, incrementally computes an FCS on the received octet, and concatenates all but the last two of those received octets to form the received DLPDU.

g) The DLE receives a single Ph-DATA indication primitive with PhICI specifying either END-OF-DATA, END-OF-DATA-AND-ACTIVITY or END-OF-ACTIVITY, and checks the syndrome of the computed FCS for correctness:

   1) If the PhICI specified END-OF-DATA or END-OF-DATA-AND-ACTIVITY, and the computed FCS syndrome was correct, then the DLE reports the reconstructed DLPDU and the two octets of received FCS as a correctly-received DLPDU suitable for further analysis.

   2) Otherwise the DLE increments its management statistics to reflect the erroneously received DLPDU.

5.4 Common DLPDU structure, encoding and elements of procedure

Each DLPDU consists of a DLPDU control field which specifies the type of DLPDU and conveys small size (fractional octet) parameters of the DLPDU; zero to two explicit address fields, each containing a DL-address, all of the same length; and for most DLPDUs, a user data field conveying all or part of a DLSDU. To this is appended before transmission, and removed after reception, an FCS field used to check the integrity of the received DLPDU.

5.5 Valid DLPDU types

For easier reading, DLPDUs are described without their DLPDU start and DLPDU end sequences.

For each type of DLPDU a name is given for the control field. This name is shown in parenthesis.

The correspondence between this name and the 8-bit control field code is given in the table below.

For each type of DLPDU we have indicated the separation between fields that contain data-link protocol control information (DLPCI) and fields containing data (DLSDU).

5.5.1 Identifier DLPDU

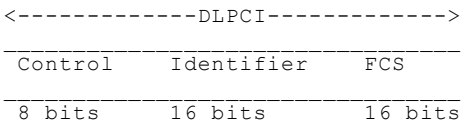


Figure 18 – Identifier DLPDU

There are four types of identifier DLPDU, whose basic structure is shown in Figure 18:

— medium allocation for identified variables (ID_DAT)

— medium allocation for message (ID_MSG)

— medium allocation for urgent explicit request (ID_RQ1)

— medium allocation for normal explicit request (ID_RQ2).

5.5.2 Variable response DLPDU

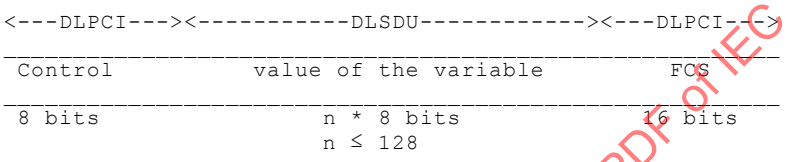


Figure 19 – Variable response DLPDU

There are six types of variable value response DLPDU, whose basic structure is shown in Figure 19:

— identified variables (RP_DAT)

— identified variables + message request (RP_DAT_MSG)

— identified variables + urgent explicit request (RP_DAT_RQ1)

— identified variables + normal explicit request (RP_DAT_RQ2)

— identified variables + urgent explicit request + message request (RP_DAT_RQ1_MSG)

— identified variables + normal explicit request + message request (RP_DAT_RQ2_MSG).

5.5.3 Request response DLPDU

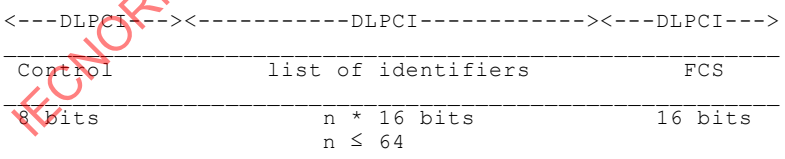


Figure 20 – Request response DLPDU

There are two types of request response DLPDU, whose basic structure is shown in Figure 20:

— list of urgent identifiers (RP_RQ1)

— list of normal identifiers (RP_RQ2).

5.5.4 Message response DLPDU

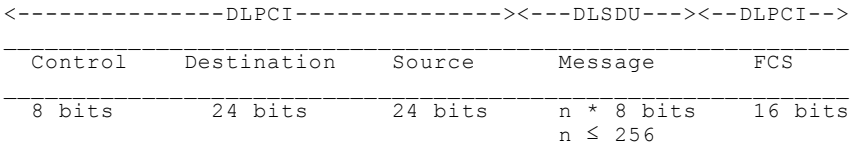


Figure 21 – Message response DLPDU

There are two types of message response DLPDU, whose basic structure is shown in Figure 21:

— acknowledged message (RP_MSG_ACK)

— unacknowledged message (RP_MSG_NOACK).

5.5.5 Acknowledgement response DLPDU

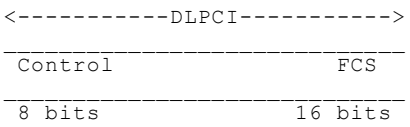


Figure 22 – Acknowledgement response DLPDU

There are two types of acknowledgement response DLPDU, whose basic structure is shown in Figure 22:

— positive acknowledgement (RP_ACK+) indicates that the message has been correctly received by the remote DLE,

— negative acknowledgement (RP_ACK-) indicates that the distant entity's queue for received message is filled.

5.5.6 End of message transaction response DLPDU

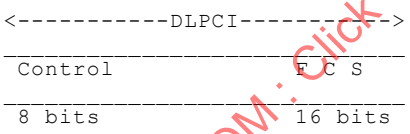


Figure 23 – End of message transaction response DLPDU

There is one type of end of message transaction response DLPDU, whose basic structure is shown in Figure 23:

NOTE The chaining of basic DLPDUs is indicated in the specifications for each service.

The RP_END DLPDU involves 2 entities: the source entity and the bus-arbitrator. The source entity returns control of the local link to the bus-arbitrator at the end of a message transaction.

5.6 DLL timers

This subclause briefly describes the timers involved in the description of data-link final state devices, proposes relationships between the various timers, and describes the importance of a station's turnaround time in the definition of its timers.

5.6.1 Common criteria

The criterion common to all stations for launching timers is the absence of activity on the local link.

Two signals are used to evaluate this criterion: "Absence of activity" (SILENCE) and "Signal present" (BUSY).

A reference timer T0 is defined as corresponding to the maximum silent time permitted on a local link.

The T0 timer is a global systems parameter.

T0 is activated upon reception of an indication of the absence of activity. This indication is emitted by the Type 2 PhL (the primitive PhL-DATA indication (SILENCE)).

T0 is deactivated by any correct functioning of the local link, upon reception of the indication of signal presence. This indication is emitted by the Type 2 PhL (the primitive PhL-DATA indication (BUSY)).

The maximum silent time permitted on the local link, that is, T0 is linked to the turnaround time of stations in the system.

5.6.2 Turnaround time

The reception or emission of the last MAC symbol is signaled by a SILENCE indication from the PhL. Likewise, the reception or emission of the first MAC symbol is signaled by an BUSY indication from the PhL.

The time lapsed between these two signals within a single station defines the station turnaround time (STT), whether the station's function be that of producer/consumer or bus-arbitrator.

Turnaround time takes into account parameters that are local to the station and parameters that are associated with the system to which the station belongs. These parameters are:

— physical reaction time (PRT): the time the PhL needs to detect the end or beginning of activity. These times depend on physical local link parameters:

  station: time needed to detect the absence or presence of a signal

  support: time needed to traverse the media and active copper or optical stars

— latency time (TI) resulting from processing time in the station.

TI takes on various values depending on the protocol situation in which the station finds itself:

a) the station has emitted a DLPDU and will emit the next DLPDU,

b) the station has received a DLPDU and will emit the next DLPDU,

c) the station has emitted a DLPDU and will receive the next DLPDU,

d) the station has received a DLPDU and will receive the next DLPDU.

All the parameters combine to form a minimum station turnaround time (under the most favorable conditions) and a maximum station turnaround time (under the least favorable conditions).

For the system to function correctly the following criteria *must* be respected:

Rule 1) a station that receives a DLPDU must always have a shorter turnaround time than the station that emitted the DLPDU, regardless of the two stations (receiving station, emitting station) involved.

This condition is expressed by:

Max(i) STT receiving station i ≤ Min(j) STT emitting station j

Rule 2) minimum turnaround time corresponds to the maximum value of physical reaction time PRT.

Rule 3) given the impact of a station's turnaround time on transmission yield (use of the bandwidth), it is important that maximum STT take yield criteria into account.

Rules 1 and 2 are interworking criteria. Rule 3 is a performance criterion that can be verified on the level of system management through access to the maximum turnaround time value and the resulting T0 timer.

NOTE The choice of minimum and maximum STT values can result in different interoperability and performance classes.

5.6.3 Timer functions

Five timers are defined using T0 as a basis.

The T1, T4, and T6 timers correspond to the use of T0 in the specific context of the status of a DLE (Station or bus-arbitrator). Activation and deactivation conditions are thus identical (SILENCE or BUSY), and the action after expiration of the timers differs according to the status of the DLE.

The T3 and T5 timers also refer to T0: their lengths (which are different from T0) are calculated using T0 length as a basis. In addition, they are activated and deactivated under the same conditions (SILENCE or BUSY).

An additional T2 timer is also used. T2 is not based on T0. This timer determines the length of a basic synchronized cycle.

All of these timers are listed in Table 5

Table 5 – DL-Timers

Name of timer	Location	Function
T1	B.A.	monitors the absence of RPxx after IDxx
T2	B.A.	monitors the filling of the synchronization window by identifiers from the basic padding sequence
T3	B.A.	monitors the absence of IDxx after RPxx switching of bus-arbitrators
T4	consuming station	monitors the absence of RP_DATxx after ID_DAT
T5	B.A.	monitors the lack of RP_END after ID_MSG
T6	message source station	monitors the absence of RP_ACK after RP_MSG_ACK

Timer definitions refer to names of statuses mentioned in the protocol descriptions presented in Clause 7.

5.6.3.1 Timer T1

— Is located in the active bus-arbitrator

— Its purpose is to monitor the emission of a variable response DLPDU or a request response DLPDU within a given time.

— Is active when the bus-arbitrator is waiting for a variable response DLPDU or a request response DLPDU (BA_WAITING_DAT or BA_WAITING_RQ status), after its emission of a variable identifier or request identifier DLPDU.

— The length of timer T1 is equal to the length of timer T0.

— Expiration of the timer gives the bus-arbitrator a new status: that of emitting the next identifier DLPDU (NEXT status).

5.6.3.2 Timer T2

— Is located in the active bus-arbitrator

— Its purpose is to monitor the length of the synchronization window of the basic cycle. The active bus-arbitrator emits the identifier DLPDUs of the basic padding sequence as long as the T2 timer has not expired.

— Is activated at the beginning of each basic synchronized cycle, when the bus-arbitrator takes on the status of emitting identifier DLPDUs (NEXT status after BA_WAITING_SYNC status).

— Is not deactivated by any event.

— The length of the synchronization window is a function of the basic cycle's aperiodic exchanges or message services;

— Expiration of the timer gives the bus-arbitrator a new status: that of emitting the next identifier DLPDU (NEXT status).

5.6.3.3 Timer T3

— Is located in potential bus-arbitrators

— Its purpose is to monitor the active bus-arbitrator's emission of identifier DLPDUs within a given time.

— Is active as long as the bus-arbitrator has potentially active status.

— The length of the T3 timer should be superior to the length of the T0 timer. It should have a different value for each of the potential bus-arbitrators. The note below suggests a value for this length.

— Expiration of the timer triggers the activation of a bus-arbitrator.

NOTE T3 timer dimensions are set as follows:

T3 = $k \times n \times$ T0 with
n>2 a function of the geographic address of the bus-arbitrator to which T3 is attached and
k a coefficient that allows the bus-arbitrator's turnaround time to be covered.

5.6.3.4 Timer T4

— Is located in stations

— Its purpose is to monitor the emission of a variable response DLPDU on the bus within a given time, in order for example to correctly associate a DLCEP-identifier DLPDU received with the corresponding response DLPDU.

— Is active when the station is waiting for a variable response DLPDU (WAITING_DAT status).

— The length of T4 is equal to the length of T0

— Expiration of the timer causes the station to be placed in the status of waiting for the next identifier DLPDU (WAITING_ID status).

5.6.3.5 Timer T5

— Is located in the active bus-arbitrator

— Its purpose is to monitor a message source station's emission of a response DLPDU indicating the end of a message transaction within a given time.

— Is active when the bus-arbitrator is waiting for an end of message response DLPDU (BA_WAITING_END status), after its emission of a message identifier DLPDU.

— The length of timer T5 should be greater than the length of timer T0. The note below suggests a value for T5.

— Expiration of the timer gives the bus-arbitrator a new status: that of emitting the next identifier DLPDU (NEXT status).

NOTE The length of timer T5 is equal to two times the length of timer T0.

With such a choice of lengths collision between two operators is avoided. For example, when an acknowledgement DLPDU RP_ACK is lost the bus-arbitrator cannot emit a second ID_MSG at the same time as the source station answers with the emission of RP_MSG_ACK (see T6).

5.6.3.6 Timer T6

— Is located in stations

— Its purpose is to monitor the emission of an acknowledgement DLPDU by the destination station within a given time.

— Is active when the source station is waiting for an acknowledgement DLPDU (WAITING_ACK status) after the end of its emission of an acknowledged message response DLPDU.

— The length of T6 is equal to the length of T0

— Expiration of the timer causes the station to be placed in the status of re-emitting the message if possible (RESTART_SOURCE status) or else in the status of waiting for the next identifier DLPDU (WAITING_ID status) after emitting the end of message transaction DLPDU.

The definition of these timers is necessary to guarantee the ability to interoperate of the various devices connected to a DLL local link.

All the timers within a single local link should have the same value. A default value is defined. These values can be modified by a station's system management as long as the equivalence of values is respected. Values will be agreed upon later in accordance with implementation needs.

6 DLPDU-specific structure, encoding and element of procedure

6.1 General

This clause describes the structure, contents of the PDU and specifies elements of procedure related to the invoked service.

Behavior of particular cases is described in detail in the relative subclause by the use of dedicated state machines.

6.2 Buffer read

Buffer reading service is local to an entity since the interactions related to this service take place only through the DLS-user/DLL interface, as shown in Figure 24. Correspondence

between DLSDU service data units and interactions between the DLL and the communication medium are provided by the buffer transfer service.

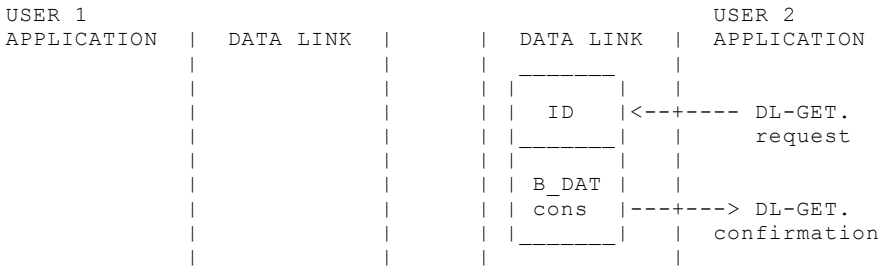


Figure 24 – Buffer reading service interactions

6.3 Buffer write

Buffer writing service is local to an entity since the interactions related to this service take place only through the DLS-user/DLL interface, as shown in Figure 25. Correspondence between DLSDU service data units and interactions between the DLL and the communication medium are provided by the buffer transfer service

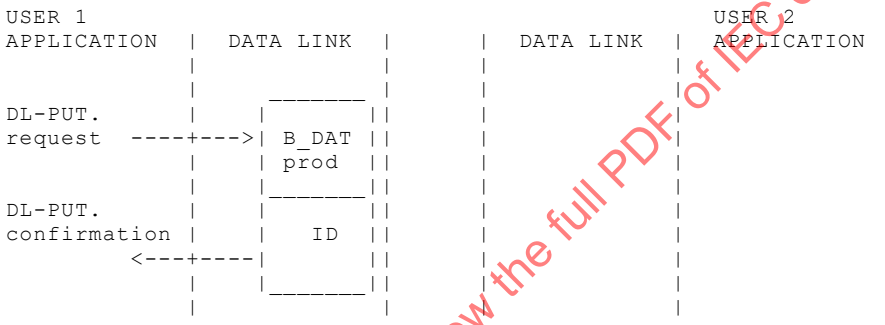


Figure 25 – Buffer writing service interactions

6.4 Buffer transfer

There are two phases to buffer transfer for all variables, as shown in Figure 26:

— the bus-arbitrator emits the identifier DLPDU associated with the variable,

— the producing entity emits the corresponding response DLPDU and generates a DL-SENT indication primitive for the DLS-user. The value of the variable is picked up by all consuming entities and a DL-RECEIVED indication primitive is generated and passed on to each of the DLS-users concerned.

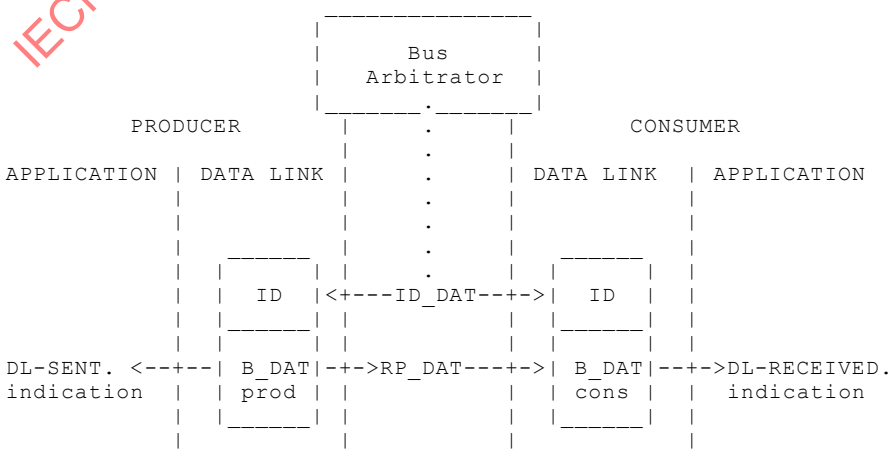


Figure 26 – Buffer transfer service interactions

6.4.1 Associated DLPDUs

A buffer transfer involves the circulation of two DLPDUs.

The bus-arbitrator broadcasts a DLCEP-identifier DLPDU (ID_DAT) that allocates the medium to the entity that produces the associated variable. The producing entity then broadcasts the variable in a response DLPDU (RP_DAT) that is picked up by the consuming entity or entities. The phases of this sequence are shown in Figure 27.

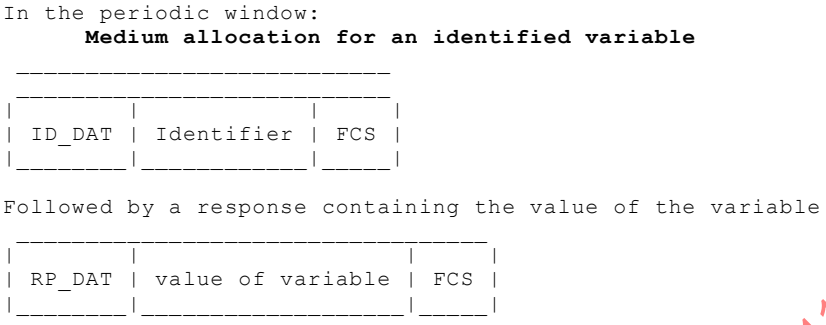


Figure 27 – Buffer transfer DLPDU sequence

NOTE A description of all DLPDUs, and the use of the various fields, can be found in 4.5.

6.5 Specified explicit request

A specified explicit request for a variable exchange can be initiated by any entity. The request triggers exchanges whose form depends on the value of the indicator RQ_INHIBIT associated with SPECIFIED_IDENTIFIER. This value is set upon configuration.

If RQ_INHIBIT = FALSE, there are three phases to the exchange:

phase 1:

Following a DL-SPEC-UPDATE request primitive, the initiating entity uses the control field of the variable response DLPDU (corresponding to the identifier specified when the service was requested) to ask the bus-arbitrator to circulate an urgent request identifier.

phase 2:

The bus-arbitrator emits the corresponding request identifier in the window allocated to triggered variable exchanges (aperiodic window),

phase 3:

The initiating entity emits a request response DLPDU whose "data" field contains the list of identifiers that the entity would like broadcast. The initiating entity also transmits a DL-SPEC-UPDATE confirm primitive to the DLS-user. One or more buffer transfers then take place within the aperiodic traffic cycle.

— the bus-arbitrator emits the identifiers of the variables requested,

— the producer of each variable requested emits a variable response DLPDU.

If RQ_INHIBIT = TRUE, there are two phases to the exchange:

phase 1:

Regardless of requests, for all identifiers set aside for specified service whose RQ_INHIBIT indicator is "true" the bus-arbitrator emits a request identifier in the periodic window.

phase 2:

The initiating entity emits a request response DLPDU whose "data" field contains the list of variable identifiers that the entity would like broadcast. The initiating entity also transmits a DL-SPEC-UPDATE confirmation primitive to the DLS-user. One or more buffer transfers then take place within the periodic traffic cycle immediately following the request.

— the bus-arbitrator emits the identifiers of the variables requested,

— the producer of each variable requested emits a variable response DLPDU.

The interaction sequences for the two classes of non-erroneous operation are shown in Figure 28 and Figure 29.

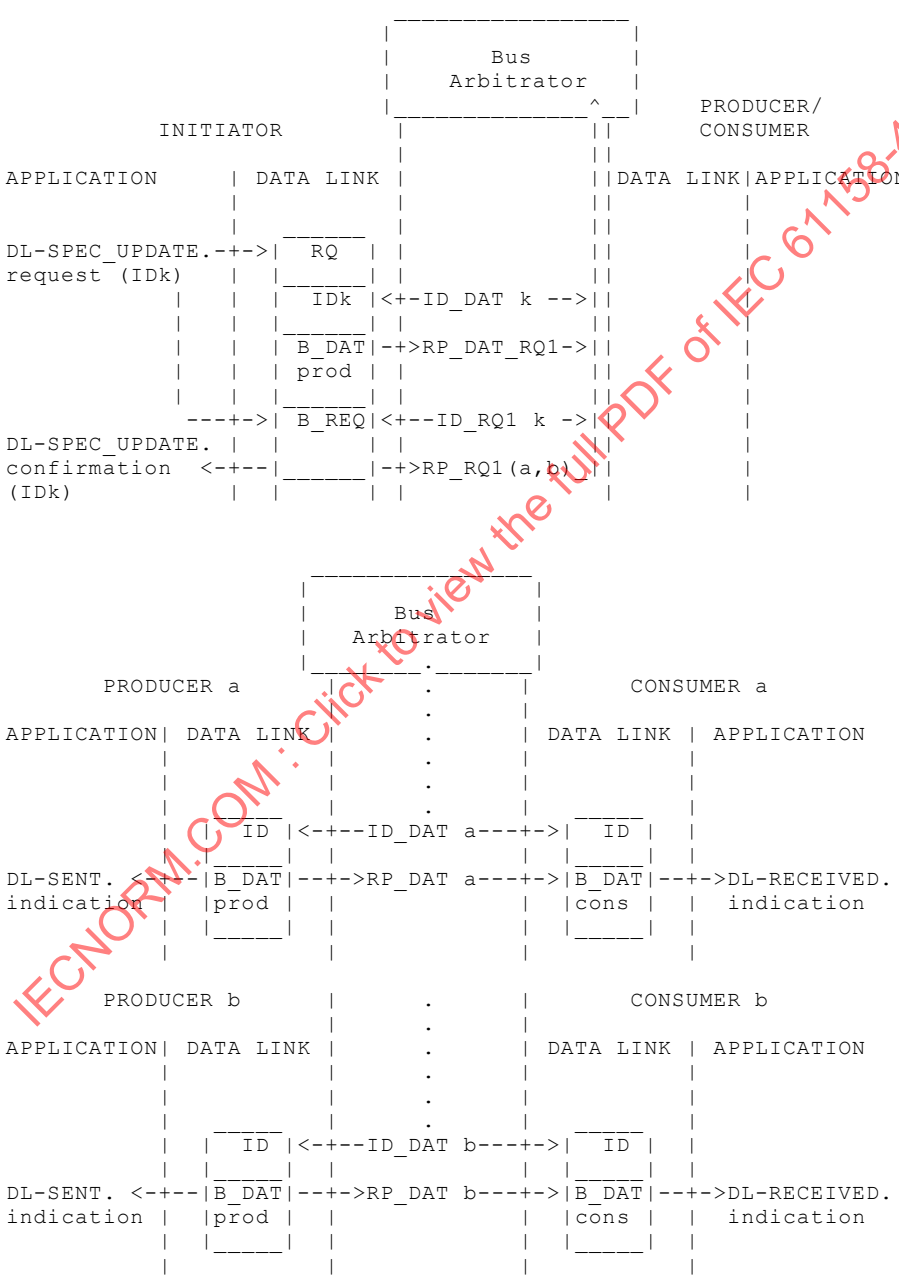


Figure 28 – Interactions within the specified explicit request for buffer transfer service in the aperiodic window

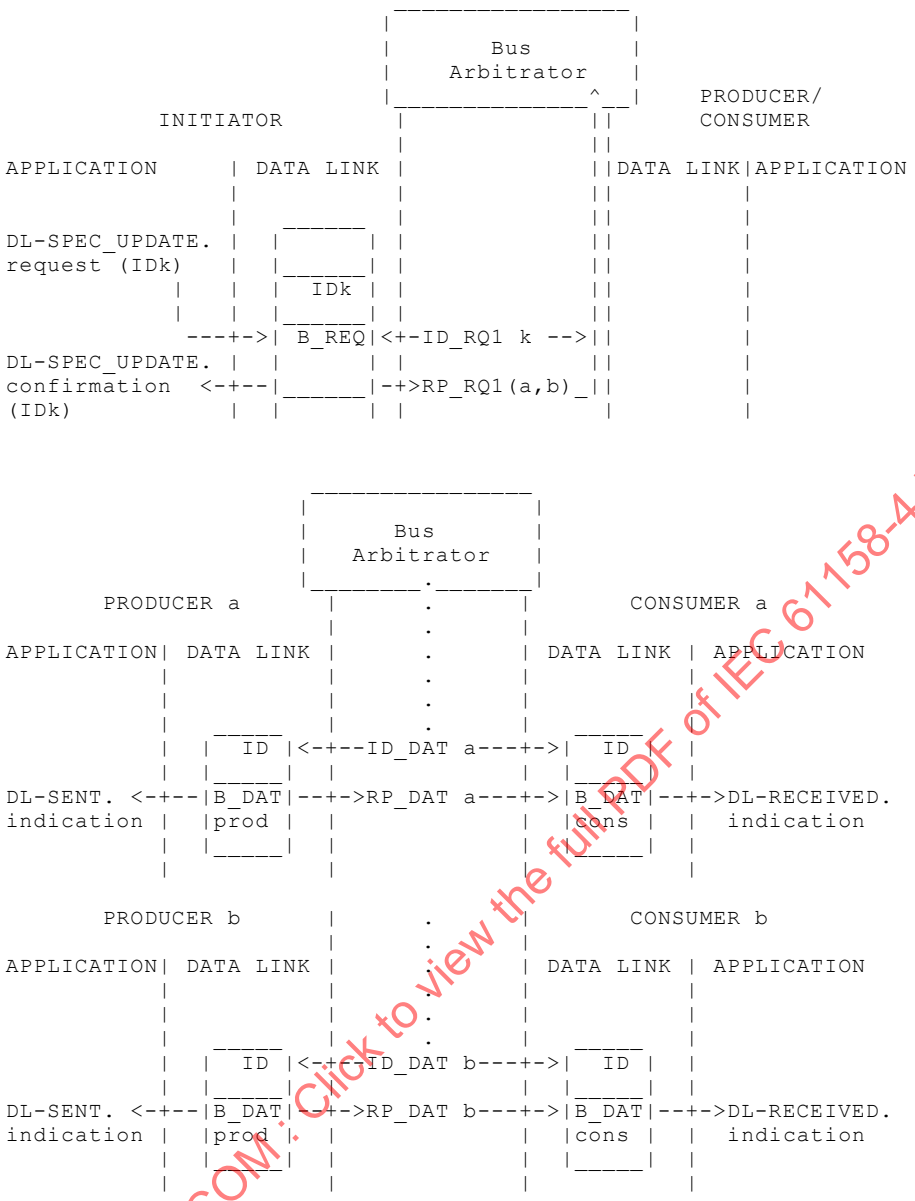


Figure 29 – Interactions within the specified explicit request for buffer transfer service in the periodic window

6.5.1 Associated DLPDUs

An explicit request for a buffer transfer corresponds to the circulation of three DLPDUs. The first of these DLPDUs is part of normal periodic scanning.

The initiating entity broadcasts a variable exchange response DLPDU whose control field includes the request to the bus-arbitrator to circulate a request identifier. The initiating entity states the priority level (RP_DAT_RQi, or RP_DAT_RQi_MSG if an aperiodic message service is also requested with this identifier).

The bus-arbitrator then broadcasts a request identifier DLPDU (ID_RQi) during an aperiodic window. The bus-arbitrator uses the identifier which carried the request.

The list of variable identifiers to be broadcast is sent to the bus-arbitrator (RP_RQi).

The bus-arbitrator responds with one or more buffer transfers: (ID_DAT, RP_DAT) transaction.

If the specified explicit request is part of the bus-arbitrator's periodic scanning cycle, the circulation of the first DLPDU is null and the other DLPDUs are exchanged during aperiodic scanning.

Figure 30 shows the DLPDU sequence for an explicit request for a transfer:

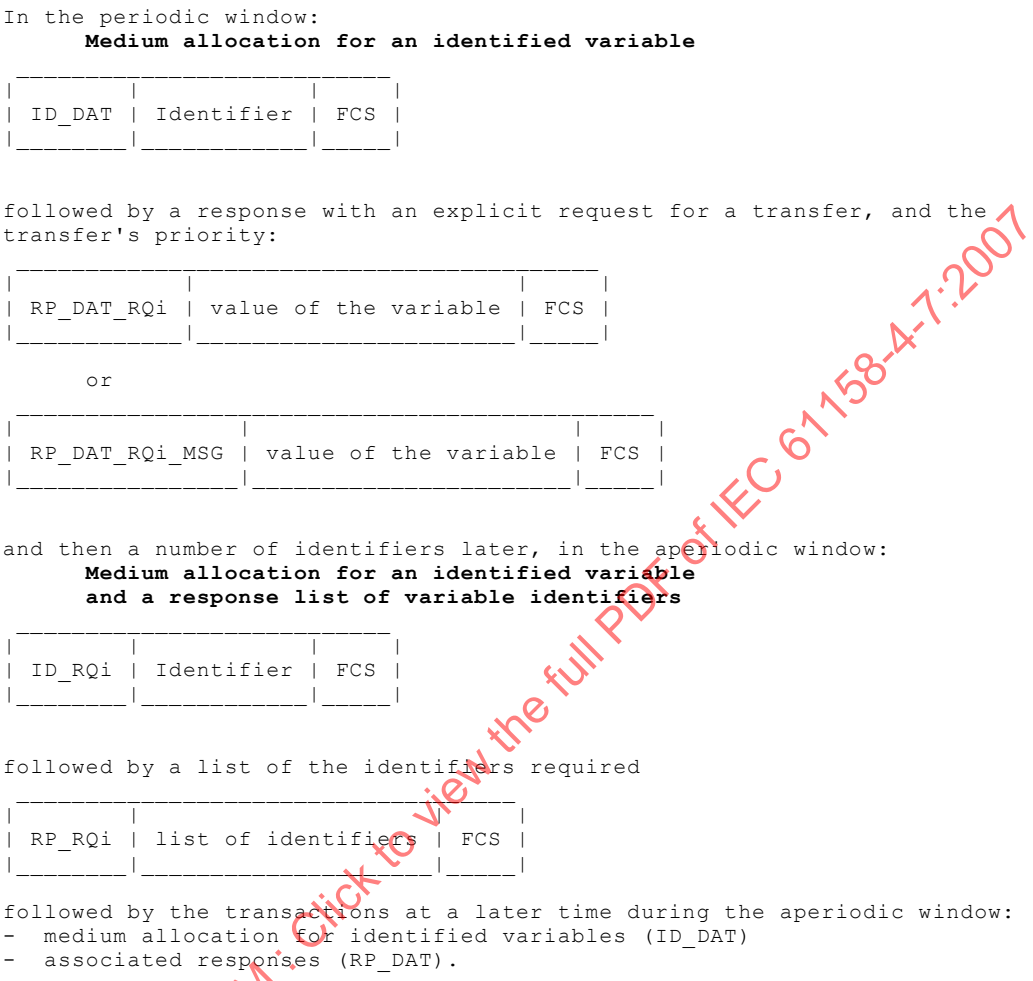


Figure 30 – DLPDU sequence for an explicit request for a transfer

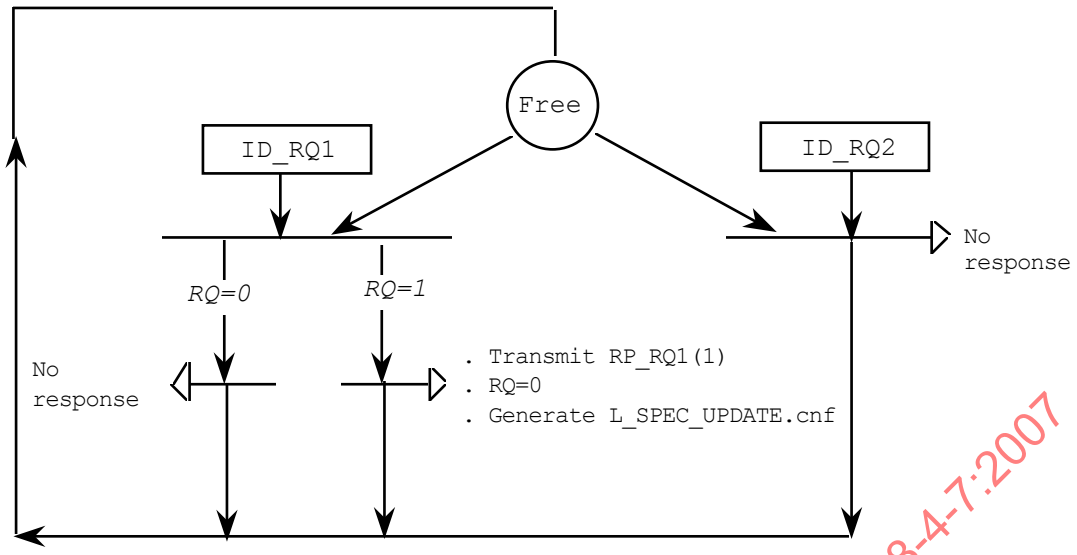
6.5.2 Behavior with mismatched parameters

6.5.2.1 Buffer transfer specified explicit request (RQ_INHIBITED = False)

The evaluation network of Figure 31 allows specifying the behavior of a producer/consumer entity in the following cases:

— reception of an ID_RQ2 DLPDU containing a DLCEP-identifier bearing a buffer transfer exchange specified explicit request (RQ=1),

— reception of an ID_RQ1 DLPDU containing a DLCEP-identifier not bearing a buffer transfer specified explicit request (RQ=0).



NOTE 1 When RQ=1 and B_REQ contains an empty list, RP_RQ1 is transmitted with an empty list.

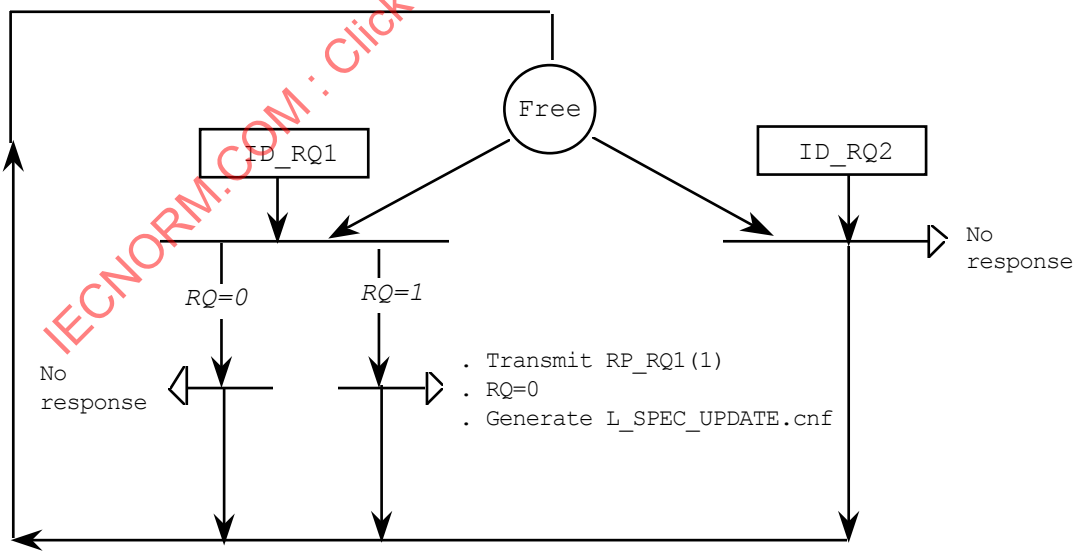
Figure 31 – Evaluation network for a buffer transfer specified explicit request with (RQ_INHIBITED = False)

6.5.2.2 Buffer transfer specified explicit request (RQ_INHIBITED = True)

The evaluation network of Figure 32 allows specifying the behavior of a producer/consumer entity in the following cases:

— reception of an ID_RQ2 DLPDU containing a DLCEP-identifier configured for the service of the buffer transfer specified explicit request such as RQ_INHIBITED=TRUE,

— reception of an ID_RQ1 DLPDU containing a DLCEP-identifier configured for the service of the buffer transfer specified explicit request such as RQ_INHIBITED=TRUE and whose content has already been transmitted on the bus.



NOTE 1 When RQ=1 and B_REQ contains an empty list, RP_RQ1 is transmitted with an empty list.

Figure 32 – Evaluation network for a buffer transfer specified explicit request with (RQ_INHIBITED = True)

6.6 Free explicit request

A free explicit request for the exchange of an identified variable can be initiated by any entity and takes place in three phases.

phase 1:

Following a DL-FREE-UPDATE request primitive, the initiating entity uses the control field of the variable response DLPDU to ask the bus-arbitrator to circulate a request identifier. The entity indicates priority.

phase 2:

The bus-arbitrator emits the corresponding request identifier in the window allocated to triggered variable exchanges (aperiodic window),

phase 3:

The initiating entity emits a request response DLPDU whose "data" field contains the list of variable identifiers that the entity would like broadcast. The initiating entity also transmits a DL-FREE-UPDATE confirmation primitive to the DLS-user.

One or more buffer transfers then take place within the aperiodic traffic cycle.

The interaction sequences for non-erroneous operation is shown in Figure 33.

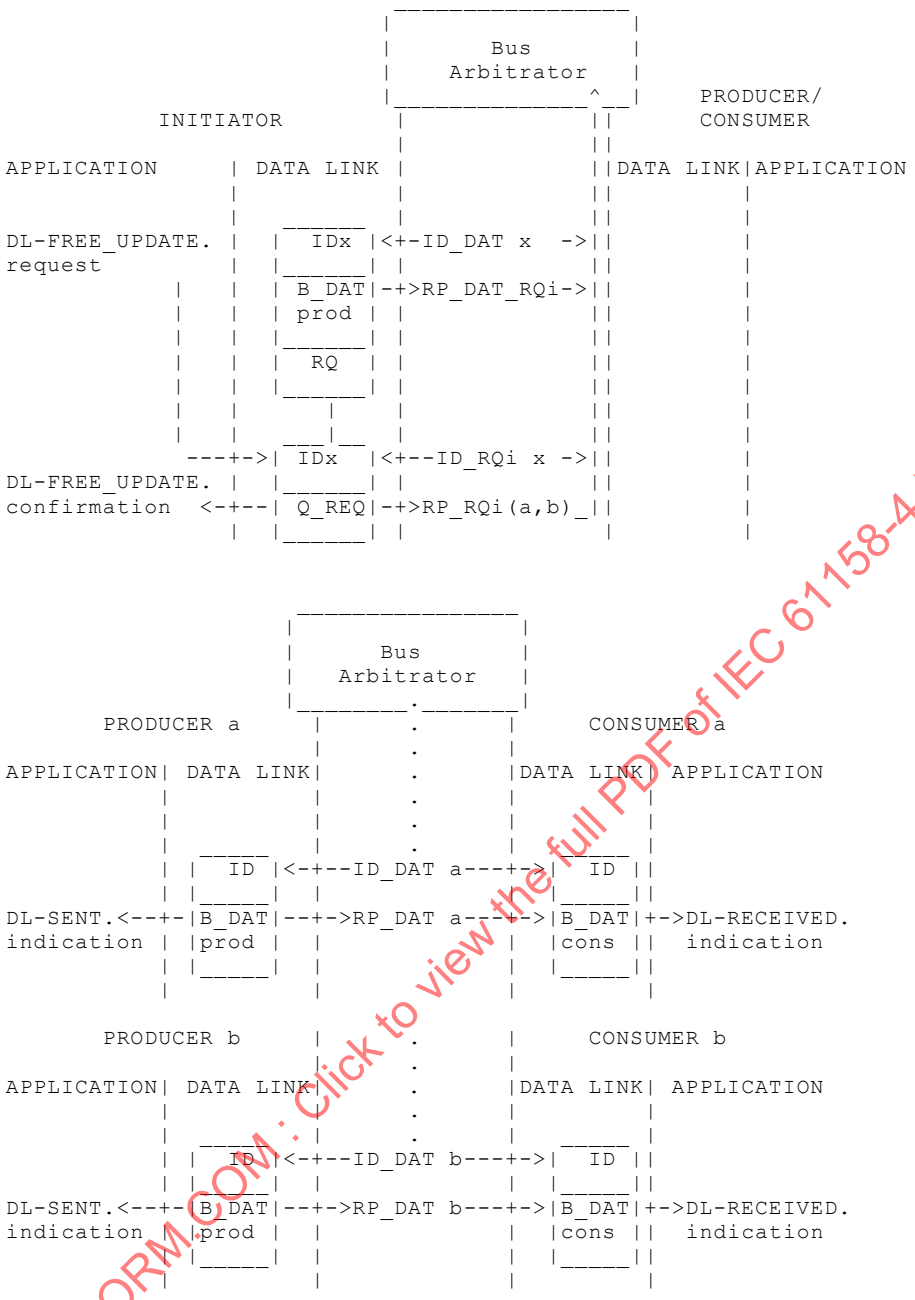


Figure 33 – Diagram of interactions within the free explicit request for buffer transfer service

6.6.1 Associated DLPDUs

A free explicit request for a buffer transfer corresponds to the circulation of three DLPDUs, as shown in 6.5.1. The first of these DLPDUs is part of normal periodic scanning.

The initiating entity broadcasts a variable exchange response DLPDU whose control field includes the request to the bus-arbitrator to circulate a request identifier. The initiating entity states the priority level (RP_DAT_RQi, or RP_DAT_RQi_MSG if an aperiodic message service is also requested with this identifier).

The bus-arbitrator then broadcasts a request identifier DLPDU (ID_RQi) during an aperiodic window. The bus-arbitrator uses the identifier, which carried the request.

The list of variable identifiers to be broadcast is sent to the bus-arbitrator (RP_RQi).

The bus-arbitrator responds with one or more buffer transfers: (ID_DAT, RP_DAT) transaction.

If the specified explicit request is part of the bus-arbitrator's periodic scanning cycle, the circulation of the first DLPDU is null and the other DLPDUs are exchanged during period scanning.

6.6.2 Behavior with mismatched parameters

The evaluation network of Figure 34 allows specifying the management of a producer/ consumer entity in the following cases.

— Reception of an ID_RQ1 or ID_RQ2 DLPDU containing a DLCEP-identifier configured for service of the buffer transfer free explicit request and not associated with Q_REQ1 or Q_REQ2 (RQ of the identifier has a value of 0).

— Reception of an ID_RQ1 or ID_RQ2 DLPDU containing a DLCEP-identifier associated with an empty Q_REQ1 or Q_REQ2 queue.

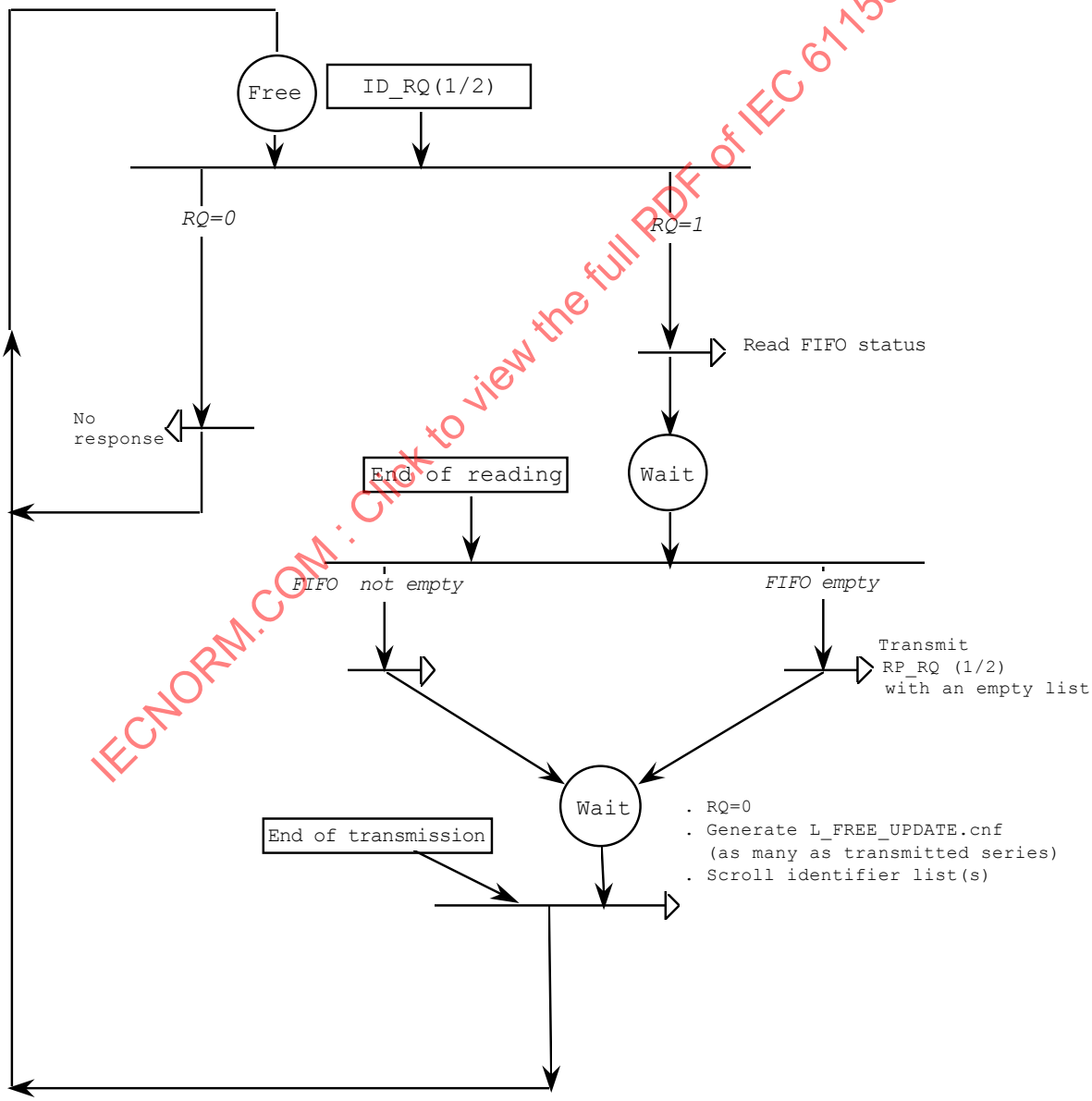


Figure 34 – Evaluation network for a free explicit request

After transmission of the RP_RQ, the request is detached from the separator concerned (RQ set to 0); If the FIFO is not empty, the request will be attached during the next reception of an ID_DAT concerning a DLCEP-identifier configured for this service. The RQ bit of this identifier will be set to 1.

6.7 Messaging

6.7.1 General introduction to message services

Message services make it possible to exchange acknowledged or unacknowledged messages.

The initiator of the message exchange request is the DLS-user that is the source of the message. The request is filled either during aperiodic scanning or during the bus-arbitrator's periodic scanning cycle, depending on configuration.

If the request is filled during aperiodic scanning the message is deemed aperiodic. A single resource is set aside for this type of message: a queue that is dynamically linked to one or more identifiers configured for this mode of message transfer. The variable response DLPDUs associated with these identifiers carry the message transfer requests. Requests are filled in the aperiodic window.

If the request is filled during periodic scanning it is deemed cyclical, and the message transfer request takes place in the periodic window by configuration. Resources are allocated upon configuration and do not change. The resource set aside is a queue attached to a DLCEP-identifier configured for this mode of transfer. The message identifier DLPDU associated with this identifier is transmitted in the periodic window.

A single identifier cannot be configured for both message transfer modes.

Service primitives for message transfer requests are identical for both aperiodic and cyclical modes. Only one parameter of the request primitive indicates the message transfer mode requested and makes it possible to call upon the proper mechanism within the source DLE.

To put the communicating entities into contact two addresses are given during the message transaction:

— a destination address (access point for a message service) made up of 24 bits that encode the local link number of the entity and the entity's sub-address within that DL-segment;

— a source address (access point for a message service) made up of 24 bits that encode the local link number of the entity and the entity's sub-address within that DL-segment.

An access point for a message service addresses an Application message service entity for emission (source) and for reception (destination). Each address is unique within the system.

The format of the destination address and the source address is described in detail in an appendix.

Each entity detects the message response DLPDU (RP_MSG_NOACK or RP_MSG_ACK) and verifies its destination address to determine whether or not it is the destination of the message.

A recovery mechanism intervenes upon detection of the loss of an acknowledgement DLPDU linked to an acknowledged message transfer.

This mechanism for recovery upon loss of acknowledgement, and the restart counters, are found within the source entity. The number of authorized restarts is the same for all source entities within the system, and this number can be set upon configuration. The recovery mechanism can lead to the duplication of messages. An algorithm for numbering messages between the source entity and the destination entity is used to detect message duplication caused by the recovery mechanism (see 4.4.3).

6.7.2 Diagram of interactions

Calling upon the unacknowledged message transfer request service causes the addition of a message to the queue for messages to be emitted. This queue is designated by the parameter SPECIFIED_IDENTIFIER. How the exchange is carried out depends on this queue's type.

6.7.2.1 Aperiodic transfers

The queue is set aside for aperiodic transfer. The transfer includes four phases:

phase 1:

Upon reception of a DLCEP-identifier that can support an aperiodic request, and that is not already in use, the source entity indicates in the control field of the variable response DLPDU its request that the bus-arbitrator circulate a message transfer DLPDU, provided that the number of occupied places in the queue is inferior to the number of identifiers associated with this resource.

In addition, for all identifiers already in use the source entity maintains the indication of a message transfer request in the variable response DLPDU.

phase 2:

The bus-arbitrator gives control of the local link to the source entity by emitting a message DLSAP-address in the window allocated to messages.

phase 3:

The source entity emits the first message stored in the queue of messages to be emitted. The entity emits the message in a message response DLPDU whose extended control DLPDU contains the destination address and source address of the message.

phase 4:

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signifies the end of the message transaction. A DL-MESSAGE confirm primitive is sent to the DLS-user (This primitive's SPECIFIED_IDENTIFIER parameter has the value NIL).

NOTE If the source entity has no message associated with the message identifier DLPDU being circulated (empty queue) it returns control of the local link to the bus-arbitrator immediately, and phase 3 is eliminated.

These phases are shown diagramatically in Figure 35.

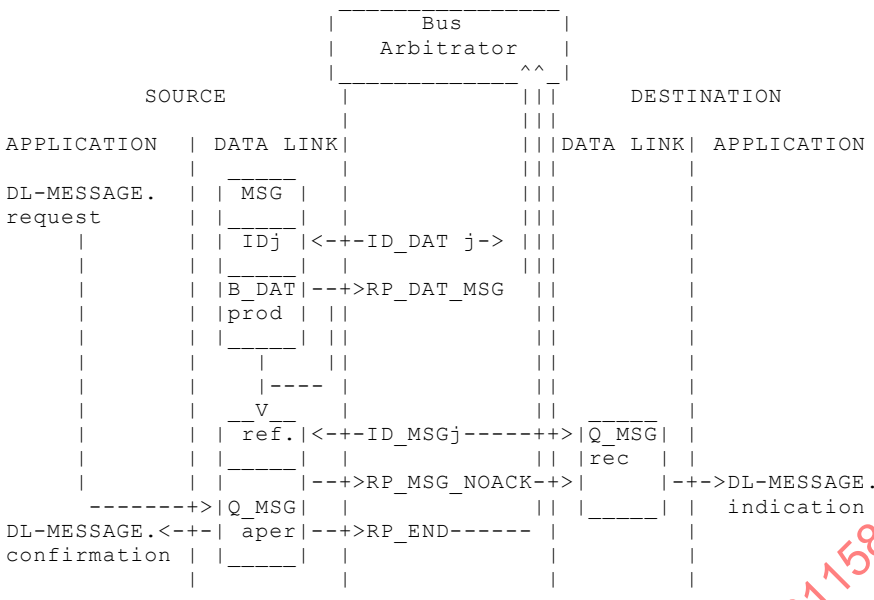


Figure 35 – Diagram of interactions within the unacknowledged message transfer request service for an aperiodic transfer

6.7.2.2 Cyclical transfers

The resource available for reception is a queue known as Q_MSGrec. The queue is set aside for cyclical transfers.

There are three phases to transfers:

phase 1:

The bus-arbitrator gives the source entity control of the local link by emitting a message identifier DLPDU in the periodic window,

phase 2:

The source entity emits the message response DLPDU whose extended control field contains the message's destination and source addresses.

phase 3

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signifies the end of the message transaction. A DL-MESSAGE confirm primitive is sent to the DLS-user (This primitive's SPECIFIED_IDENTIFIER parameter corresponds to the identifier in the message identifier DLPDU).

NOTE If the source entity has no message associated with the message identifier DLPDU being circulated (empty queue) it returns local link control to the bus-arbitrator immediately, and phase 2 is eliminated.

The resource available for reception is a queue known as Q_MSGrec.

These phases are shown diagramatically in Figure 36.

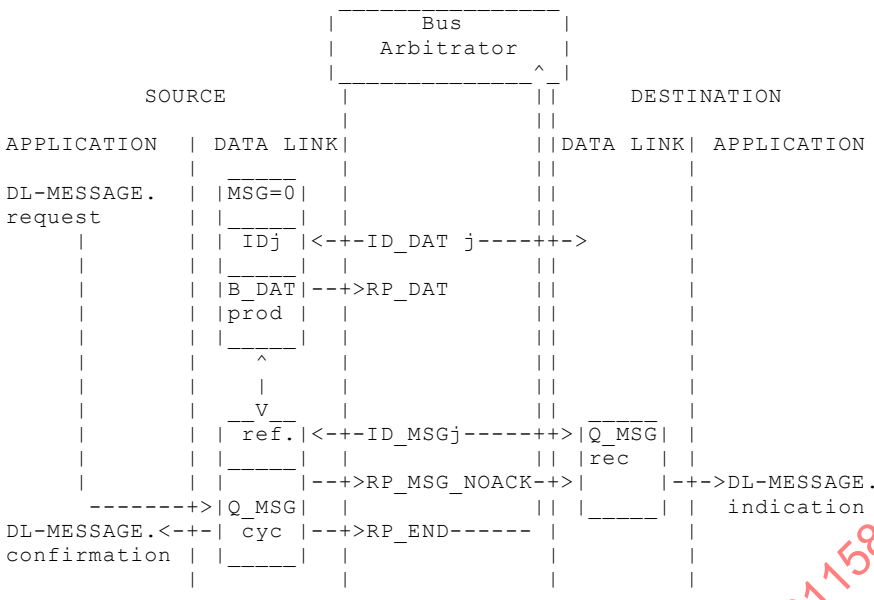


Figure 36 – Diagram of interactions within the unacknowledged message transfer request service for a cyclical transfer

6.7.3 Associated DLPDUs

An unacknowledged message transfer request involves the circulation of four DLPDUs in the case of an aperiodic message transfer. In the case of a cyclical message transfer, the first DLPDU is not broadcast.

6.7.3.1 Aperiodic message transfer

In the case of an aperiodic message transfer the source entity broadcasts a variable response DLPDU whose control DLPDU contains the request that the bus-arbitrator circulate a message identifier DLPDU (RP_DAT_MSG or RP_DAT_RQi_MSG). The first of these DLPDUs is part of normal cyclical traffic.

The identifier refers to the queue for messages to be emitted that is set aside for aperiodic message transfer.

The bus-arbitrator gives local link control to the source entity during an aperiodic message window by broadcasting a message identifier DLPDU (ID_MSG) with a DLCEP-identifier that has carried a request.

The message is emitted by the source entity (RP_MSG_NOACK) which states in the extended control field the message destination and source addresses.

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signals the end of the message transaction (RP_END).

On reception of a PDU containing a destination address which is homogeneous with a group address, layer 2 consults the local directory (by specific functions) in order to have the cross-reference between this group address and one or more local individual addresses to which is sent the DL-MESSAGE indication(Ad,As,m) indication primitive with the group address contained in the PDU as destination address.

Figure 37 shows the DLPDU sequence as it occurs in the periodic and aperiodic windows:

Figure 37 – DLPDU sequence for an aperiodic message transfer

6.7.3.2 Cyclical message transfer

In the case of a cyclical message transfer, the bus-arbitrator gives control to the source entity by broadcasting a message identifier DLPDU (ID_MSG) in the periodic window.

The message is emitted by the source entity (RP_MSG_NOACK) which notes in the extended control field the destination and source addresses.

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signals the end of the message transaction (RP_END).

On reception of a PDU containing a destination address which is homogeneous with a group address, layer 2 consults the local directory (by specific functions) in order to have the cross-reference between this group address and one or more local individual addresses to which is sent the DL-MESSAGE indication(Ad,As,m) indication primitive with the group address contained in the PDU as destination address.

Medium allocation for an identified variable is shown in Figure 38.

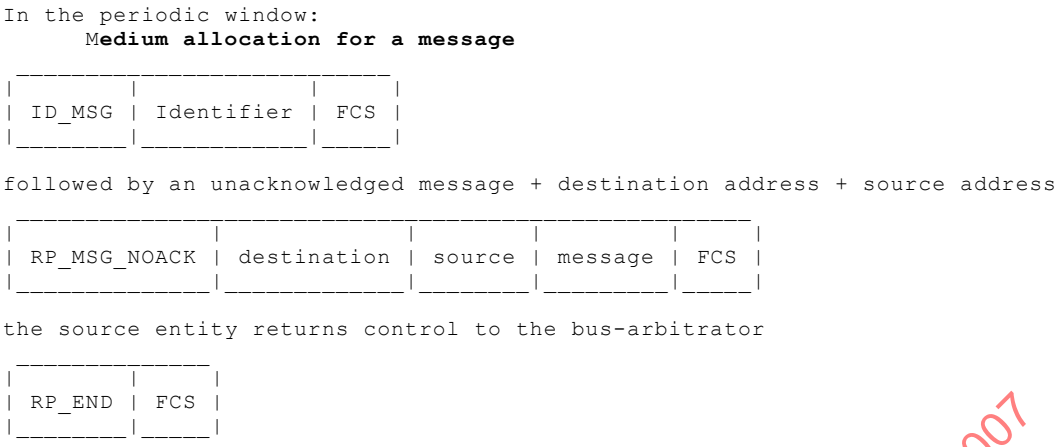


Figure 38 – DLPDU sequence for a cyclical message transfer

6.8 Acknowledged messaging

6.8.1 Diagram of interactions

Calling upon the acknowledged message transfer request service (DL-MESSAGE-ACK request) causes the addition of a message to the queue of messages to be emitted. This queue is designated by the parameter SPECIFIED_IDENTIFIER. The manner in which exchanges are carried out depends on the type of queue.

6.8.1.1 Queue dedicated to aperiodic transfer

If the queue is set aside for aperiodic transfer the exchange includes five phases:

phase 1:

Upon reception of a DLCEP-identifier that can support an aperiodic request, and that is not already in use, the source entity indicates in the control field of the variable response DLPDU its request that the bus-arbitrator circulate a message transfer DLPDU, provided that the number of occupied places in the queue is inferior to the number of identifiers associated with this resource.

In addition, for all identifiers already in use the source entity maintains the indication of a message transfer request in the variable response DLPDU.

phase 2:

The bus-arbitrator gives control to the source entity by emitting a message DLSAP-address in the window allocated to messages.

phase 3:

The source entity emits the first message stored in the queue of messages to be emitted. The entity emits the message in a message response DLPDU whose extended control DLPDU contains the destination address and source address of the message.

phase 4:

The destination entity emits the acknowledgement DLPDU immediately, except if the address destination is a group address. In this case, the entity ignores the DLPDU.

phase 5:

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signifies the end of the message transaction. A DL-MESSAGE-ACK confirm primitive is sent to the DLS-user (This primitive's SPECIFIED_IDENTIFIER parameter has the value NIL).

NOTE If the source entity has no message associated with the message identifier DLPDU being circulated (empty queue) it returns local link control to the bus-arbitrator immediately, and phases 3 and 4 are eliminated.

The resource available for reception is a queue known as Q_MSGrec.

These phases are shown diagramatically in Figure 39.

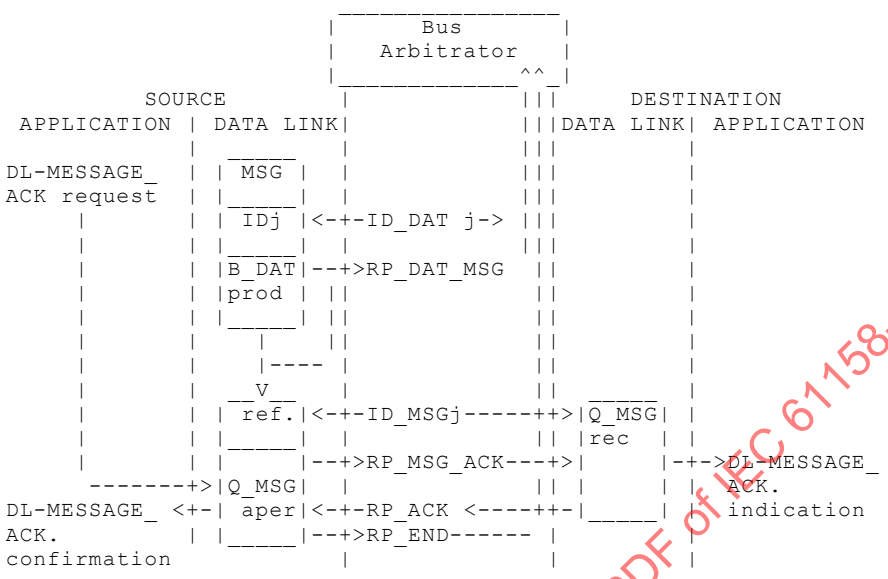


Figure 39 – Diagram of interactions within the acknowledged message transfer request service for an aperiodic transfer

6.8.1.2 Queue dedicated to cyclical transfers

The queue is set aside for cyclical transfers. There are four phases to transfers:

phase 1:

The bus-arbitrator gives the source entity control of the local link by emitting a message identifier DLPDU in the periodic window**.**

phase 2:

The source entity emits the message response DLPDU whose extended control field contains the message's destination and source addresses.

phase 3:

The destination entity emits the acknowledgement DLPDU immediately, except if the address destination is a group address. In this case, the entity ignores the DLPDU.

phase 4:

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signifies the end of the message transaction. A DL-MESSAGE-ACK confirm primitive is sent to the DLS-user. (This primitive's SPECIFIED_IDENTIFIER parameter corresponds to the identifier in the message identifier DLPDU).

NOTE If the source entity has no message associated with the message identifier DLPDU being circulated (empty queue) it returns local link control to the bus-arbitrator immediately, and phases 2 and 3 are eliminated.

The resource available for reception is a queue known as Q_MSGrec.

These phases are shown diagramatically in Figure 40.

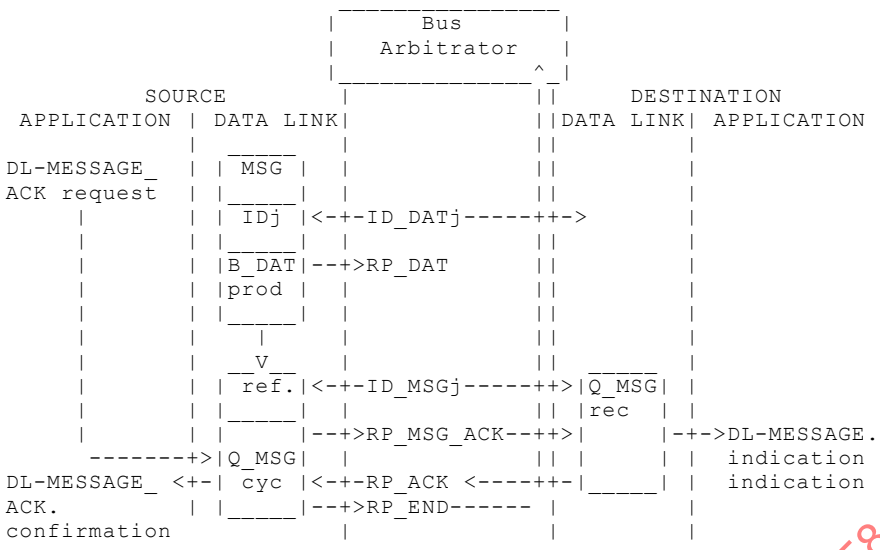


Figure 40 – Diagram of interactions within the acknowledged message transfer request service for a cyclical transfer

6.8.2 Associated DLPDUs

An acknowledged message transfer request involves the circulation of five DLPDUs in the case of an aperiodic message transfer. In the case of a cyclical message transfer, the first DLPDU is not broadcast.

6.8.2.1 Aperiodic message transfer

In the case of an aperiodic message transfer the source entity broadcasts a variable response DLPDU whose control DLPDU contains the request that the bus-arbitrator circulate a message identifier DLPDU (RP_DAT_MSG or RP_DAT_RQi_MSG). The first of these DLPDUs is part of normal cyclical traffic.

The identifier refers to the queue for messages to be emitted that is set aside for aperiodic message transfer.

The bus-arbitrator gives local link control to the source entity during an aperiodic message window by broadcasting a message identifier DLPDU (ID_MSG) with the identifier that has carried the request.

The message is emitted by the source entity (RP_MSG_ACK) which states in the extended control field the message destination and source addresses.

The destination entity immediately emits the acknowledgement DLPDU.

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signals the end of the message transaction (RP_END).

Figure 41 shows the DLPDU sequence as it occurs in the periodic and aperiodic windows:

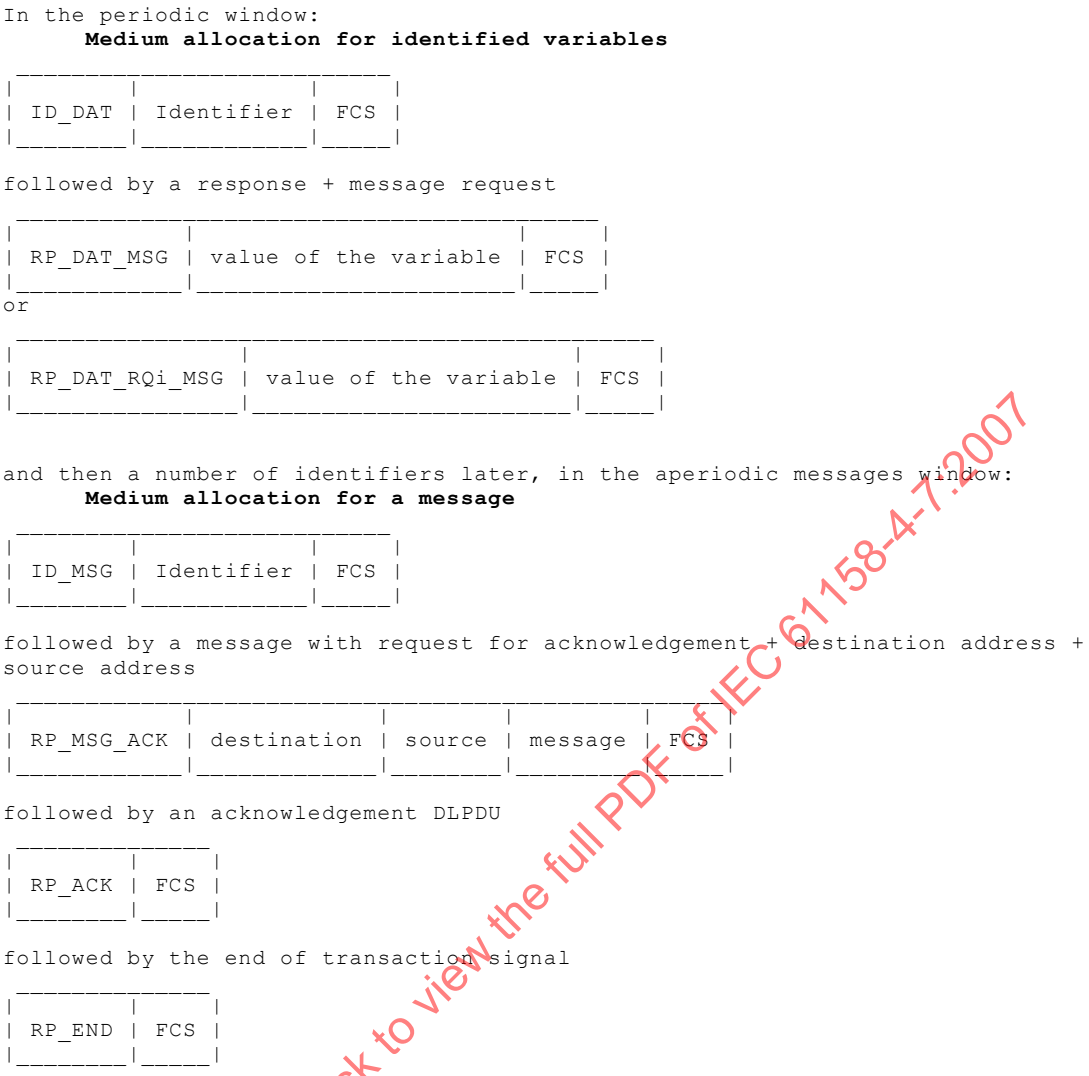


Figure 41 – DLPDU sequence for an aperiodic message transfer

6.8.2.2 Cyclical message transfer

In the case of a cyclical message transfer, the bus-arbitrator gives control to the source entity by broadcasting a message identifier DLPDU (ID_MSG) in the periodic window.

The message is emitted by the source entity (RP_MSG_ACK) which notes in the extended control field the destination and source addresses.

The destination entity emits the acknowledgement DLPDU (RP_ACK) immediately.

The source entity returns control to the bus-arbitrator by emitting a DLPDU that signals the end of the message transaction (RP_END).

Figure 42 shows the DLPDU sequence as it occurs in the periodic window:

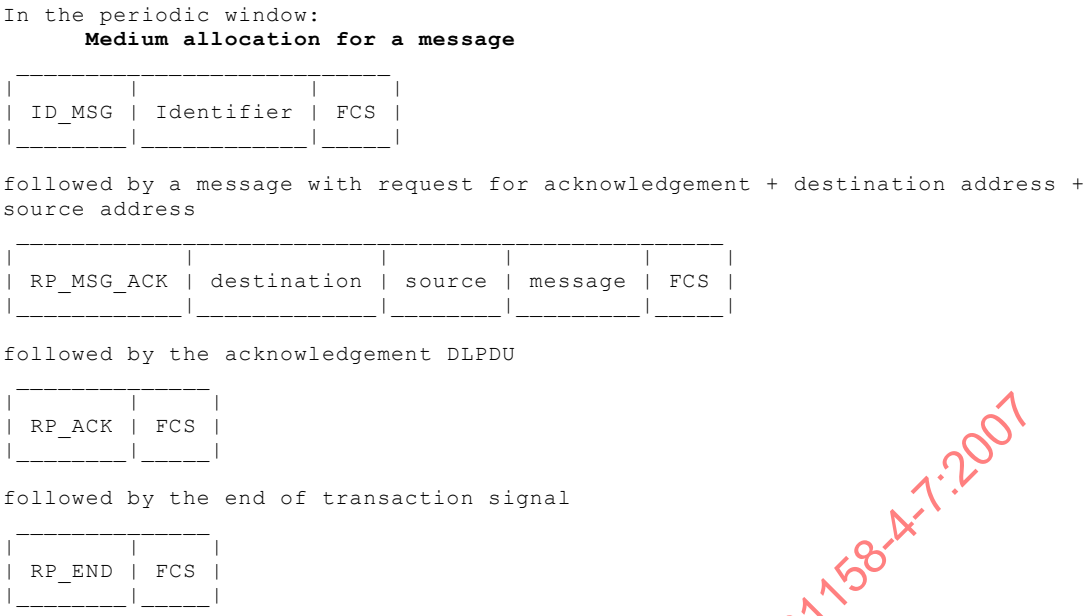


Figure 42 – DLPDU sequence for a cyclical message transfer

The RP_MSG_ACK and RP_ACK control fields containing the numbering bit Bp given by the algorithm defined in 6.9.

NOTE This DL-protocol provides SDA and SDN services as specified in IEC 60955 (PROWAY C). The DLL primitives also directly correspond to primitives specified in the documents ISO/IEC 8802-2 and ISO TC 97 SC6 WG1 N169. The DL-MESSAGE primitive corresponds to the DLSDU primitive. The DL-MESSAGE-ACK primitive corresponds to the DLSDU_ACK primitive.

6.9 Numbering of acknowledged messages

The algorithm describing the message numbering protocol uses the following data:

— each destination entity records the source address (ADRsource_stoc) of the last message response DLPDU received, and the associated even/odd bit (Bc),

— upon reception of any DLPDU other than RP_MSG_ACK, entities give the stored source address the value "empty",

— recovery is immediate (no other intermediate transaction). Recovery is provided by the source entity, which manages a waiting period for the acknowledgement DLPDU and a restart counter,

— the source entity sends the message with the indication Bp, and changes the value of this bit upon transmission of RP_END.

When the system is initialized each entity has the following information:

— ADRsource_stoc = “empty”,

— Bp has no significance,

— Bc has no significance.

The algorithm has three parts. One part is implemented by the message source entity, another by the destination entity, and the third by the bus-arbitrator.

6.9.1 Destination algorithm:

```
Upon reception of RP_MSG_ACK,
compare ADRsource received with ADR source_stoc
if equal compare Bp received with Bc
  if equal
     if there is room to store
        then send RP_ACK+Bp
             store message
             Bc=1-Bp
     else send RP_ACK-Bp
  else send RP_ACK+Bp
       ignore message because duplicate
else
  if there is room to store
     send RP_ACK+Bp
     Bc=1-Bp
     store the message
     ADRsource_stoc=ADR source received
  else send RP_ACK-Bp

if reception of DLPDU other than RP_MSG_ACK
     ADRsource_stoc=empty
```

6.9.2 Source algorithm:

```
Upon reception of ID_MSG(IDk) and IDk recognized by the source entity
send message with Bp indication
wait for RP_ACK

if reception RP_ACK
             set MSG of IDk at 0
             send confirmation with status (ack ±)
             reset restart counter to zero
             send RP_END and Bp=1-Bp

if expiration of waiting time for RP_ACK,
  test restart counter

  if counter > maximum number of restarts authorized
     set MSG of IDk at 0
     reset restart counter to zero
     send confirmation with negative status
     send RP_END and Bp=1-Bp

  if counter ≤ maximum number of restarts authorized
     retransmit same RP_MSG_ACK
     increment restart counter
     wait for RP_ACK
```

6.9.3 bus-arbitrator algorithm:

After sending ID_MSG

The bus-arbitrator is waiting for RP_END:

if the timer for waiting for RP_END has expired

go on to the next identifier

if reception of RP_END

go on to next identifier

if reception of DLPDU other than RP_END, or of an invalid DLPDU

continue to wait for RP_END

6.10 Behavior with mismatched parameters

6.10.1 Message aperiodic transfer (acknowledged or not)

The evaluation network of Figure 43 allows specifying the behavior of a producer/consumer entity in the following cases.

— Reception of an ID_MSG DLPDU containing a DLCEP-identifier configured for aperiodic transfer of messages and not associated with the Q_MSG.aper queue (the identifier MSG has a value of 0).

— Reception of an ID_MSG DLPDU containing a DLCEP-identifier configured for aperiodic transfer of messages and associated with the Q_MSG.aper queue (the identifier MSG has a value of 1), the latter being empty.

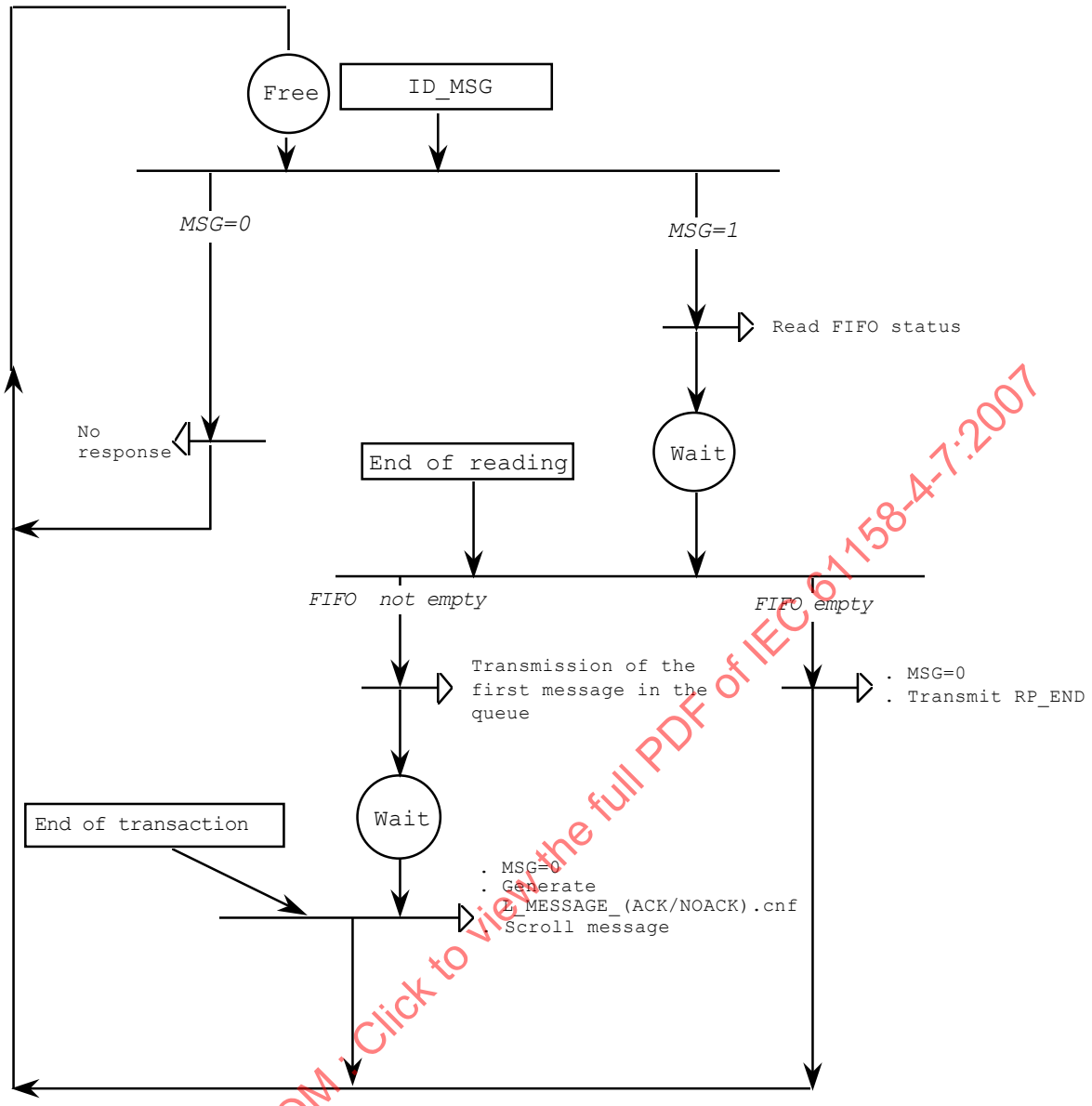


Figure 43 – Evaluation network for message aperiodic transfer

6.10.2 Message cyclic transfer (acknowledged or not)

The evaluation network of Figure 44 allows specifying the behavior of a producer/consumer entity, on reception of an ID_MSG DLPDU containing a DLCEP-identifier configured for the cyclic transfer of messages and whose Q_MSG.cyc queue is empt.

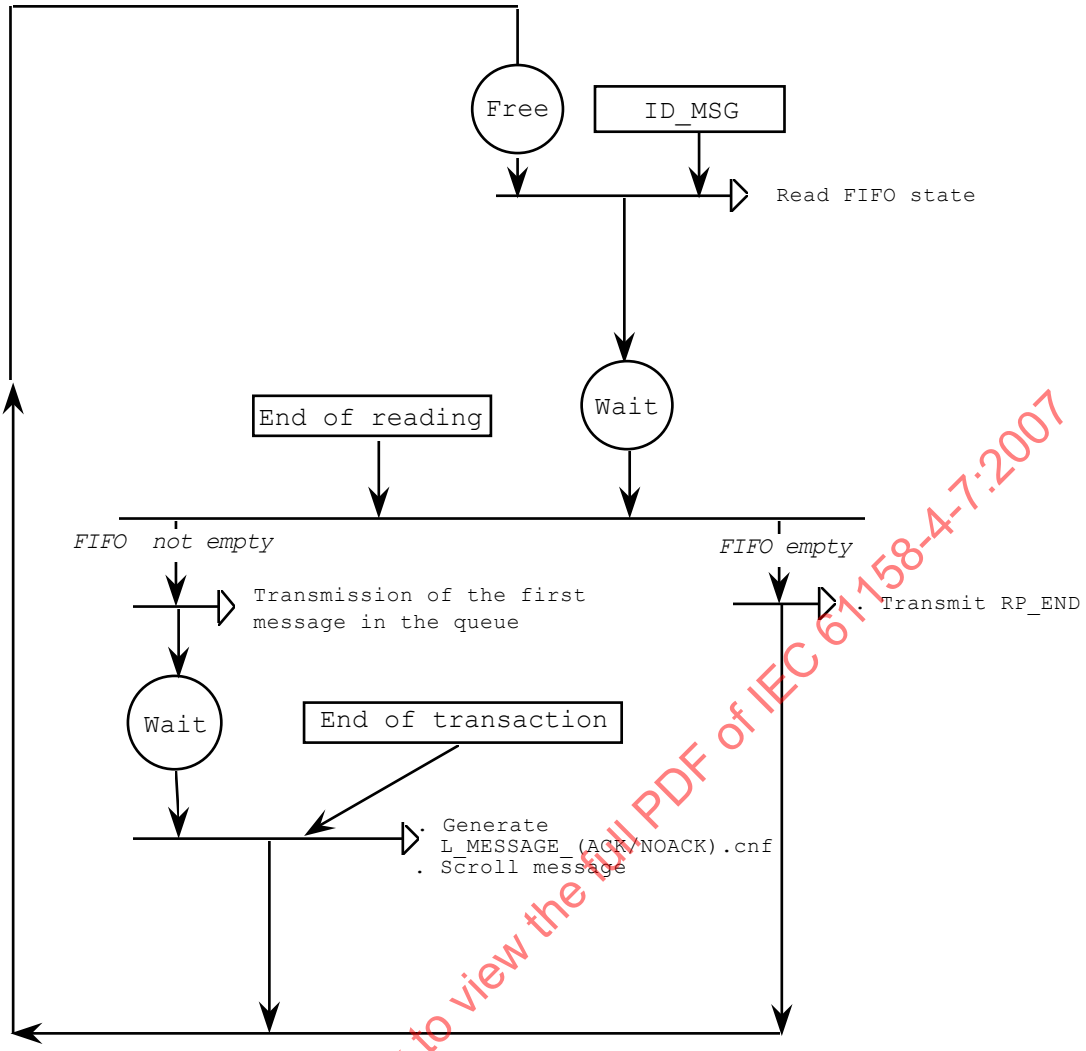


Figure 44 – Evaluation network for message cyclic transfer

7 DL-service elements of procedure, interfaces and conformance

7.1 General

These finished status machines describe the overall functioning of a DLE as a function of the DLPDUs circulating on the medium, and they use the time-fillers defined above.

Transition tables complete the description of the functioning of a DLE by stating actions carried out upon reception of a DLPDU.

The protocols associated with each service provided by the DLL are described for a producer/consumer entity and for the bus-arbitrator.

The interfaces to the DLS-user, DL- Management and the PhL are summarized and the Type 7 conformance classes are shown in Figure 183.

7.2 Producer/consumer entity

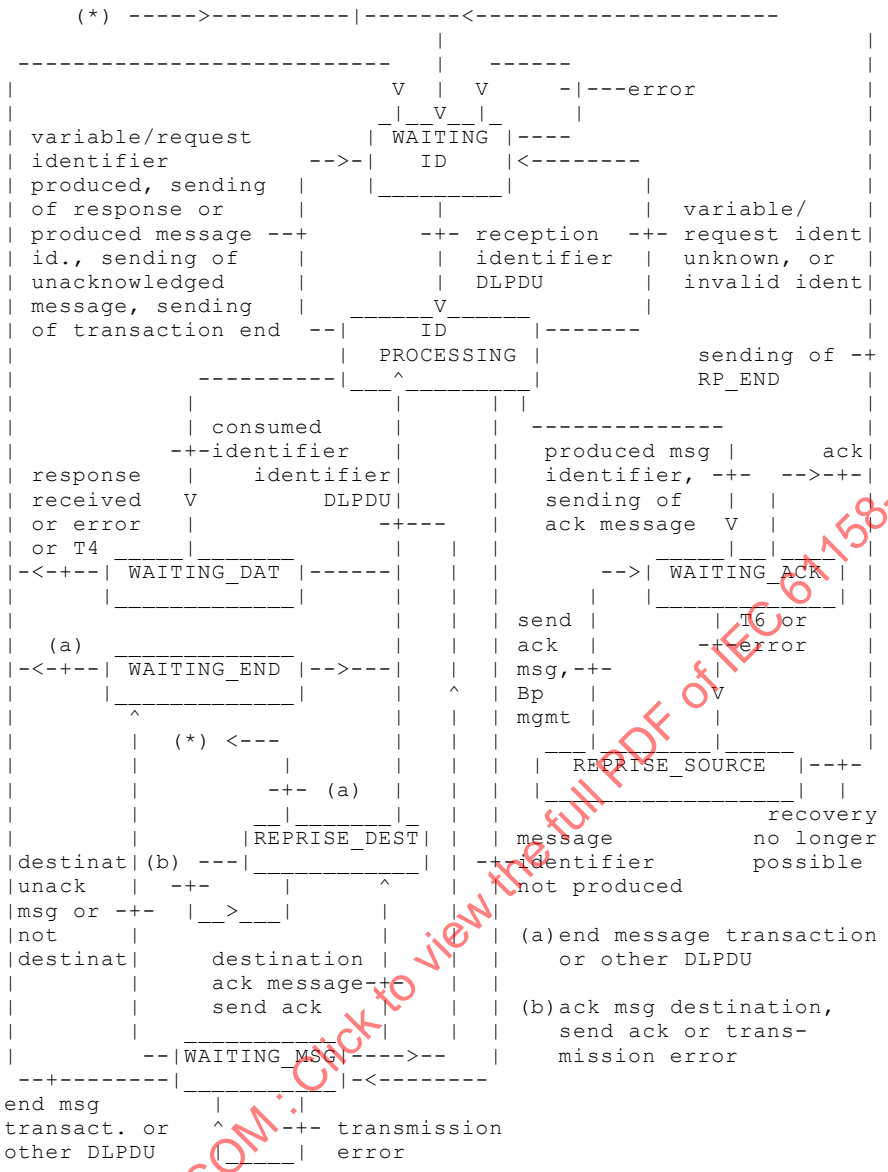


Figure 45 – Simplified states machine for a producer/consumer entity

The status machine indicates the behavior of a station during reception and transmission. This machine takes into account in its various statuses:

— reception of various types of link DLPDU,

— time-outs specific to the station (they are not those of the BA).

It is important to note that these various link DLPDUs taken into account in the various statuses of this machine can only be taken into account in so far as the latter at least observe the temporal characteristics of the protocol.

a) Temporal characteristics of the protocol without interruption.

In the case of normal sequencing of the protocol, the temporal characteristics of the latter are fixed by the turn-around time value during production of the various stations. Each DLPDU received by a device which must provide a response is separated from the latter on the bus by a time interval equal to its own turn-around time during production.

NOTE Knowing that the device which has the maximum turn-around time during production conditions the choice of the time-out T0.

b) Temporal characteristics of the protocol in case of interruption

If the DLPDU sequencing is degraded for whatever reason (disturbance on the bus, absence of production) the temporal characteristics of the protocol are fixed by the bus arbitrator.

Thus for example, an ID_XX DLPDU shall be taken into account in the various reception statuses only if the BA observes the following temporal constraints.

— In the WAIT_DAT status, the BA must wait at least T1=T0 before proceeding in the transmission of an ID_XX. Knowing that time-out T1 on the BA level is set by the line silence following transmission of a n ID_DAT DLPDU and that it can elapse only if silence is maintained all through its life.

— In the MSG_WAIT or END_WAIT or DEST_RESTART, the BA must attain at least T5=2T0, in case of non-reception of RP_END before proceeding with transmission of an ID_XX DLPDU. Knowing that time-out T5 at the BA level is set by each line silence following transmission of an ID_MSG DLPDU and that it can only elapse if the silence is maintained all through its life.

NOTE In the case of an arbitrator which has identified the cause of this interruption (for example, producer absent) and which would like to turn around immediately to pass to transmission of the following identifier, operation of the protocol is not guaranteed.

7.2.1 Transitions table for a producer/consumer entity

```
WAITING_ID (initial status)
   ID_xx:                                                --> PROCESSING_ID

   other DLPDU:                                            --> WAITING_ID
   transmission error:                                     --> WAITING_ID


PROCESSING_ID
   ID_DAT,ID produced:
      if the identifier is invalid, no response
                                                           --> WAITING_ID

      if indicator of urgent explicit request emit RP_DAT_RQ1
      if indicator of normal explicit request emit RP_DAT_RQ2
      if indicator of message request emit RP_DAT_MSG
      if 2 indicators emit RP_DAT_RQi_MSG
      else emit RP_DAT

      generate a DL-SENT indication (ID produced)          --> WAITING_ID

   ID_DAT,ID consumed: activate T4                        --> WAITING_DAT
   ID_DAT, new ID:                                         --> WAITING_ID

   ID_MSG,ID produced:
      if msg exists and is unack, emit RP_MSG_NOACK
                                  emit RP_END              --> WAITING_ID

      if msg exists and is ack, emit RP_MSG_ACK
                                activate T6               --> WAITING_ACK

      if msg does not exist, emit RP_END                   --> WAITING_ID

   ID_MSG, new ID:                                        --> WAITING_MSG

   ID_RQ1,ID produced:
      emit RP_RQ1
      generate DL-SPEC_UPDATE confirm (ID produced, OK)
      or DL-FREE_UPDATE confirm (OK)                       --> WAITING_ID

   ID_RQ1, new ID:                                         --> WAITING_ID

   ID_RQ2, ID produced:
      emit RP_RQ2
      generate DL-FREE_UPDATE confirm (OK)                 --> WAITING_ID

   ID_RQ2, new ID:                                         --> WAITING_ID
```

```
WAITING_DAT
   RP_DAT or RP_DAT_MSG or RP_DAT_RQi or RP_DAT_RQi_MSG:
      store value of the variable
      generate a DL-RECEIVED indication (ID consumed)
                                                          --> WAITING_ID

   ID_xx:                                               --> PROCESSING_ID

   other DLPDU:                                           --> WAITING_ID
   transmission error:                                    --> WAITING_ID
   T4 timer: indicate transaction aborted                 --> WAITING_ID


WAITING_MSG
   RP_MSG_NOACK, my address is destination:
      receive message
      generate a DL-MESSAGE indication (ADRdestination, ADRsource,
      message)                                            --> WAITING_END

   RP_MSG_NOACK,other destination address:                --> WAITING_END

   RP_MSG_ACK, my address is destination:
      generate message numbering bit
      receive msg, send RP_ACK
      generate a DL-MESSAGE_ACK indication (ADRdestination, ADRsource,
      message)                                            --> RECOVERY_DEST
   RP_MSG_ACK, other destination address                  --> WAITING_END

   RP_END: end of message transaction                     --> WAITING_ID

   ID_xx:                                               --> PROCESSING_ID

   other DLPDU:                                           --> WAITING_ID
   transmission error:                                    --> WAITING_MSG


WAITING_ACK
   RP_ACK:
      generate the message numbering bit (see 6.9)
      generate a DL-MESSAGE_ACK confirm
      (ADRdestination, ADRsource, status)
   emit RP_END                                            --> WAITING_ID

   T6 timer:                                          --> RECOVERY_SOURCE

   other DLPDU:                                       --> RECOVERY_SOURCE
   transmission error:                                --> RECOVERY_SOURCE


WAITING_END
   RP_END:                                                --> WAITING_ID

   ID_xx                                              --> PROCESSING_ID

   other DLPDU:                                           --> WAITING_ID
   transmission error:                                    --> WAITING_ID


RECOVERY_DESTINATION
   RP_END:                                                --> WAITING_ID

   RP_MSG_ACK: management of Bp (see 6.9)
               send ack                               --> RECOVERY_DEST

   ID_xx                                              --> PROCESSING_ID

   other DLPDU:                                           --> WAITING_ID
   transmission error:                                --> RECOVERY_DEST


RECOVERY_SOURCE
   if recovery possible
      emit same RP_MSG_ACK
      activate T6                                         --> WAITING_ACK
   else send RP_END
      indicate transaction aborted                        --> WAITING_ID
```

NOTE Types of errors are defined in 4.6.22 (DLL-counter class description) of EN 50170-7.

7.3 Protocol elements by service

NOTE The protocols associated with the services provided by a producer/consumer entity are specified by describing the entity's behavior for each primitive exchanged with the DLS-user.

7.3.1 Reception of a DL-PUT request

Upon reception of a DL-PUT request primitive (IDk, DLSDU) a producer triggers the following actions:

* semantic testing (unknown identifier or incorrect length)

— semantics incorrect

immediate transmission of DL-PUT confirm (IDk, unknown identifier or length of DLSDU incorrect)

— semantics correct

* test validity of identifier

— identifier invalid

immediate transmission of DL-PUT confirm (IDk, invalid identifier)

— identifier valid

* test availability of buffer

— buffer unavailable,

immediate transmission of DL-PUT confirm (IDk, buffer unavailable)

— buffer available,

write the new value

immediate transmission of DL-PUT confirm (IDk, OK)

The various tests carried out are based on data concerning:

— recognition of identifier parameter,

— the identifier is known if it has been declared as a produced identifier by the producer/consumer entity,

— compatibility of length of DLSDU with the buffer B_DATprod (length less than 128 octets),

— validity of the identifier,

— a DLCEP-identifier is valid if the data-link configuration of the associated variable is valid,

— availability of buffer,

— buffer unavailable if the same B_DATprod buffer is already being accessed by the local link. To manage this conflict the standard recommends giving the local link highest priority in all cases.

7.3.2 Reception of a DL-GET request

Upon reception of a DL-GET request primitive (IDk) a producer triggers the following actions:

* test identifier IDk

— identifier unknown

immediate transmission of DL-GET confirm (IDk, identifier unknown)

— identifier known

* test availability of buffer

— buffer unavailable,

immediate transmission of DL-GET confirm (IDk, buffer unavailable)

— buffer available

* test validity of identifier

— identifier invalid

immediate transmission of DL-GET confirm (IDk, invalid identifier)

— identifier valid

immediate transmission of DL-GET confirm (IDk, DLSDU, reading OK).

The various tests carried out are based on data concerning:

— recognition of identifier parameter

the identifier is known if it has been declared as a consumed identifier by the producer/consumer entity,

— availability of buffer

buffer unavailable if the same B_DATcons buffer is already being accessed by the local link. To manage this conflict the standard recommends giving the local link highest priority in all cases,

— validity of the identifier,

— a DLCEP-identifier is valid if the data-link configuration of the associated variable is valid.

7.3.3 Reception of a DL-SPEC-UPDATE request

Upon reception of a DL-SPEC-UPDATE request primitive (IDk, LIST) a producer triggers the following actions:

* test IDk

— IDk unknown or not configured for this service

immediate transmission of DL-SPEC-UPDATE confirm (IDk, identifier unknown)

— IDk known and configured for this service

* test availability of B_REQ (IDk) buffer

— buffer unavailable

immediate transmission of DL-SPEC-UPDATE confirm (IDk, buffer unavailable)

— buffer available

* test transmission of preceding request

— request not transmitted (RQ of IDk=1)

. immediate transmission of DL-SPEC-UPDATE confirm (IDk, override) of the preceding request

. store list of identifiers (this list can be empty)

— request transmitted (RQ of IDk=0)

. set request indicator (RQ of IDk=1)

. store list of identifiers

* reception of ID_DAT (IDk) produced

* test RQ of IDk

— RQ of IDk inhibited by configuration

emit only RP_DAT

— RQ of IDk not inhibited by configuration

emit RP_DAT_RQ

* reception of ID_RQ (IDk)

. emit RP_RQ with list of identifiers requested

. generate DL-SPEC-UPDATE confirm (IDk, OK)

. set RQ of IDk to 0

The various tests carried out are based on data concerning:

— availability of buffer

buffer unavailable if B_REQ is already being accessed by the local link,

— waiting for the request to be taken into account by the bus-arbitrator

RQ of IDk is set at 1 by DL-SPEC-UPDATE request

RQ of IDk is set at 0 upon transmission of RP_RQ.

A specific transfer request that contains an empty variables list makes it possible to clear a buffer of specific transfer requests.

7.3.4 Reception of a DL-FREE-UPDATE

Upon reception of a DL-FREE-UPDATE request primitive (LIST, PRIORITY) a producer triggers the following actions:

* test queue

— sufficient space not available,

immediate transmission of DL-FREE-UPDATE confirm (priority, saturation)

— space available and queue not empty

store list of identifiers

— queue empty

. store list of identifiers

. wait for next produced ID_DAT

* reception of produced ID_DAT (IDp)

* test IDp

— IDp not authorized

wait for next produced ID_DAT

— IDp authorized

. associate IDp with the queue concerned

. set RQ of IDp at 1

. send RP_DAT_RQ

* reception of ID_RQ (IDp)

* test status of queue

— queue empty no answer

— queue not empty

. emit RP_RQ with all stored lists of identifiers (up to a limit of 64 identifiers)

. generate as many DL-FREE-UPDATE confirm (OK) primitives as there are lists stored

. run through the lists of identifiers

. set RQ of IDp at 0

* test queue status

— queue not empty

wait for next produced ID_DAT

The various tests carried out are based on data concerning:

— status of the queue Q_REQi: queue filled or not, queue empty or not

wait for an authorized produced ID_DAT for DL-FREE-UPDATE. A mechanism makes it possible to prohibit the use of certain identifiers for DL-FREE-UPDATE.

7.3.5 Reception of a DL-MESSAGE request

Upon reception of a DL-MESSAGE request primitive (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, message) a source entity triggers the following actions:

* test SPECIFIED_IDENTIFIER

— incorrect parameter (identifier not configured for cyclical message transfer, or value of parameter different from NIL)

immediate transmission of DL-MESSAGE confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, incorrect parameter)

— parameter correct

* test status of queue

— queue full

. immediate transmission of DL-MESSAGE confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, queue full)

— queue not full

. store message

. wait for ID_MSG (IDp) or ID_DAT (IDp)

* reception of ID_DAT (IDp)

* test IDp

— IDp configured for cyclical message transfer

. emit RP_DAT

. wait for next produced ID_DAT

— IDp configured for aperiodic message transfer

* test MSG of IDp

- MSG of IDp = 1

. emit RP_DAT_MSG

. wait for next ID_DAT

— MSG of IDp = 0

* test status of the resource Q_MSGaper set aside for aperiodic message transfer

- queue empty

. emit RP_DAT

- queue not empty and number of identifiers associated with the resource equal to the number of places occupied in said resource

. emit RP_DAT

- queue not empty and number of identifiers associated with the resource less than number of places occupied in the resource

. establish reference between IDp and the queue for messages waiting to be emitted set aside for aperiodic message transfer

. set MSG of IDp at 1

. emit RP_DAT_MSG

. wait for next ID_DAT

* reception of ID_MSG (IDp)

* test IDp

— IDp shows no reference to a resource

. no response

— IDp shows reference to a resource

* test status of resource given by IDp

- if resource is empty, send RP_END

— else

. emit RP_MSG_NOACK with the first message of the queue given by IDp

. send RP_END

. send DL-MESSAGE confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, message sent)

. free up place in queue

. if the resource is the queue set aside for aperiodic message transfer:

. set MSG of IDp at 0

. cancel reference between IDp and the queue.

The various tests carried out are based on data concerning:

— status of the queue used: empty, full, or space available,

— indicator showing whether the bus-arbitrator has taken the message request into account in the case of aperiodic message transfer

MSG is set at 1 through attribution of IDp to the queue set aside for aperiodic message transfer.

MSG is set at 0 upon emission of RP_END.

Number of identifiers associated with the queue set aside for aperiodic message transfer, and number of places occupied in this queue.

7.3.6 Receipt of a DL-MESSAGE-ACK request

Upon reception of a DL-MESSAGE-ACK request primitive (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, message) a source entity triggers the following actions:

* test SPECIFIED_IDENTIFIER

—incorrect parameter (identifier not configured for cyclical message transfer, or value of parameter different from NIL) immediate transmission of DL-MESSAGE-ACK confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, incorrect parameter)

parameter correct

*test ADR Destination

ADR destination is a group address

. ignore the message

. immediate transmission of DL-MESSAGE-ACK confirm(Spec Id, ADR Dest, ADR Source, Group Address)

* test status of queue

— queue full

. immediate transmission of DL-MESSAGE-ACK confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, queue full)

— queue not full

. store message

. wait for ID_MSG (IDp) or ID_DAT (IDp)

* reception of ID_DAT (IDp)

* test IDp

— IDp configured for cyclical message transfer

. emit RP_DAT

. wait for next produced ID_DAT

— IDp configured for aperiodic message transfer

* test MSG of IDp

— MSG of IDp = 1

. emit RP_DAT_MSG

. wait for next ID_DAT

— MSG of IDp = 0

* test status of the resource Q_MSGaper set aside for aperiodic message transfer

- queue empty

. emit RP_DAT

- queue not empty and number of identifiers associated with the resource equal to the number of places occupied in said resource

. emit RP_DAT

- queue not empty and number of identifiers associated with the resource less than number of places occupied in the resource

. establish reference between IDp and the queue for messages waiting to be emitted set aside for aperiodic message transfer

. set MSG of IDp at 1

. emit RP_DAT_MSG

. wait for next ID_DAT

* reception of ID_MSG (IDp)

* test IDp

— IDp shows no reference to a resource

. no response

— IDp shows reference to a resource

* test status of resource given by IDp

— if resource is empty, send RP_END

— else

. emit RP_MSG_ACK with the first message of the queue referenced by IDp

. launch timer for waiting for RP_ACK

. wait for acknowledgement DLPDU

* reception of RP_ACK

* test acknowledgement

- acknowledgement positive

. send RP_END

. send DL-MESSAGE-ACK confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, message sent)

. free up place in queue

. if the resource is the queue set aside for aperiodic message transfer:

. set MSG of IDp at 0

. cancel reference between IDp and the queue.

— acknowledgement negative

. send RP_END

. send DL-MESSAGE-ACK confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, acknowledgement negative)

. free up place in queue

. if the resource is the queue set aside for aperiodic message transfer:

. set MSG of IDp at 0

. cancel reference between IDp and the queue.

* expiration of timer for acknowledgement reception

* test restart counter

—- the counter has not reached its maximum value

— emit RP_MSG_ACK with the first message in the queue referenced by IDp

— launch the timer for waiting for RP_ACK

— wait for acknowledgement DLPDU

—- the counter has reached its maximum value

— send RP_END

— send DL-MESSAGE-ACK confirm (SPECIFIED_IDENTIFIER, ADRdestination, ADRsource, maximum number of restarts attained)

— free up place in queue

— if the resource is the queue set aside for aperiodic message transfer:

— set MSG of IDp at 0

— cancel reference between IDp and the queue.

The various tests carried out are based on data concerning:

— status of the queue used: empty, full, or space available,

— indicator showing whether the bus-arbitrator has taken the message request into account in the case of aperiodic message transfer MSG is set at 1 through attribution of IDp to the queue set aside for aperiodic message transfer MSG is set at 0 upon emission of RP_END,

— current value of the restart counter

number of identifiers associated with the queue set aside for aperiodic message transfer, and number of places occupied in this queue.

7.3.7 Invalid identifier

This part of standard define the actions to be performed at reception of an ID_XX DLPDU concerning an invalid identifier.

The invalidation of a DLCEP-identifier concerns at least one of the four services: cyclic messaging, specified explicit periodic request for transfer of identified objects, consumption, production. In all cases, a producer/consumer entity does not answer the demands from the medium which concern the invalidated service(s) of a DLCEP-identifier.

Besides, at invalidation of a DLCEP-identifier in production (Product_ID_Validity attribute set to FALSE), the possible associations with Q_REQ1 or Q_REQ2 as well as with Q_MSG.aper are interrupted. The attachment attribute relative to Q_REQ1 or Q_REQ2 is set to FALSE if the identifier was associated with one of these files before being invalidated; the Attachment_number attribute relative to Q_MSG.aper is decreased by one if the identifier was associated with this file before being invalidated.

7.4 Bus arbitrator operation

7.4.1 General description of the bus arbitrator

The role of the bus-arbitrator is to "give permission to speak" to each information producer, while taking into account the services needed by the user application.

The bus-arbitrator thus has three functions:

1) periodic scanning of variables or triggered periodic scanning of variables and messages,
2) triggered scanning of variables,
3) triggered scanning of messages.

In addition, the bus-arbitrator can provide a synchronization function to guarantee the constant length of a scanning cycle.

Each type of scanning takes place in a periodic window, an aperiodic variables window, an aperiodic messages window, or a synchronization window respectively.

The four windows together make up a basic scanning cycle.

The functions of the bus-arbitrator are broken down into three layers, as are other DL-services. For each of these layers there is a set of arbitrating services and arbitrating data. The complete specification of the bus-arbitrator will be found in 5.1 (bus-arbitrator) of EN 50170-3.

The remainder of this subclause describes the data-link functions provided by the bus-arbitrator. The data-link level of a bus-arbitrator entity offers the following functions:

a) emission of identifier DLPDU sequences,

b) reception of DLPDUs containing identifiers,

c) detection of requests for aperiodic variable and message exchanges,

d) emission of identifier DLPDUs used for synchronization.

The resources used to provide these functions are also described: they include the identifier sequences, queues for requests for aperiodic variable and message exchanges, the restart buffer.

The bus-arbitrator's data-link level oversees the proper outcome of the various basic transactions, and executes requests from the DLS-user for services, in order to properly carry out Basic Cycles and Macro Cycles.

7.4.2 Management of basic transactions

7.4.2.1 General

A basic transaction is made up of the successive DLPDUs related to a single service.

Managing basic transactions means controlling the succession of DLPDUs that make up a transaction. Valid basic transactions are:

ID_DAT + RP_DAT

ID_DAT + RP_DAT_MSG

ID_DAT + RP_DAT_RQi

ID_DAT + RP_DAT_RQi_MSG

ID_MSG + RP_MSG_NOACK + RP_END

ID_MSG + RP_MSG_ACK + RP_ACK + (repetitions) + RP_END

ID_RQi + RP_RQi

Subclause 7.4.6 specifies the bus-arbitrator's simplified states machine as well as the associated transitions table.

7.4.2.2 Emitting a variable identifier

After emitting a variable identifier DLPDU (ID_DAT) the bus-arbitrator activates the timer T1 and then waits for a variable response DLPDU.

a) If the DLPDU received is a simple response (RP_DAT) the bus-arbitrator goes on to the next identifier to be scanned.

b) If the DLPDU received is a variable response accompanied by a message transfer request (RP_DAT_MSG), and if the bus-arbitrator is in the cyclical scanning phase, it records the identifier that carried the request in the queue for message transfer requests (Q_IDMSG) and goes on to the next identifier to be scanned.

c) If the DLPDU received is a variable response accompanied by an explicit request for a buffer transfer (RP_DAT_RQi), and if the bus-arbitrator is in the cyclical scanning phase, it records the identifier that carried the request in the queue for explicit buffer transfer requests that has the proper priority (Q_IDRQi) and goes on to the next identifier to be scanned.

d) If the DLPDU received is a variable response accompanied by a request for a message transfer and an explicit request for a buffer transfer (RP_DAT_RQi_MSG) and if the bus-arbitrator is in the cyclical scanning phase, it records the identifier that carried the request in the queue for message requests (Q_IDMSG) and in the queue for explicit buffer transfer requests that has the proper priority (Q_IDRQi) and goes on to the next identifier to be scanned.

e) If the DLPDU received is of a different type the bus-arbitrator detects an error and goes on to the next identifier to be scanned.

f) If the DLPDU received has a flawed FCS the bus-arbitrator detects a transmission error and goes on to the next identifier to be scanned.

g) If the T1 timer expires the bus-arbitrator detects the loss of a response DLPDU and goes on to the next identifier to be scanned.

7.4.2.3 Emitting a message identifier

After emitting a message identifier DLPDU (ID_MSG), the bus-arbitrator activates the T5 timer and waits for the response DLPDU indicating the end of the message transaction.

a) If the DLPDU received is an end of message transaction the bus-arbitrator goes on to the next identifier to be scanned.

b) If the DLPDU received is an unacknowledged message (RP_MSG_NOACK), an acknowledged message (RP_MSG_ACK) or an acknowledgement response DLPDU (RP_ACK), the bus-arbitrator activates T5 and continues to wait for a response DLPDU indicating the end of the message transaction.

c) If the DLPDU received is of a different type the bus-arbitrator detects an error and goes on to the next identifier to be scanned.

d) If the DLPDU received has an incorrect residual FCS (see 5.2.5.2) the bus-arbitrator detects a transmission error and continues to wait for the end of message transaction DLPDU.

e) If the timer expires the bus-arbitrator detects the loss of an end of message transaction DLPDU and goes on to the next identifier to be scanned.

7.4.2.4 Emitting a request identifier

After emitting a request identifier DLPDU (ID_RQi), the bus-arbitrator activates the T1 timer and waits for a request response DLPDU.

a) If the DLPDU received is an urgent request response DLPDU (RP_RQ1) and if the bus-arbitrator is in its periodical scanning phase, it records the list of identifiers in the recovery buffer and immediately scans all the variable identifiers stored.

b) If the DLPDU received is an urgent request response DLPDU and if the bus-arbitrator is in its aperiodic scanning phase, it records the list of identifiers in the aperiodic queue being processed and goes on to the next identifier to be scanned.

c) If the DLPDU received is a normal request response DLPDU (RP_RQ2) the bus-arbitrator records the list of identifiers in the aperiodic queue being processed and goes on to the next identifier to be scanned.

d) If the DLPDU received is of a different type the bus-arbitrator detects an error and goes on to the next identifier to be scanned.

e) If the DLPDU received has a flawed FCS the bus-arbitrator detects a transmission error and continues to wait for the end of message transaction DLPDU.

f) If the timer T1 expires the bus-arbitrator detects the loss of a response DLPDU and goes on to the next identifier to be scanned.

7.4.2.5 Synchronization window

During the synchronization window the bus-arbitrator goes through a loop composed of the following actions.

a) It emits a synchronization identifier DLPDU (this DLPDU corresponds, for example, to a DLCEP-identifier that is neither produced nor consumed) and activation of the T1 timer (used for waiting for a response DLPDU).

b) If the bus-arbitrator receives a response DLPDU or detects the expiration of the T1 timer, it restarts loop operations.

c) If the bus-arbitrator detects the expiration of the T2 timer signifying the end of the Basic Cycle, it abandons the loop and goes on to the next Basic Cycle.

This function guarantees the constant length of a Basic Cycle. The precise timing of the beginning of a cycle is thus at best equal to the length of the emission of a DLCEP-identifier DLPDU followed by a timer used to wait for a response DLPDU.

7.4.3 Resources used

The DLL uses two categories of resources to implement the data-link level functions of the bus-arbitrator:

— the basic sequences of identifier numbers that constitute the cyclical portion of the scanning table and make it possible to generate the bus's periodic flow,

— the queues used to store the numbers of identifiers that have carried a normal (Q_IDRQ2) or urgent (Q_IDRQ1) variable request,

— the queues used to store the numbers of identifiers that are the object of aperiodic variable exchange requests (Q_RPRQ),

(These four queues make it possible to generate the bus's aperiodic flow.)

— a recovery buffer for the immediate emission of identifiers involved in a triggered periodic request for a variable exchange,

— a basic padding sequence used to generate synchronization window flow.

For further details, see 4.2.2.

7.4.4 Chaining of basic transactions

7.4.4.1 Bus arbitrator cycles

The basic function of the bus-arbitrator is to grant access to the bus by emitting identifier DLPDUs.

Several types of identifier DLPDUs have been defined, and a hierarchy in the order of emission of these identifiers can be set up to accommodate specific user needs.

The bus-arbitrator's highest priority function is to allocate the medium so that the cyclical transfer of variables, requests, and messages is guaranteed.

Allocation of the medium for explicit requests for variable or message transfers has a lower priority.

Screening time requirements are not identical for all the variables scanned cyclically, but are always multiples of a basic period.

A Basic Cycle includes the bus-arbitrator's scanning of:

— a set of variable, request, and cyclical message identifiers based on the application. This set of identifiers characterizes the individual Basic Cycle,

— plus the aperiodic exchanges window,

— plus the message services window,

— plus the synchronization window.

A Macro Cycle is the group of Basic Cycles that are juxtaposed to obtain the scanning of all the application's cyclical identifiers.

The Macro Cycle covers all the application's cyclical communications needs, and provides an available bandwidth for triggered transactions.

A Basic Cycle thus has a maximum of 4 phases:

P1: periodic scanning of variables (cyclical ID_DAT), triggered periodic scanning of variables (cyclical ID_RQi) or triggered periodic scanning of messages (cyclical ID_MSG)

P2: triggered message scanning

P3: triggered scanning of variables with management of the two priority levels

P4: wait for synchronization signal if a synchronized scanning cycle is requested.

Restart is possible in P1 and P2.

P3 can precede P2.

A Macro Cycle includes at least one Basic Cycle.

All Basic Cycles include a P1 (periodic scanning) phase.

Chaining of cycles

A DLL system incorporates several variable-scanning periods, but bus-arbitrator operations include only two levels of chaining:

— chaining of sequences to make up a Basic Cycle,

— chaining of Basic Cycles to make up a Macro Cycle.

In addition, chaining of basic cycles can be

— with synchronization,

— without synchronization.

Consequently, the DLS-user sends the bus-arbitrator a set of service request primitives that describe the execution of the Basic Cycles (the flow unit on the bus) and the Macro Cycles (the repetitious unit on the bus, made up of the chaining of one or several Basic Cycles).

These primitives are used to express user needs both in terms of the information to be exchanged and in terms of the screening time constraints to be respected.

If the synchronized scanning function has not been requested, Phase P4 of the Basic Cycle is eliminated. The maximum length of the Basic Cycle can still be guaranteed, however. In this operating mode if the Basic Cycle continues beyond its maximum length the next Basic Cycle is automatically begun.

7.4.4.2 Cycle configuration parameters

Certain configuration parameters are connected with each phase of the Basic Cycle.

Parameters linked to phase P1 of a Basic Cycle are

— the list of basic sequences that characterize the basic cycle. These identifiers will be scanned using a variable, request, or cyclical message identifier (ID_DAT, ID_RQ1 or ID_MSG).

In the case of a cyclical request transaction (ID_RQ1, RP_RQ1) the number of transactions (ID_DAT, RP_DAT) that immediately follow is limited by the entity making the request and is known in advance. Thus a maximum length for the transaction is guaranteed.

The parameter linked to phase P2 of a Basic Cycle is

— the time limit on the window allocated to triggered message transfers. This limit is defined with respect to the beginning of the Basic Cycle. When this window exists, it corresponds at minimum to the transmission of an ID_MSG DLPDU, followed by an RP_MSG_NOACK DLPDU and a RP_END DLPDU. If acknowledged message transfers and a restart mechanism are supported the window corresponds to the transmission of an ID_MSG DLPDU, followed by a RP_MSG_ACK DLPDU, followed by a RP_ACK DLPDU and enough time for as many repetitions of RP_MSG_ACK, RP_ACK as the number of authorized restarts allows. All these DLPDUs are followed, of course, by an RP_END DLPDU.

The parameter linked to phase P3 of a Basic Cycle is

— the time limit on the window allocated to triggered aperiodic variable exchanges. This limit is defined with respect to the beginning of the Basic Cycle. When this window exists, it corresponds at minimum to the transmission of an ID_RQi DLPDU, followed by an RP_RQi DLPDU.

The parameter linked to phase P4 of a Basic Cycle is

— the total length of a Basic Cycle and the basic padding sequence emitted while waiting for the indication signaling the end of the Basic Cycle.

7.4.5 Multiple bus arbitrators

The information presented in this subclause is further developed in the document C46 605 Network Management.

The overall results given below are based on the hypothesis that the multiple bus-arbitrators are connected to a single physical support:

— at a given instant ONLY ONE BA is active,

— a procedure for choosing the active BA is defined,

— the other BA listen to what is occurring on the physical support and manage their own scanning tables, but do not emit any identifier DLPDUs,

— a timer value is specified in the final states machine of each producer/consumer entity in order to detect any prolonged silence on the part of the active BA (timer T3),

— a protocol for the exchange of synchronization data can be used to synchronize Macro Cycles.

7.4.6 Bus arbitrator

A simplified state machine for the active bus-arbitrator is shown in Figure 46 and Table 6.

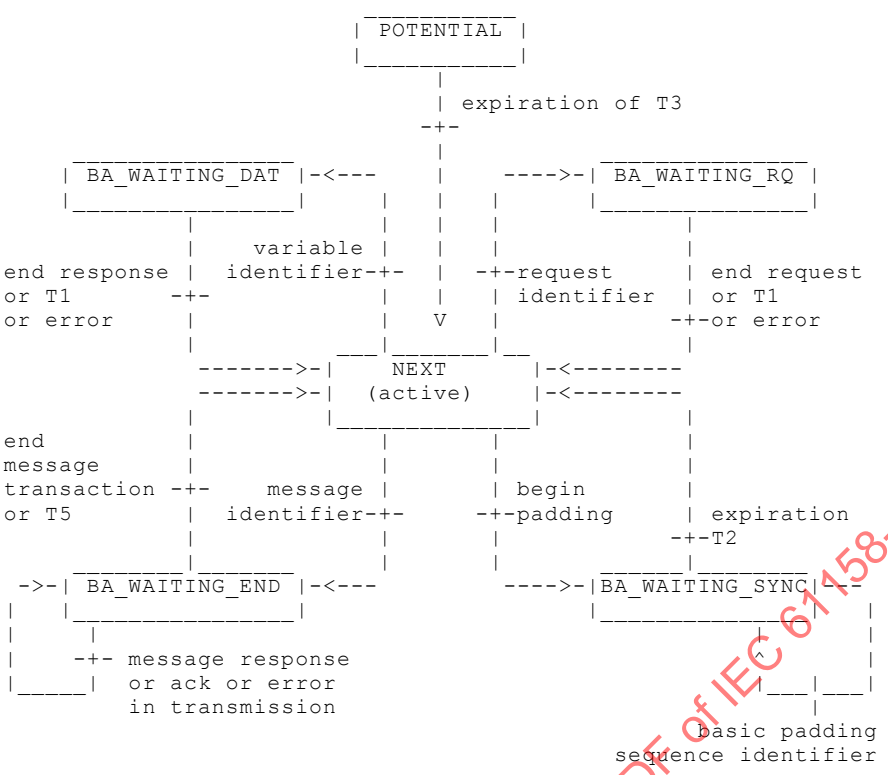


Figure 46 – Active bus arbitrator's simplified state machine

Table 6 – Bus arbitrator state transition table

Current state Event	Action	Next state
(initial)		
expiration T3		--→ NEXT
NEXT		
emission ID_DAT	activate T1	--→ BA_WAITING_DAT
emission ID_RQ1	activate T1	--→ BA_WAITING_RQ
emission ID_RQ2	activate T1	--→ BA_WAITING_RQ
emission ID_MSG	activate T5	--→ BA_WAITING_END
emission basic padding sequence		--→ BA_WAITING_SYNC
BA_WAITING_DAT		
RP_DAT		--→ NEXT
RP_DAT_MSG	record ID message emitter	--→ NEXT
RP_DAT_RQ1	record ID request emitter	--→ NEXT
RP_DAT_RQ2	record ID request emitter	--→ NEXT
RP_DAT_RQ1_MSG	record ID request emitter and ID message emitter	--→ NEXT
P_DAT_RQ2_MSG	record ID request emitter and ID message emitter	--→ NEXT
other DLPDU		--→ NEXT
transmission error		--→ NEXT
T1 timer	transaction aborted	--→ NEXT
BA_WAITING_END		
RP_END		--→ NEXT
RP_MSG_NOACK, RP_MSG_ACK, RP_ACK	activate T5	--→ BA_WAITING_END
other DLPDU		--→ NEXT
transmission error		--→ BA_WAITING_END
T5 timer	transaction aborted	--→ NEXT
BA_WAITING_RQ		
RP_RQ1	record list of identifiers in urgent queue	--→ NEXT
RP_RQ2	record list of identifiers in normal queue	--→ NEXT
other DLPDU		--→ NEXT
transmission error		--→ NEXT
T1 timer	transaction aborted	--→ NEXT
BA_WAITING_SYNC		
T2 not expired	emit identifier for the basic padding sequence	-→BA_WAITING_SYNC
other DLPDU		--→ NEXT
T2 timer		--→ NEXT

7.5 Bridges

7.5.1 Introduction

This subclause describes additional functions to allow the organization of a DL-subnetwork into several local links interconnected by bridges, as typified in Figure 47.

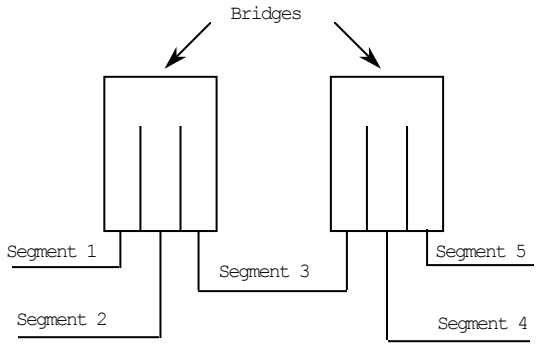


Figure 47 – Typical bridge usage

The specification of a bridge thus requires definition of the principle and the various possible modes for switching the information from one DL-segment to another.

7.5.2 Bridges

This subclause allows the reader to locate the bridge with respect to the DLL communication architecture functions.

7.5.2.1 Situation of the bridge in the DLL communication architecture

The bridge has a certain number of functions specific to the DLL (DLL) and particularly to the Logic Control (LC) level in order to allow a DL-subnetwork organization into several local links.

Each of the logical DL-segments of such a DL-subnetwork has a medium access control (MAC) independent from that of the other DL-segments. Consequently, there will be an active bus-arbitrator on each of the local links.

Figure 48 shows the placement of bridges in the OSI Basic Reference Model (ISO/IEC 7498).

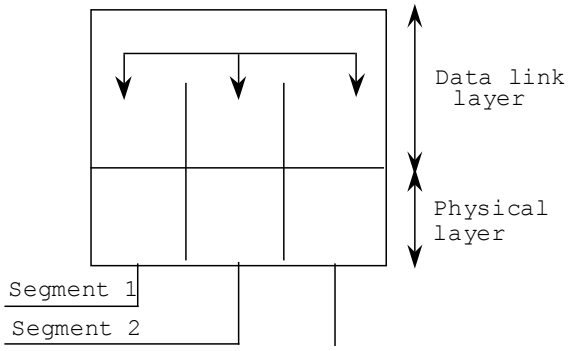


Figure 48 – Architectural placement of bridges in OSI Basic Reference Model (ISO/IEC 7498)

Because a bridge can store a received DLPDU for retransmission, the PhLs of the various local links can be of different speeds. They also can have differing media and modulation.

NOTE 1 Change of media or coding/modulation can also be achieved through Ph-relay devices, commonly called repeaters.

It is quite possible to envisage linking devices which are generalizations of bridges, and which could interconnect different types of DL-protocol. Usually there would be some loss of DL-service when transitioning the different protocol types.

All these possibilities of diversity pose specific problems, which are not dealt with in this addendum. However, a MAC address of the same size is required in all the local links, otherwise it is necessary to include address cross-reference tables.

NOTE 2 Having different access control on two interconnected DL-segments corresponds to considering, for example, a DL-subnetwork with a predominantly centralized administration such as the Type 7 DL-protocol, and another with a predominantly decentralized administration such as found in ISO/IEC 8802-4 or the Type 2 or Type 3 DL-protocol.

7.5.2.2 Communication through a bridge

Bridges are DL-relay devices which allow users of various local links to communicate without the user applications having to know or take into consideration the existence of these bridges, as in Figure 49.

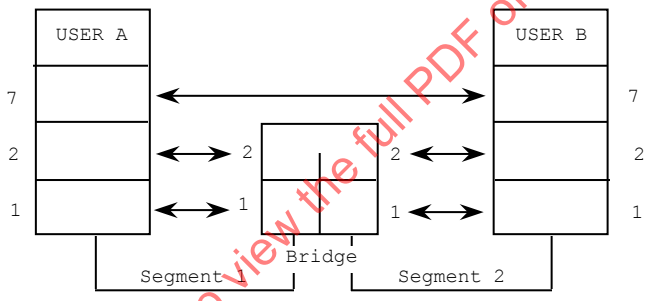


Figure 49 – Representation of an extended link communication

There are thus direct exchanges between the final devices belonging to various DL-segments at the level of the DLS-user and the users, whereas exchanges at the medium access control (MAC) level and the physical (Ph) level are performed from DL-segment to DL-segment.

The functions of such a bridge exclusively concern the messaging services.

NOTE The buffer transfer services can only be executed in a single-DL-segment environment.

7.5.3 Bridge function

7.5.3.1 General

A bridge is formed of several channels each of which has a PhL and a link layer allowing connection between various DL-segments.

The DLL functions relative to the cyclic messaging (outside the service primitives) are minimum functions, which must be supported to accept the bridges.

7.5.3.2 Link functions required

The link functions supported by each bridge channel can contain all the link layer services in case the bridge is also a station. However the minimum link layer resources and functions necessary for operation of the bridge are those of the cyclic messaging, with or without acknowledgement. These include:

— a queue, containing the messages to be transmitted, meant for cyclic transfer of messages called Q_MSGcyc,

— a queue full/not full indicator, associated with Q_MSG.cyc,

— a DLCEP-identifier configured for messaging,

— the attributes associated with a link entity supporting the messaging service with acknowledgement:

— . value of the even/odd bit, as source entity,

— . maximum value of the restart counter,

— . current value of the restart counter.

— a mechanism supporting the cyclic messaging service, whether acknowledged or not.

7.5.3.3 Properties

The extended link must have the properties of connectivity and non-meshing.

Connectivity consists in ensuring that there is always a path from any given DL-segment to any other DL-segment of the extended link .

Non-meshing consists in ensuring that the transmission of a message between the transmitter DL-segment and the destination DL-segment can only be executed along a single path (to prevent messages being received more than once).

These two rules allow calculating the local data base specific to each bridge in order to ensure its operation.

7.5.4 Object model

7.5.4.1 Bridge object description

A bridge is modeled on the link level with the help of the object described in Table 7:

Table 7 – Bridge object description

Class:	BRIDGE
Key Attribute:	Bridge identification
Attribute:	Operating status [ON; OFF]
Attribute:	*List of*
Attribute:	Channel *reference*

7.5.4.1.1 Identification of the bridge

This attribute corresponds to a number allowing unique identification of a bridge in the DL-subnetwork.

7.5.4.1.2 Operating mode

A bridge connected to a DL-subnetwork has two operating modes:

OFF: in this state a bridge does not ensure any of its functions. The passage from this state to another can only be obtained by a local action at the device level.

ON: in this state a bridge ensures its functions. The services offered by each channel are thus usable; the channels, considered individually, are not necessarily in operation.

7.5.4.1.3 List of channel references

This attribute allows designating all the bridge channels. Each of these channels has minimum functions allowing acceptance of the bridge retransmission mechanism.

7.5.4.2 Channel object description

Each of the bridge channels is modeled with the help of the object described in Table 8:

Table 8 – Channel object description

Class:	CHANNEL
Key Attribute:	Channel number
Attribute:	Segment number
Attribute:	Channel validity [Valid; Invalid]
Attribute:	Segment directory *reference*
Attribute:	Network directory *reference*

7.5.4.2.1 Channel number

This attribute is a number identifying a channel relative to a bridge.

7.5.4.2.2 Segment number

This attribute designates the segment connected to the bridge channel identified by the "Channel number" attribute.

7.5.4.2.3 Channel validity

A bridge channel has two states:

— VALID: This state corresponds to normal operation of a bridge channel.

  It is able to:

  — retransmit messages from other channels,

  — redirect received messages towards other channels.

— INVALID: In this state a bridge channel does not ensure any of its functions.

The passage from one state to another is ensured by the system management. These two states are necessary for the configuration and addition/deletion of bridges in the DL-subnetwork.

When the bridge is in the OFF state, each of its bridge channels is in the INVALID state.

7.5.4.2.4 Segment directory reference

The segment directory has a bridge channel giving the rules for retransmission of a message received by this channel on another channel of the same bridge, when the message destination address has a segment number (S/R=1).

The set of segment directories must be built to respect the properties of connectivity and non-meshing.

7.5.4.2.5 Network directory reference

The network directory which has a bridge channel gives the rules for retransmission of a message received by this channel on one or more channels of the same bridge when the message destination address does not have a segment number (S/R=0).

The set of network directories must be built in a manner observing the properties of connectivity and non-meshing.

7.5.4.3 Segment directory object description

The segment directory is modeled with the help of the object described in Table 9:

Table 9 – Segment directory object description

Class:	SEGMENT DIRECTORY
Key Attribute:	Directory number
Attribute:	*List of*
Attribute:	Recognized segment number
Attribute:	Retransmission channel number

7.5.4.3.1 Directory number

This attribute corresponds to a number allowing identification of the individual directory associated with a bridge channel.

7.5.4.3.2 Recognized segment number and retransmission channel number

The received messages with a destination address containing this segment number will be retransmitted on the associated retransmission channel.

7.5.4.4 Network directory object description

The network directory is modeled with the help of the object described in Table 10:

Table 10 – Network directory object description

Class:	NETWORK DIRECTORY
Key Attribute:	Directory number
Attribute:	*List of*
Attribute:	Retransmission channel number

7.5.4.4.1 Directory number

This attribute is a number allowing identification of the network directory associated with a bridge channel.

7.5.4.4.2 Retransmission channel number list

The messages received with a destination address not containing a segment number are retransmitted on each of the retransmission channels.

7.5.5 Mechanisms

The inter-DL-segment messaging can be acknowledged or not. In the first case, the acknowledgement is partial; each of the transactions performed on each of the DL-segments is acknowledged, but there is no acknowledgement from end to end.

The restart mechanism defined in this document is used in the case of transmission of acknowledged messages.

7.5.5.1 Retransmission rule

On reception of a message on one channel:

— Either the destination address contains a DL-segment number belonging to the channel segment directory: the bridge then retransmits the message on the corresponding retransmission channel.

— Or the destination address does not contain a DL-segment number: the bridge then retransmits the message on all the retransmission channels belonging to the receive channel network directory.

7.5.5.2 Retransmission triggering

On reception of an RP_MSG_ACK or RP_MSG_NOACK DLPDU containing a destination address observing the retransmission rule, a bridge channel stores the message:

— in the Q_MSG cycle of the retransmission channel if the destination address contains a DL-segment number,

— in the Q_MSG.cyc file of each of the retransmission channels if the destination address does not contain a DL-segment number.

On reception of a messaging DLPDU, a bridge channel does not call the indication primitives. The acknowledgement status (positive or negative) is determined by the state (full or not) of the retransmission channel Q_MSG.cyc queue.

7.5.5.3 Retransmission mechanism

The retransmission mechanism is triggered on reception of an RP_MSG_NOACK or RP_MSG_ACK DLPDU. The evaluation networks of Figure 50 and Figure 51 describe this mechanism.

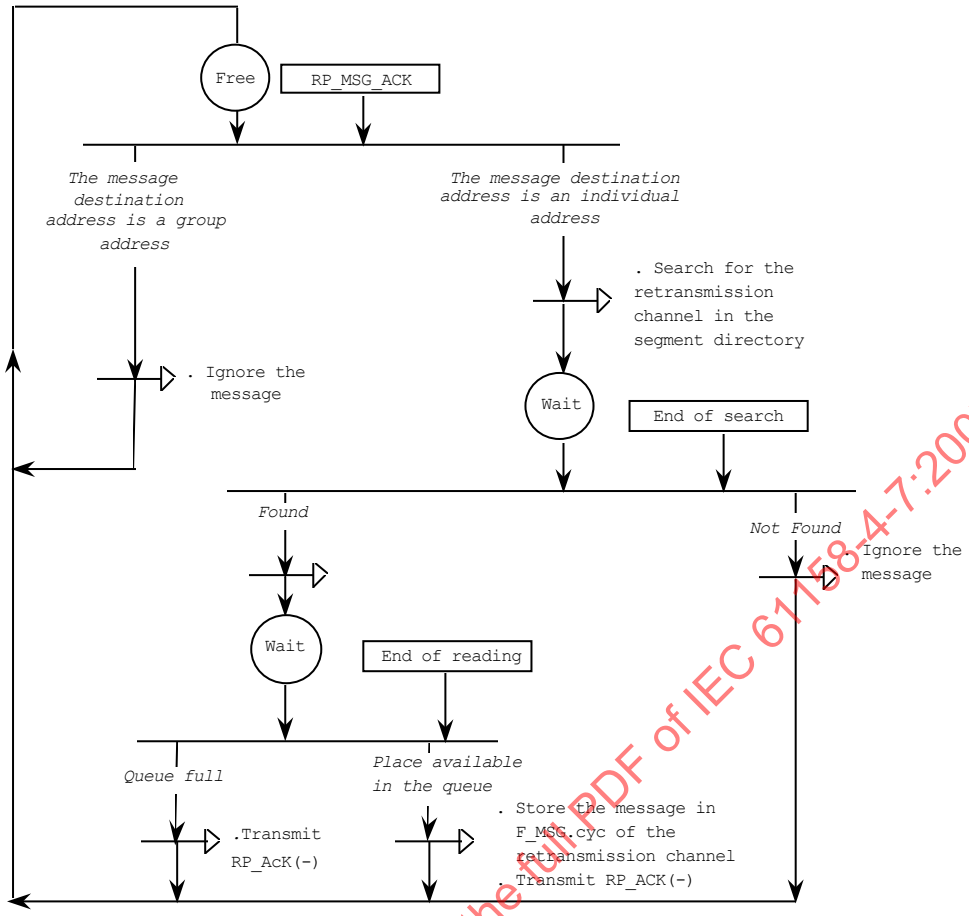


Figure 50 – Evaluation network for reception of an RP_MSG_ACK DLPDU

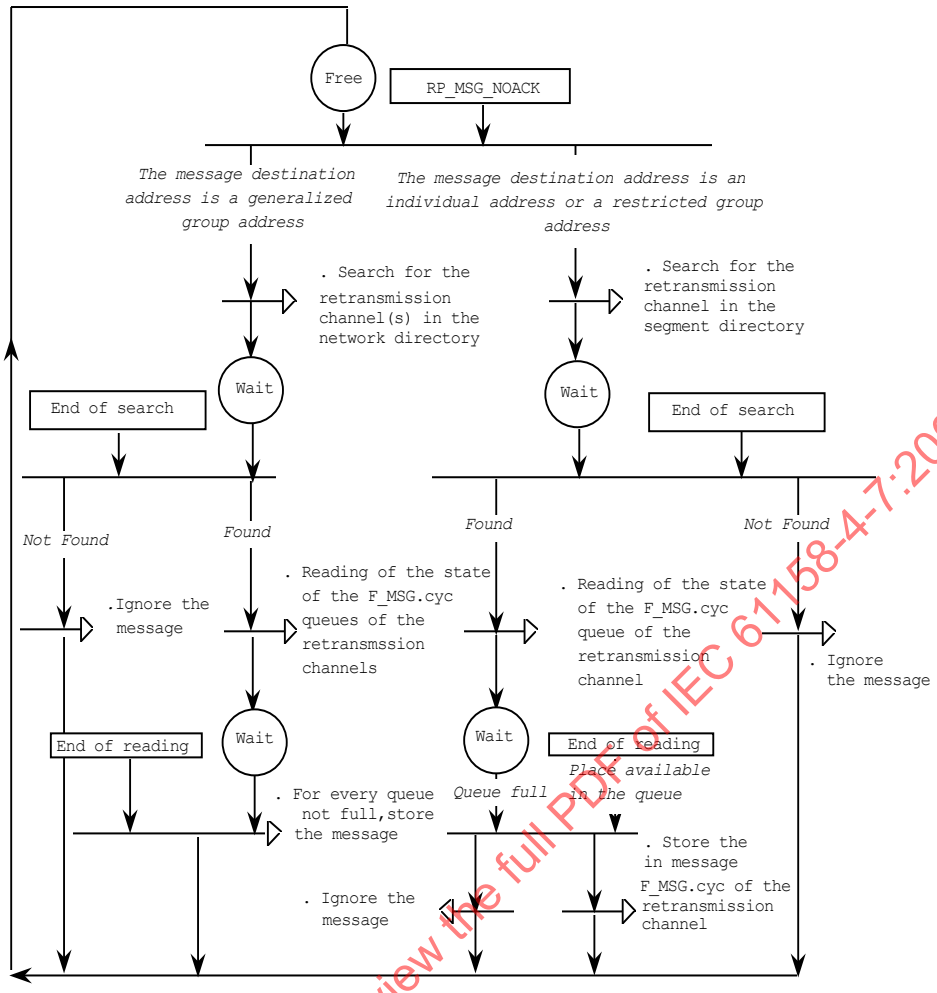


Figure 51 – Evaluation network for reception of an RP_MSG_NOACK DLPDU

7.6 Interfaces

7.6.1 DLS-user interface

The DLL provides the DLS-user with the following set of services:

— buffer writing,
— buffer reading,
— buffer transfer,
— specified explicit request for a buffer transfer,
— free explicit request for a buffer transfer,
— unacknowledged message transfer request,
— acknowledged message transfer request.

The primitives associated with each service completely describe the interface between the data-link and DLS-users. Detailed specification of primitives can be found in Part 3 of this standard.

The primitives associated with each service are given in Table 11.

Table 11 – Service primitives by type

Type of service	Primitive
Buffer writing:	DL-PUT request
	DL-PUT confirm
Buffer reading:	DL-GET request
	DL-GET confirm
Buffer transfer:	DL-SENT indication
	DL-RECEIVED indication
Specified explicit request for a buffer transfer:	DL-SPEC-UPDATE request
	DL-SPEC-UPDATE confirm
Free explicit request for a buffer transfer:	DL-FREE-UPDATE request
	DL-FREE-UPDATE confirm
Unacknowledged message transfer request:	DL-MESSAGE request
	DL-MESSAGE confirm
	DL-MESSAGE indication
Acknowledged message transfer request	DL-MESSAGE-ACK request
	DL-MESSAGE-ACK indication
	DL-MESSAGE-ACK confirm

7.6.2 DL- Management interface

DL-Management offers a number of services that will be specified in the Management document. These services make possible:

Control of DLE functioning modes such as:

— initialization of the DLE operating function,

— control of bus arbitrating,

— changing the bus-arbitrator functioning mode.

Reports on events concerning changes within the DLE such as:

— expiration of a timer,

— queue full,

— bus-arbitrator function change from PASSIVE to ACTIVE.

Parameter setting for DLE functions such as:

— installation of the various data-link objects, that is, the resources associated with identifiers for cyclical and aperiodic services, queues for messages, the bus-arbitrator's scanning table,

— turnaround time,

— the basic timer T0,

— the maximum value of the restart counter,

— the assigning of a physical address to the device.

Access to DLE error and performance counters such as:

— DLPDU fragment error,

— FCS error,

— DLPDU length error,

— pierced DLPDU error,

— coding error,
— overrun error,
— DLPDU sequencing error,
— DLPDU type not identified,
— message reception file saturated,
— message acknowledgement lost,
— number of cyclical transactions,
— number of aperiodic transactions,
— number of message service transactions,
— number of messages refused,
— total number of restarts.

This set of device-level locally available services is used locally by a specific application dedicated to system management.

7.6.3 PhL interface

See IEC 61158-2.

7.7 Conformance

The DLL provides the DLS-user with the following set of services:

a) buffer writing,
b) buffer reading,
c) buffer transfer,
d) specified explicit request for buffer transfer,
e) free explicit request for buffer transfer,
f) unacknowledged message transfer request,
g) acknowledged message transfer request.

A DLE provides the buffer transfer service and the buffer writing service in all cases, and, according to conformance classes, at least one of the following services:

1) buffer reading,
2) specified or free explicit request for a buffer transfer,
3) unacknowledged message transfer request,
4) acknowledged message transfer request.

The reading service makes it possible to extract data from the DLL and route it to the DLS-user. The service is initiated by the DLS-user for the purpose of reading the value of an identified variable.

The writing service makes it possible to deposit data from the DLS-user in the DLL. The service is initiated by the DLS-user for the purpose of writing the value of an identified variable.

The buffer transfer service participates in variable exchanges. In this service the producer of a variable emits a variable response DLPDU for the variable concerned. This DLPDU is then received by the consumer or consumers of the variable. The service generates a report (indication) for each variable emitted or received via the medium and sends this report to the DLS-user.

The specified explicit request for buffer transfer makes it possible to give the bus-arbitrator a request for the circulation of one or more identifier DLPDUs. The transfer of the associated identified variable or variables is thus triggered. This service is initiated by the DLS-user for the purpose of placing one or more identifiers in circulation.

When the service is requested it is linked to a specific identifier, since the request is transmitted to the bus-arbitrator during the cyclical variable exchange concerning this identifier.

The request is fulfilled during the aperiodic variable window (triggered scanning of variables) or during the bus-arbitrator's periodic window (periodic triggered scanning of variables) depending on configuration.

The free explicit request for buffer transfer makes it possible to give the bus-arbitrator a request for the circulation of one or more identifier DLPDUs. Transfer of the associated identified variable or variables is thus triggered. This service is initiated by the DLS-user for the purpose of placing one or more identifiers in circulation.

No identifier is specified when this service is requested. The request is transmitted to the bus-arbitrator during the first exchange of a variable produced by the initiating entity.

The request is fulfilled only during the bus-arbitrator's aperiodic variable window (triggered scanning of variables).

When the bus-arbitrator receives an explicit (specified or free) request for a buffer transfer it begins a transaction that conforms to the buffer transfer service previously described.

For each free request for a buffer transfer, two priority levels are provided: the bus-arbitrator processes the request in the urgent service or normal service mode.

Only urgent service is provided for a specified explicit request for a buffer transfer.

These two types of service make it possible to manage the priority levels of triggered variable exchanges without negative impact on cyclical variable exchanges or message transfers.

The unacknowledged message transfer request service causes the bus-arbitrator to put a message identifier DLPDU in circulation. Circulation of this DLPDU triggers the transfer of an unacknowledged message.

The request is fulfilled either during aperiodic scanning or during the bus-arbitrator's periodical scanning, depending on configuration.

If the request is filled during aperiodic scanning the unacknowledged message transfer is deemed aperiodic. The message transfer request is transmitted to the bus-arbitrator during the first exchange of a variable produced by the source entity. The message identifier DLPDU is then put in circulation during the bus-arbitrator's aperiodic messages window (triggered message scanning).

If the request is filled during the bus-arbitrator's periodic scanning, the unacknowledged message transfer is deemed cyclical. In this case the message transfer request is in the periodical window because of configuration. Thus the message identifier DLPDU is put in circulation without prior request.

In both cases the message exchange between entities then makes use of an unacknowledged message response DLPDU.

When the transaction has been completed the source entity sends the bus-arbitrator a DLPDU indicating the end of the message transaction.

The acknowledged message transfer request service causes the bus-arbitrator to put a message identifier DLPDU in circulation. Circulation of this DLPDU triggers the transfer of an acknowledged message.

The request is fulfilled either during aperiodic scanning or during the bus-arbitrator's periodical scanning, depending on configuration.

If the request is filled during aperiodic scanning the acknowledged message transfer is deemed aperiodic. The message transfer request is transmitted to the bus-arbitrator during the first exchange of a variable produced by the source entity. The message identifier DLPDU is then put in circulation during the bus-arbitrator's aperiodic messages window (triggered message scanning).

If the request is filled during the bus-arbitrator's periodic scanning, the acknowledged message transfer is deemed cyclical. In this case the message transfer request is in the periodical window because of configuration. Thus the message identifier DLPDU is put in circulation without prior request.

In both cases the message exchange between entities then makes use of an acknowledged message response DLPDU.

Upon reception of the message, the receiving entity immediately sends the source entity an acknowledgement DLPDU.

When the transaction has been completed the source entity sends the bus-arbitrator a DLPDU indicating the end of the message transaction.

NOTE Specifications for the DLL support several Application message service entities of various types.

A DLE offers the conformance classes enumerated in Table 12:

Table 12 – Conformance classes

Class	1	2	3	4	5	6	7	8	9
Service									
Buffer transfer	X	X	X	X	X	X	X	X	X
Buffer write	X	X	X	X	X	X	X	X	X
Buffer read		X	X	X	X	X	X	X	X
Free explicit request				X	X			X	X
Specified explicit request			X		X				X
Unacknowledged message						X	X	X	X
Unacknowledged message							X	X	X

Annex A
(informative)

Exemplary FCS implementation

This annex provides an example implementation of FCS generation and FCS syndrome checking for Type 7.

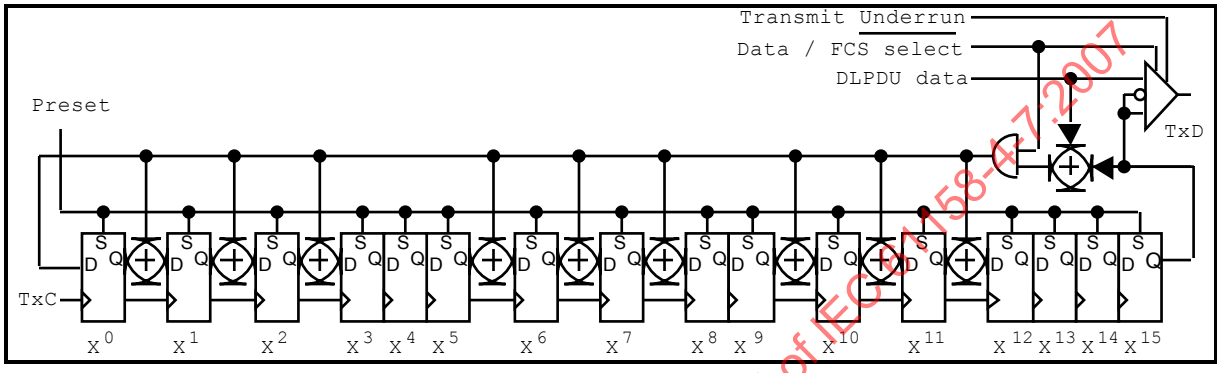


Figure A.1 – Example of FCS generation

In this example, shown in Figure A.1, the FCS is computed in a register consisting of 16 presettable master-slave flip-flops which are interconnected as a linear feedback shift register, with its least significant bit depicted on the left. The initial preset of the register before transmission serves to include the initial L(X) term in the FCS computation. Feedback is disabled during transmission of the FCS itself, and the FCS is transmitted complemented to provide the final L(X) inclusion in the FCS computation. Also shown is optional logic to inhibit the final complementation and transmit a massively incorrect FCS in the case of a transmitter underrun.

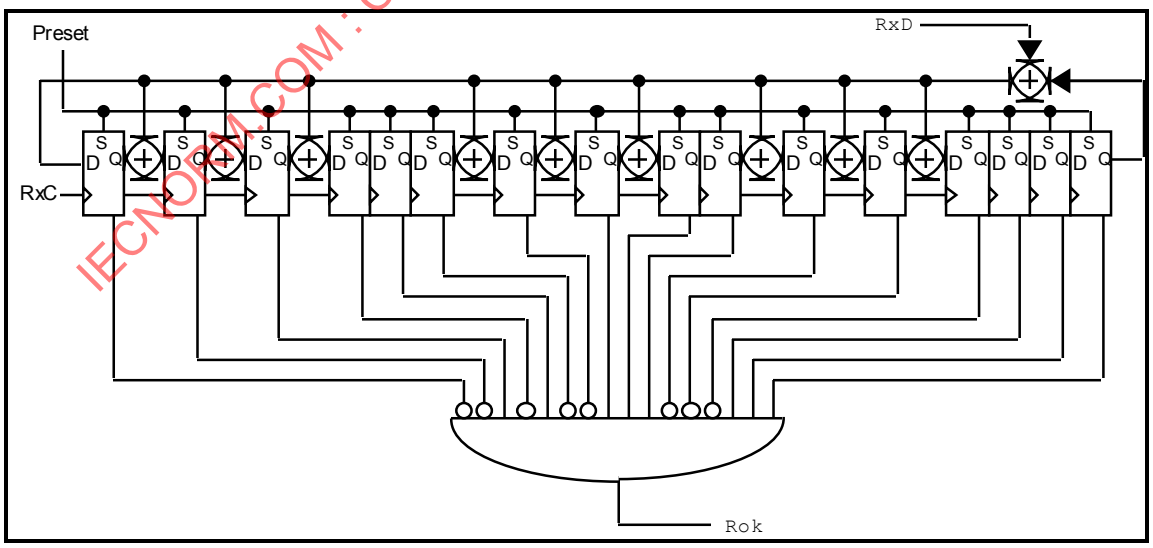


Figure A.2 – Example of FCS syndrome checking on reception

In this example, shown in Figure A.2, the residual FCS is computed in a similar register. The Q outputs of the 16 flip-flops are compared to the expected residual value by the 16-input "and" gate, half of whose inputs are complemented.

In this example, the FCS is computed in a register consisting of 16 presettable master-slave flip-flops which are interconnected as a linear feedback shift register, with its least significant bit depicted on the left. The initial pre-set of the register before transmission serves to include the initial L(X) term in the FCS computation. Feedback is disabled during transmission of the FCS it self, and the FCS is transmitted complemented to provide the final L(X) inclusion in the FCS computation. Also shown is optional logic to inhibit the final complementation and transmit a massively incorrect FCS in the case of a transmitter underrun.

Annex B
(informative)

Object modeling

B.1 Modeling of the IDENTIFIER object

Class:	IDENTIFIER
Key Attribute:	I_Number
Attribute:	Type_APER.CYC (TRUE, FALSE)
Constraint:	Type_APER.cyc=TRUE
Attribute:	*Inherit from* BUFFER
Attribute:	Validity_ID_APER.cyc (TRUE, FALSE)
Attribute:	RQ (IN_PROGRESS, VACANT)
Attribute:	Type_MSG.cyc (TRUE, FALSE)
Constraint:	Type_MSG.cyc=TRUE
Attribute:	*Inherit from* QUEUE
	(Q_MSG_CYC)
Attribute:	Validity_ID_Consumed (TRUE. FALSE)
Attribute:	Type_Prod (TRUE.FALSE)
Constraint:	Type_Prod=TRUE
Attribute:	*Inherit from* BUFFER
	(B_DAT.prod)
Attribute:	Validity_ID_Produced (TRUE. FALSE)
Attribute:	Transmission (PERIODIC, APERIODIC)
Constraint:	Transmission=PERIODIC
Constraint:	Type_APER.cyc=FALSE
Attribute:	Aperiodic_Request (NON_SPEC_RQ_INHIBITED, FREE, NONE)
Constraint:	Aperiodic_Request=NON_SPEC_RQ_INHIBITED
Attribute:	*Inherit from* BUFFER
	(B_REQ)
Attribute:	RQ (IN_PROGRESS, VACANT)
Constraint:	Aperiodic_Request=FREE
Attribute:	RQ (IN_PROGRESS, VACANT)
Attribute:	PR (URGENT, NORMAL)
Attribute:	Reference QUEUE
	(Q_REQ1 OR Q_REQ2)
Constraint:	Type_MSG.cyc=FALSE
Attribute:	Type_MSG.aper (TRUE, FALSE)
Constraint:	Type_MSG.aper=TRUE
Attribute:	MSG (IN_PROGRESS, VACANT)
Attribute:	Reference FILE
	(Q_MSG.aper)

B.2 Description of the IDENTIFIER object attributes

B.2.1 I_Number

This attribute corresponds to a 16-bit word associated with a DL-subnetwork variable and which characterizes the variable uniquely in the DL-subnetwork.

B.2.2 Type_APER.cyc

This attribute characterizes the range of the buffer transfer specified requests.

When a local DLS-user performs a buffer transfer specified explicit request, the associated identifiers are deposited in the B_REQ buffer associated with the identifier indicated at the request.

This request can either remain local in case the B_REQ buffer is scanned during the cyclic part of the scanning table or be transmitted to the bus arbitrator in the other case.

This attribute takes the value TRUE when the buffer transfer specified explicit request service is supported by this identifier and that the scanning of the B_REQ buffer is included in the cyclic part of the scanning table (no RP_DAT_RQ to be transmitted). This service can be offered by produced, consumed, produced and consumed identifiers or by identifiers which are neither produced nor consumed.

This attribute takes the value FALSE:

— when no buffer transfer specified explicit request service is supported by this identifier,

— when the buffer transfer specified explicit request service is supported by this identifier but the request must be transmitted to the bus arbitrator (Scanning of the B_REQ buffer included in the cyclic part of the scanning table). This service can be offered only by the produced identifiers.

When the identifier supports the specified explicit request service, the Type_APER.cyc attribute corresponds to RQ_INHIBITED.

B.2.3 Inherit from BUFFER

The Identifier class inherits from the BUFFER class, thus giving it the attributes relative to B_REQ.

B.2.4 ID_APER.cyc validity

This binary attribute concerns any identifier supporting the buffer transfer specified explicit request periodic service. When this attribute takes the value TRUE, the list of identifiers contained in B_REQ transits on the bus after passage of the identifier request concerned. When this attribute takes the value FALSE, the producer/consumer entity does not answer the ID_RQ requests on this identifier.

B.2.5 RQ

This attribute characterizes the status of a buffer transfer request.

When the buffer transfer explicit requests are transmitted to the bus arbitrator, the requester has information representing the request status (bit RQ).

In the case of a specified request, this attribute has the value IN_PROGRESS, between the instant at which the request was placed in B_REQ and the end of transmission of the RP_RQ following reception of the corresponding ID_RQ type PDU.

In the case of a free request, this attribute has the value IN_PROGRESS between the instant at which a DLCEP-identifier is assigned to the queue in which the request has been placed and the end of transmission of the RP_RQ following reception of the corresponding ID_RQ type PDU.

Outside these intervals, the status of the request is VACANT to indicate that the identifier does not bear any buffer transfer explicit request.

B.2.6 Type MSG.cyc

Two message transfer modes are defined: one (aperiodic) uses a queue Q_MSG.aper reserved for this transfer mode and linked dynamically to identifiers, the other (cyclic) uses a set of Q_MSG.cyc queues of which each one is linked statically to a specific identifier.

This attribute takes the value TRUE when the scanning of the message identifier DLPDU is included in the bus arbitrator cyclic scanning (no RP_DAT_MSG to be transmitted). This service can be offered by produced, consumed, produced and consumed identifiers or by identifiers which are neither produced nor consumed.

It takes the value FALSE:

— when no messaging service is supported by this identifier,

— when the messaging service supported by this identifier requires that the request be transmitted to the bus arbitrator so that the message identifier DLPDU circulates in the messaging periodic window. This service can only be offered by produced identifiers (see Type_MSG.aper).

Inherit from FILE

The Identifier class inherits from the Queue class which contains all the attributes concerning the Q_MSG.cyc resources. The latter corresponds to the queue associated with each identifier supporting the cyclic messaging.

B.2.7 Type_Cons

This attribute defines if a DLCEP-identifier is consumed or not.

Inherit from BUFFER

The Identifier class inherits from the BUFFER class thus giving it the attributes concerning the B_DATcons.

B.2.8 ID_Consumed_Validity

This binary attribute concerns all consumed identifiers. When this attribute takes the value TRUE, a DLCEP-identifier consumes the data meant for it. When this attribute takes the value FALSE, a DLCEP-identifier does not answer in consumption to the requests of the bus arbitrator on this identifier.

B.2.9 Type_Prod

This defines if a DLCEP-identifier is produced or not.

Inherit from BUFFER

The identifier class inherits from the BUFFER class thus giving it the attributes concerning the B_DAT.prod.

B.2.10 Produced ID validity

This binary attribute concerns all the produced identifiers. When this attribute takes the value TRUE, the value contained in B_DAT.prod transits on the bus after passage of the variable identifier.

When this attribute takes the value FALSE, a DLCEP-identifier does not reply in production to the requests of the bus arbitrator on this identifier. Besides, the possible associations between this identifier and Q_RQEQ1, Q_REQ2 or Q_MSG.aper are interrupted.

B.2.11 Transmission

A DLCEP-identifier is exchanged periodically if it appears in the cyclic part of the bus arbitrator scanning table. Otherwise the exchange is triggered on explicit request of a DLS-user.

This attribute conditions the possibility of having some services on this identifier; buffer transfer specified explicit request with RQ_INBIBITED false, buffer transfer free explicit request and periodic messaging.

It thus allows restricting the latter to cyclic identifiers only.

Aperiodic request.

Three buffer transfer explicit request services are defined.

— The identifier which will support the request is specified. This request is not transmitted to the bus arbitrator. The B_REQ is scanned cyclically (RQ_INHIBITED true). In this case the attribute takes the value TRUE.

— The identifier which will support the request is specified. This request is transmitted to the bus arbitrator, the B_REQ buffer is not scanned cyclically. In this case the RQ_INHIBITED attribute takes the value FALSE and the Aperiodic_Request attribute the value SPEC_NON_RQ_INHIBITED value.

— The identifier which will support the request is not specified. The Aperiodic_Request takes the value FREE.

These three types of service are exclusive. If the Aperiodic_Request attribute has the value NOTHING, the identifier will not support any.

B.2.12 Inherit from BUFFER

The identifier class inherits from the BUFFER class, thus giving it the attributes relative to B_REQ.

B.2.13 RQ

See above.

B.2.14 PR

This attribute indicates if the buffer transfer free explicit request must be processed urgently or at the normal speed.

For a specified explicit request, only the urgent priority exists.

The existence of two priority levels for the free explicit requests implying the corresponding resources in the station (Q_REQi) and in the bus arbitrator (Q_IDRQi).

B.2.15 Reference_QUEUE

This attribute allows designating the Q_REQ1 or Q_REQ2 resource, objects resulting from the instance of the queue class. These queues are available to all the identifiers supporting the buffer transfer free explicit request service.

Q_REQ1 contains list(s) of identifiers from one or several normal free explicit requests which have not yet been transmitted to the bus arbitrator.

B.2.16 MSG.aper type

Two message transfer modes are defined: one (aperiodic) uses an queue reserved for this transfer mode and dynamically linked to identifiers, the other (cyclic) uses a set of queues each of which is statically linked to a specific identifier.

This attribute takes the value TRUE when the scanning of the message identifier DLPDU is not included in the bus arbitrator cyclic scanning, that is it is necessary to transmit a bus arbitrator request so that consequently the message identifier DLPDU circulates in the course of the aperiodic window reserved for the messaging. This service can only be offered by produced identifiers.

It takes the value FALSE:

— When no messaging service is supported by this identifier,

— When the scanning of the message identifier DLPDU is included in the bus arbitrator cyclic scanning (see Type_MSG.cyc).

B.2.17 MSG

This attribute characterizes the state of a message aperiodic transfer request.

It has the value IN_PROGRESS between the instant when the identifier has been associated with the queue reserved for transfer of messages and the reception of the message transaction linked to the corresponding ID_MSG DLPDU. Outside this interval, the status of the request is VACANT to indicate that the identifier bears no message transfer request.

B.2.18 QUEUE_Reference

This attribute allows designating the Q_MSG.aper object resource resulting from the instance of the Queue class. This queue is available to all the identifiers supporting the aperiodic messaging service.

It contains the message(s) resulting from one or several message transfer requests which have not yet been satisfied.

B.3 Modeling of the QUEUE object

Class:	QUEUE
Key Attribute:	Q_Number
Attribute:	Queue_status (EMPTY, FULL, PLACE AVAILABLE)
Attribute:	Type (Q_REQ, Q_MSG)
Constraint:	Type=Q_REQ
Attribute:	Content Q_REQ (identifier lists queue)
Attribute:	Connection (TRUE, FALSE)
Constraint:	Type=Q_MSG
Attribute:	Content Q_MSG (message queue)
Attribute:	Num_Attachment (integer)

B.4 Description of the QUEUE object attributes

B.4.1 Q_Number

This attribute associated with a queue allows characterizing the latter uniquely in the DLE.

B.4.2 Queue_status

This attribute describes the actual use of the queue. The possible values are:

An Q_MSG queue is empty when it contains no message. An F_MSG queue can contain an empty message; this will consist only of the source address and the destination address. In this case the Queue_Status attribute takes the value PLACE AVAILABLE or FULL.

B.4.3 Q_REQ content

This attribute corresponds to the content of the Q_REQ1 and Q_REQ2 queues. It is the set of identifier lists which have not yet been transmitted to the bus arbitrator. Each list has been covered locally by a free explicit request and is transmitted on the local link automatically. At an RP_RQ reply several lists can be transmitted in so far as all these lists are not made of more than 64 identifiers.

B.4.4 Attachment

This attribute indicates if the queue is associated with a DLCEP-identifier or not.

B.4.5 Q_MSG content

This attribute corresponds to the content of the Q_MSG.received, Q_MSG.Cyc and Q_MSG.aper. files. It is a messaging file.

In the case of Q_MSG.cyc and Q_MSG.aper. files, it includes all the messages covered by a local message transfer request which has not been satisfied. In the case of Q_MSG.received, it is the set of messages for which the station has been recognized as destination.

B.4.6 Attachment_Number

This attribute indicates how many associations are missing between the queue and its identifiers. It is a whole number greater than 0 and lower than the number of places in the queue. It thus corresponds to the number of messages present in the queue less the number of identifiers associated with the latter.

B.5 Modeling of the BUFFER object

Class:	BUFFER
Key Attribute:	B_Number
Attribute:	Type (B_DAT, B_REQ)
Attribute:	Availability (OCCUPIED, FREE)
Constraint:	Type=B_DAT
Attribute:	Content Prod_Cons (variable value)
Constraint:	Type=B_REQ
Attribute:	Content B_REQ (list of identifiers)

B.6 Description of the BUFFER object attributes

B.6.1 B_Number

This attribute associated with a buffer allows characterizing the latter uniquely in a DLE.

B.6.2 Type

This attribute allows determining if the buffer is of the B_DAT type (produced or consumed) or B_REQ.

B.6.3 Availability

This attribute is meant to solve the problems of access conflict which could take place.

B.6.4 Prod_Cons content

This attribute corresponds to the content of a B_DAT or B_DATcons buffer. This is the value of a produced or consumed variable.

B.6.5 B_REQ content

This attribute corresponds to the content of a B_REQ buffer. It is a list of identifiers coming from a buffer transfer specified explicit request.

Annex C
(informative)

Topology of multi-segment DL-subnetwork

C.1 Introduction

This annex describes how to specify the topology of a multi-segment DL-subnetwork. The aim is to propose a data structure, which could be minimal while allowing correct operation of the bridge retransmission function.

The topology of a DL-subnetwork can first be specified globally, in order to verify a certain number of properties (connectivity, non-meshing, etc.); then on the basis of this specification the local data base specific to each bridge must be calculated in order to ensure it operates correctly.

Although this appendix proposes a method to achieve this goal, only the specifications of the data structures, global or local to each bridge, which define the DL-subnetwork topology, as well as the properties which it should fulfil, must be taken into account in the standard. The suggested method shows how to obtain a solution to the problem by taking into account certain optimization problems.

C.2 Global specification

The topology of a multi-segment DL-subnetwork can be defined by the following elements:

— the set S of its segments: $S = \{ s_i \mid i \in [1, m] \}$

— the set B of its bridges: $B = \{ b_k \mid k \in [1, m] \}$

— and for each bridge of B, the data of a matrix B^k of dimension n × n. whose coefficients b_{ij}^k are defined by:

— $b_{ij}^k = 0$ if i = j;

— $b_{ij}^k = \infty$ if the bridge b_k does not allow transfer of messages from segment s_i to segment S_j ;

— $b_{ij}^k = \alpha$ with $\alpha \in \mathbf{R}^{+*}$, if the bridge b_k allows the transfer of messages from segment s_i towards segment s_j, with α as load coefficient which allows taking into account of a different efficiency rate according to the transfers.

A load coefficient b_{ij}^k can represent the load, as a rate of occupation of the medium, of the retransmission segment s_j. In reality, either the destination is directly s_j, or there are several paths possible, passing through intermediate segments, to reach s_j and in this case the choice shall be to pass by the least loaded path.

It is of course possible to take as coefficients of the same value (1 for example).

If a bridge allows two-way retransmission with the same load coefficient for the two directions, its matrix is symmetrical.

The matrix B^k of a bridge also allows knowing all the segments to which it is connected:

— either in reception, S_r^k = { segments whose corresponding line in the matrix includes at least one non-null finite coefficient}; note n_r^k = card (S_r^k),

— or in transmission, S_e^k = {segments whose corresponding column in the matrix includes at least one non-null finished coefficient}; note n_e^k = card (S_e^k).

C.3 Local specification

The information which a bridge must have locally allows it to answer the following question: when I receive a message on such a segment $sr_i \in S_r^k$ destined for another segment s_j, must I do nothing or must I retransmit on segment $se_h \in S_e^k$).

To fulfil this purpose, it is enough to allocate to each bridge b_k a transfer matrix T^k with dimensions $n_r^k \times n$, whose elements r_{ij}^k are defined by:

— the line index $i \in [1, n_r^k]$ references segments sr_i connected in reception ($\in S_r^k$),

— the column index j $\in [1, n]$ references the segments s_j of the DL-subnetwork ($\in S$),

— $r_{ij}^k = 0$ if on reception of a message on segment $sr_i \in S_r^k$ addressed to segment s_j, the bridge shall not do anything, either because s_j cannot be reached via this bridge, or because $sr_i = s_j$ (a bridge shall not retransmit a message received from a segment towards this same segment),

— $r_{ij}^k = se_h$, with $se_h \in S_e^k$, if on reception of a message on segment $sr_i \in S_r^k$ addressed to segment s_j, the bridge must retransmit to segment se_h.

NOTE Indexes i and h correspond to channel numbers whereas sr_i is the segment connected in reception to channel i and se_h is the segment connected in transmission to channel h.

C.4 Properties

The properties which should satisfy the DL-subnetwork are topological connectivity and non-meshing.

Topological connectivity consists in ensuring that there is always a path from any given segment of S to any other segment of S.

Non-meshing consists in ensuring that the transmission of a message from a transmitter located on segment s_e and addressed to a receiver located on segment s_r can be routed by only one path (thus preventing the message from being received more than once).

In fact, it is the definition of the local specification of each bridge and the calculation of its transfer matrix which ensure this property: by definition, on reception of a message on segment sr_i addressed to segment s_j, either the bridge does not retransmit it, in particular if segment sr_i is equal to segment s_j, or the bridge retransmits it on a single segment se_h, whereas by calculation of the matrix, it is necessary to make sure that one and only one bridge, connected in reception to segment sr_i, retransmits the message.

C.5 Methods

The method consists in calculating the matrix C of the minimum loads of the paths between any two segments. By this way we check the topological connectivity since none of these coefficients is infinite.

The second stage consists in calculating the transfer matrix of each bridge so that this gives the global DL-subnetwork the property of non-meshing while preserving the property of topological connectivity.

C.5.1 Minimum load matrix

a) Load matrix of rank P

Definition: the load matrix of rank P, C^P, with dimensions n × n, is the matrix whose coefficients c_{ij}^P give the minimum load to travel from segment i towards segment j by passing via not more than P bridges.

We have:

— $c_{ii}^P = 0$,

— $c_{ij}^P = \infty$, if there is no path from segment i to segment j by passing via not more than P bridges;

— if c_{ij}^P is finite, the optimal corresponding path includes not more than P bridges (it can include less than P).

Obtaining by recurrence:

— The coefficients c_{ij}^1 of the load matrix of rank 1, C^1, are given by:

— $$c_{ij}^1 = \min_{k \in [1,m]} \left(b_{ij}^k \right)$$

We have:

— $c_{ii}^1 = 0$,

— $c_{ij}^1 = \infty$, if there is no permanent bridge allowing transfer from segment i towards segment j,

— c_{ij}^1 is finite, if there exists one or more bridges allowing the transfer from segment i towards segment j.

— The coefficients c_{ij}^P , for P > 1, of the load matrix of rank P, C^P, are given by:

$$c_{ij}^P = \min_{k \in [1,m]} \left(c_{ik}^1 + c_{kj}^{P-1} \right)$$

In reality, the minimum load between two segments i and j passing via P bridges corresponds to a path composed of:

— a bridge allowing the passage from segment i to segment k, with a minimum load c_1,

— and a path with minimum load c_2 between this segment k and segment j, passing via P-1 bridges.

The intermediate bridge is most often used so that $c_1 + c_2$ are minimal.

b) Minimum load matrix

Definition: the minimum load matrix C, dimension n × n, is the matrix whose coefficients c_{ij} give the minimum load to go from segment i to segment j. We thus have:

— $c_{ii} = 0$;

— $c_{ij} = \infty$, if there is no path from segment i to segment j ;

— if c_{ij} is finite, there is a path of length L (s_1, s_2,... , s_L) with s_1 = *i* and s_L. = *j*, passing via bridges b^{k_h}, h ∈ [1, *L* - 1], whose load is c_{ij} with:

$$c_{ij} = \sum_{h=1}^{h=L-1} b^{k_h}_{s_h s_{h+1}}$$

By definition the topological connectivity is well ensured by the fact that all the minimum load matrix coefficients *C* are finite.

Property: the minimum load matrix *C* is the limit of the load matrixes of rank P, when P tends to infinity:

$$C = \lim_{P \to \infty} C^P$$

In reality, the series C^P is stationary, at least from rank m, where m is the number of bridges. A path which passes via more than m bridges passes at least twice via the same bridge and cannot thus have a minimum load.

Suppose Q the row from which the C^P series is stationary ($C^{Q+1} = C^Q$).

C.5.2 Calculation of the transfer matrices

Suppose now that the DL-subnetwork is topologically connected.

The transfer matrices T^k are calculated by iteration according to the number P of bridges, from 1 to Q, requiring a minimum load path from a source segment s and a destination segment d.

At the start, $T^k = 0$ for every k.

The transfer matrix coefficients are referenced in the same manner as in C.3, that is:

— the line index i ∈ [1, nr^k] references the sr_i segments connected in reception (∈ Sr^k),

— the column index j ∈ [1, *n*] references the s_j segments of the DL-subnetwork (∈ *S*).

a) Passage of a segment s to a segment d via 1 bridge

For all pairs of segments s and d such that c^1_{sd} is finite.

for one and only one k (as selected) such that $c^1_{sd} = b^k_{sd}$,

the following assignment is performed for the transfer matrix T^k:

— for i ∈ [1, nr^k] such that sr_i = *s* and for j ∈ [1, *n*] such that s_j = *d* , then $r^k_{ij} = s_j$

b) Passage of a segment *s* to a segment d via P bridges

For every pair of segments *s* and d such that c^P_{sd} is finite whereas c^{P-1}_{sd} is infinite,

for one and only one k (as selected) such that $\exists\, s' \,\middle|\, c_{sd}^{P} = b_{ss'}^{k} + c_{s'd}^{P-1}$,

the following assignment is thus performed for the T^k transfer matrix:

— for i ∈ [1, nr^k] such that sr_i = s and for j ∈ [1, n] such that s_j = d , then $r_{ij}^{k} = s_h$ with s_h = s'

In the last two paragraphs, the bridge k which verifies the necessary property is not necessarily unique, but an assignment must be made for a single bridge to ensure the property of non-meshing.

Annex D
(informative)

Management of transmission errors

D.1 Transmission of RP_DAT_XX

If a transmission error is detected during the transmission of a given response DLPDU, the transmission is interrupted, and:

— the DL-SEND primitive is not generated;

— in the case of a configured identifier for the service of the free explicit request service, the RQ indicator is not modified;

— in the case of a DLCEP-identifier configured for message aperiodic transfer, the MSG indicator value is unchanged.

The transmission of DL-SEND_ind and the management of the RQ and MSG indicators are performed only at the end of the RP_DAT_XX DLPDU transmission and if the variable producer has not detected an error during transmission. The evaluation DL-subnetwork of Figure D.1 describes this operation.

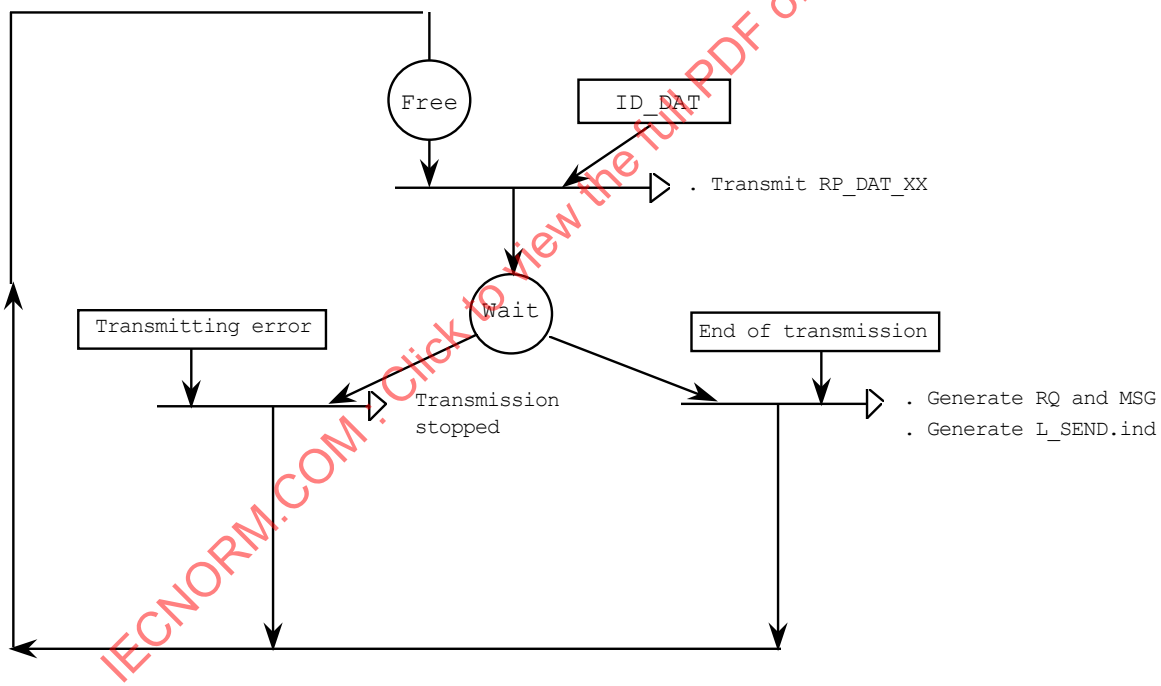


Figure D.1 – Evaluation DL-subnetwork for transmission of RP_DAT_XX

D.2 Transmission of a free RP_RQ(1/2)

On detection of an error during transmission of RP_RQ1 or RP_RQ2, forming continuation of a buffer transfer free explicit request, the producing entity does not generate any DL-FREE-UPDATE confirm primitive and does not scroll the list of identifiers. The evaluation DL-subnetwork of Figure D.2 describes this operation.

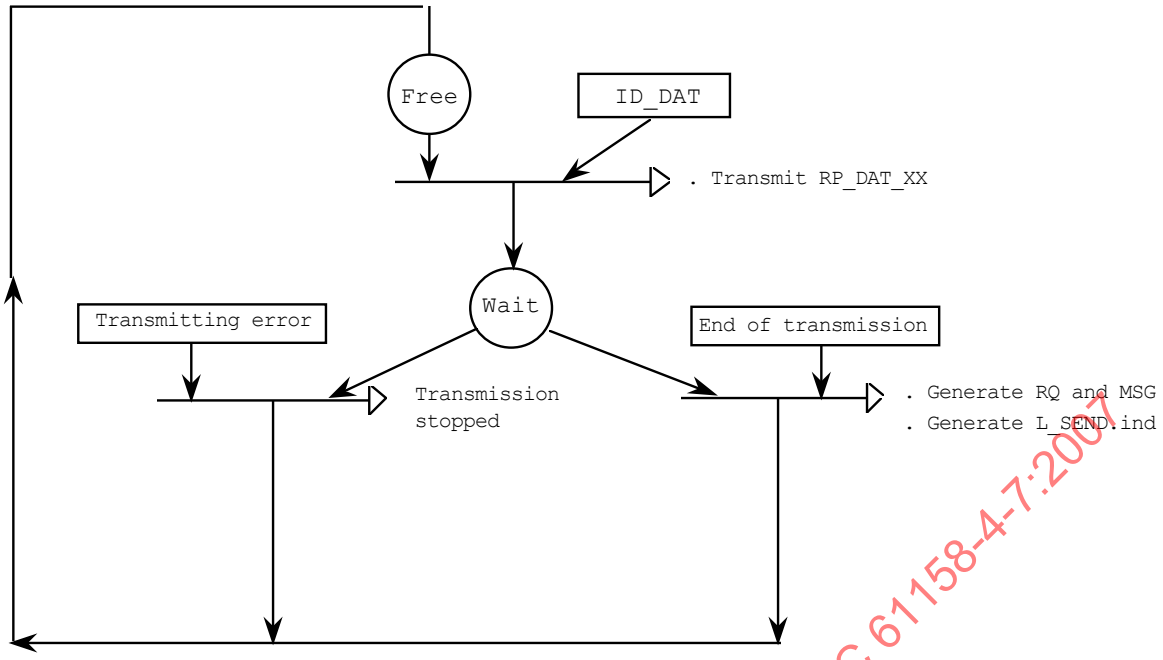


Figure D.2 – Evaluation DL-subnetwork for transmission of a free RP_RQ(1/2)

After transmission of the RP_RQ without error, the request is detached from the identifier concerned (RQ set to 0); If the FIFO is not empty, the request will be attached at the next reception of an ID_DAT concerning a DLCEP-identifier configured for this service. The RQ bit of this identifier shall be set to 1.

D.3 Transmission of the specified RP_RQ1

On detection of an error during transmission of RP_RQ1, following a buffer transfer specified explicit request, the producing entity interrupts the transmission of the DLPDU and does not generate a DL-SPEC-UPDATE confirm primitive.

Besides, the RQ indicator remains set to 1: the buffer transfer request shall be transmitted to the BA at the next transaction in case the RQ_INHIBITED indicator of the identifier has the FALSE value. The evaluation DL-subnetwork of Figure D.3 describes this operation.

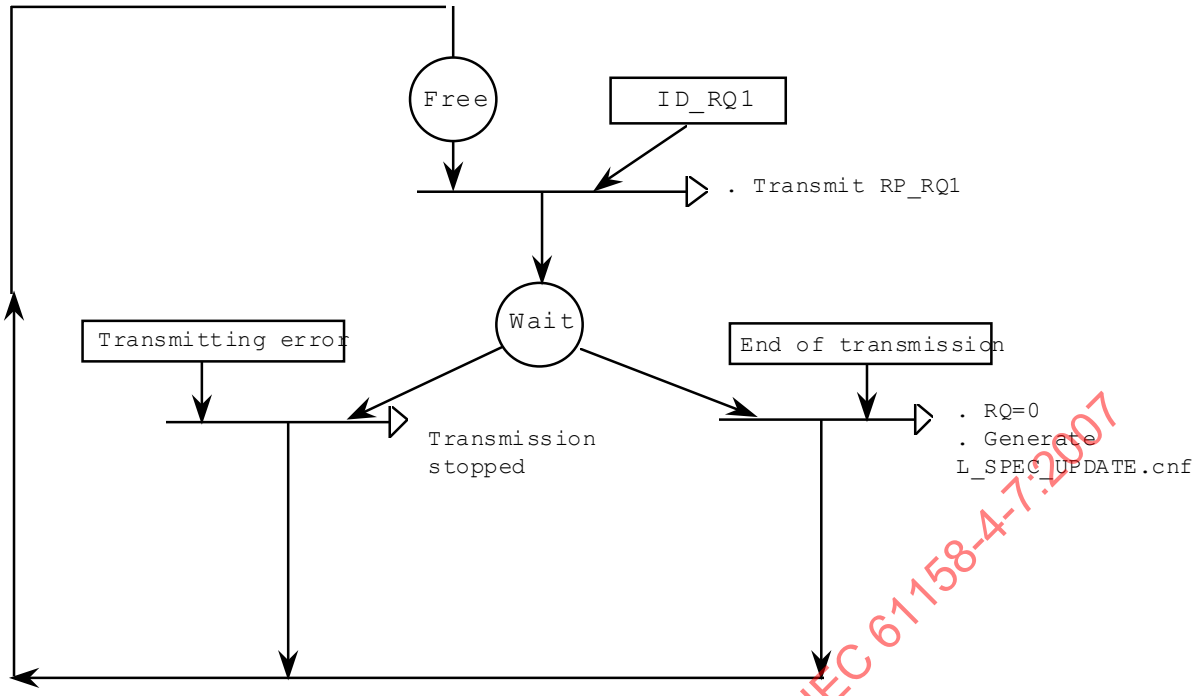


Figure D.3 – Evaluation DL-subnetwork for transmission of the specified RP_RQ1

D.4 Transmission of RP_MSG_NOACK

Several behaviors are possible during an error at transmission of an RP_MSG_NOACK:

— The producer/consumer entity interrupts the transmission and restores in the initial context, keeping the message. The transmission of the RP_MST_NOACK is then postponed to the next request from the bus arbitrator.

— The producer/consumer entity interrupts the transmission keeping only the message. The link layer indicates the loss of the message to the DLS-user with the help of the appropriate confirmation.

D.4.1 First behavior

On detection of an error during transmission of RP_MSG_NOACK, the producing entity interrupts the transmission of the DLPDU.

Besides:

— It does not generate the primitive of the DL-MESSAGE-ACK confirm primitive.

— The place in the message queue is not freed.

— If the identifier associated with the request is not included in the BA periodical scanning table, the MSG indicator remains positioned to 1; the message transfer request will be transmitted to the BA at the next transaction.

The evaluation DL-subnetwork of Figure D.4 describes this operation.

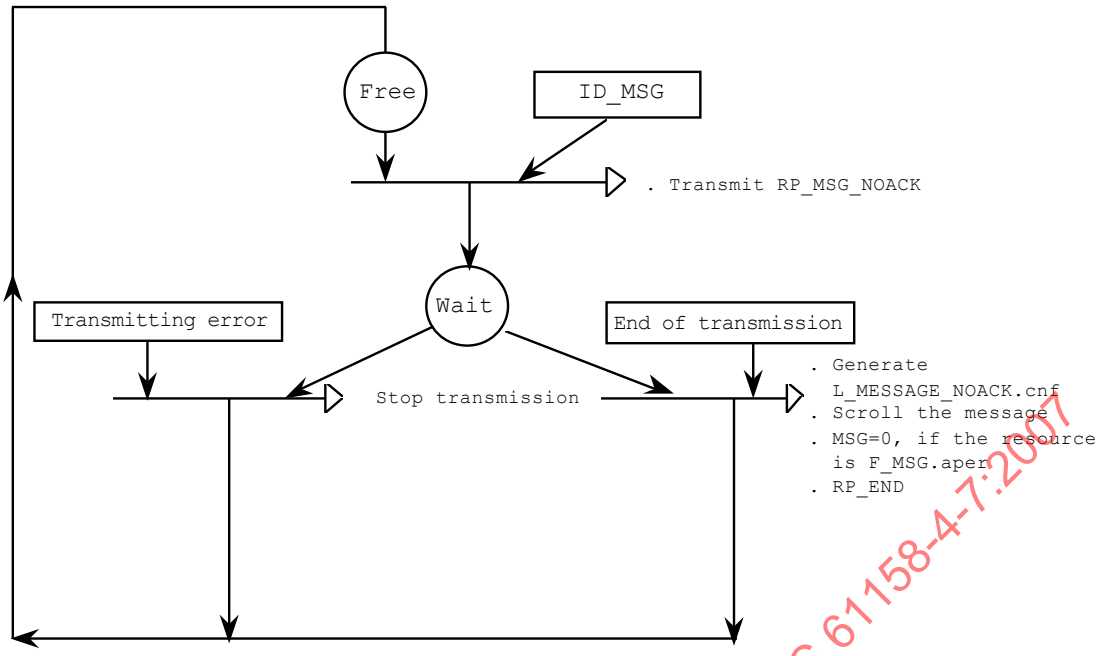


Figure D.4 – Evaluation DL-subnetwork for transmission of RP_MSG_NOACK, first behavior

D.4.2 Second behavior

On detection of an error during the transmission of RP_MSG_NOACK, the producing entity interrupts the transmission of the DLPDU.

Besides:

— It generates a DL-MESSAGE-ACK confirm primitive whose status signifies Transmission error (1).

— The place in the queue is freed.

The evaluation DL-subnetwork of Figure D.5 describes this operation.

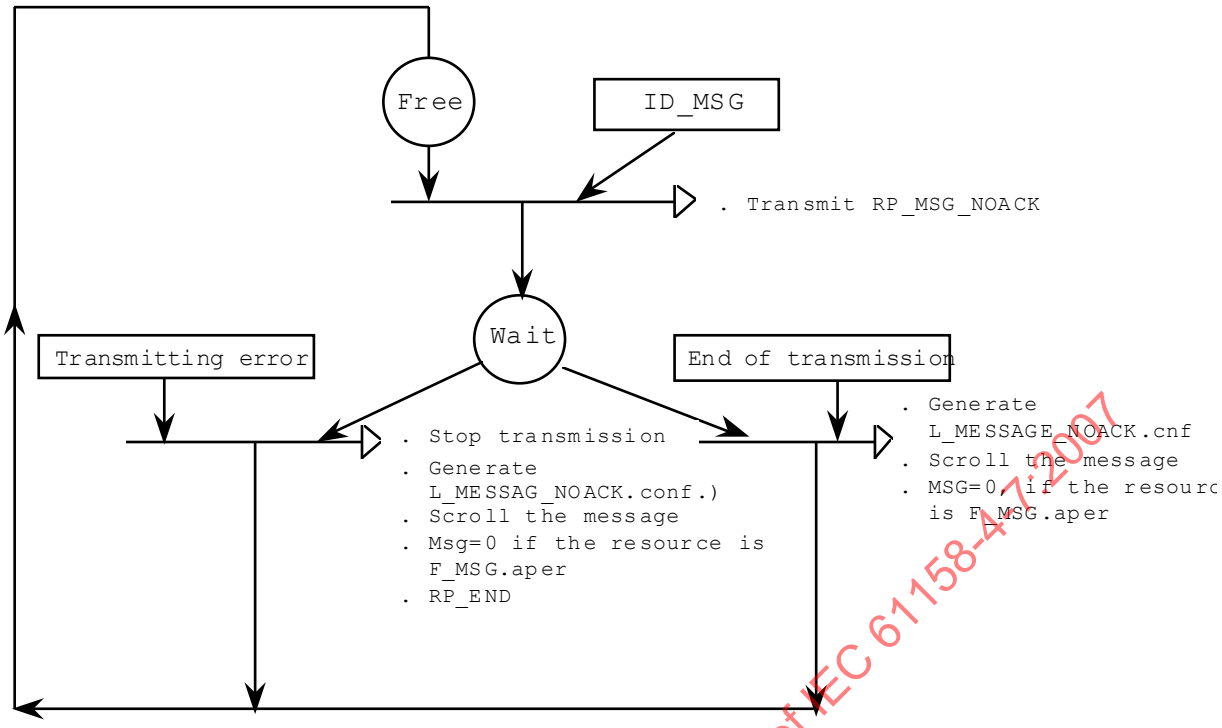


Figure D.5 – Evaluation DL-subnetwork for transmission of RP_MSG_NOACK, second behavior

NOTE This status, not provided for by this standard, is to be added in the case of choice of this behavior at error during transmission.

D.5 Transmission of RP_MSG_ACK

Several behaviors are possible during an error at transmission of RP_MSG_ACK:

— The producer/consumer entity interrupts the transaction and is restored in the initial context, keeping the message. The transmission of the RP_MSG_ACK is then postponed to the next request coming from the bus arbitrator.

— The producer consumer entity interrupts the transmission and if the restart counter permits, retransmits the RP_MSG_ACK DLPDU immediately.

D.5.1 First behavior

On detection of an error during transmission of RP_MSG_ACK, the producing entity interrupts the transmission of the DLPDU.

Besides:

— It does not generate the DL-MESSAGE-ACK confirm primitive.

— The place in the message queue is not freed.

— If the identifier associated with the request does not form part of the BA periodic scanning table, the MSG indication shall be positioned at 1; the message transfer request will be transmitted to the BA at the next transaction.

The evaluation DL-subnetwork of Figure D.6 describes this operation.

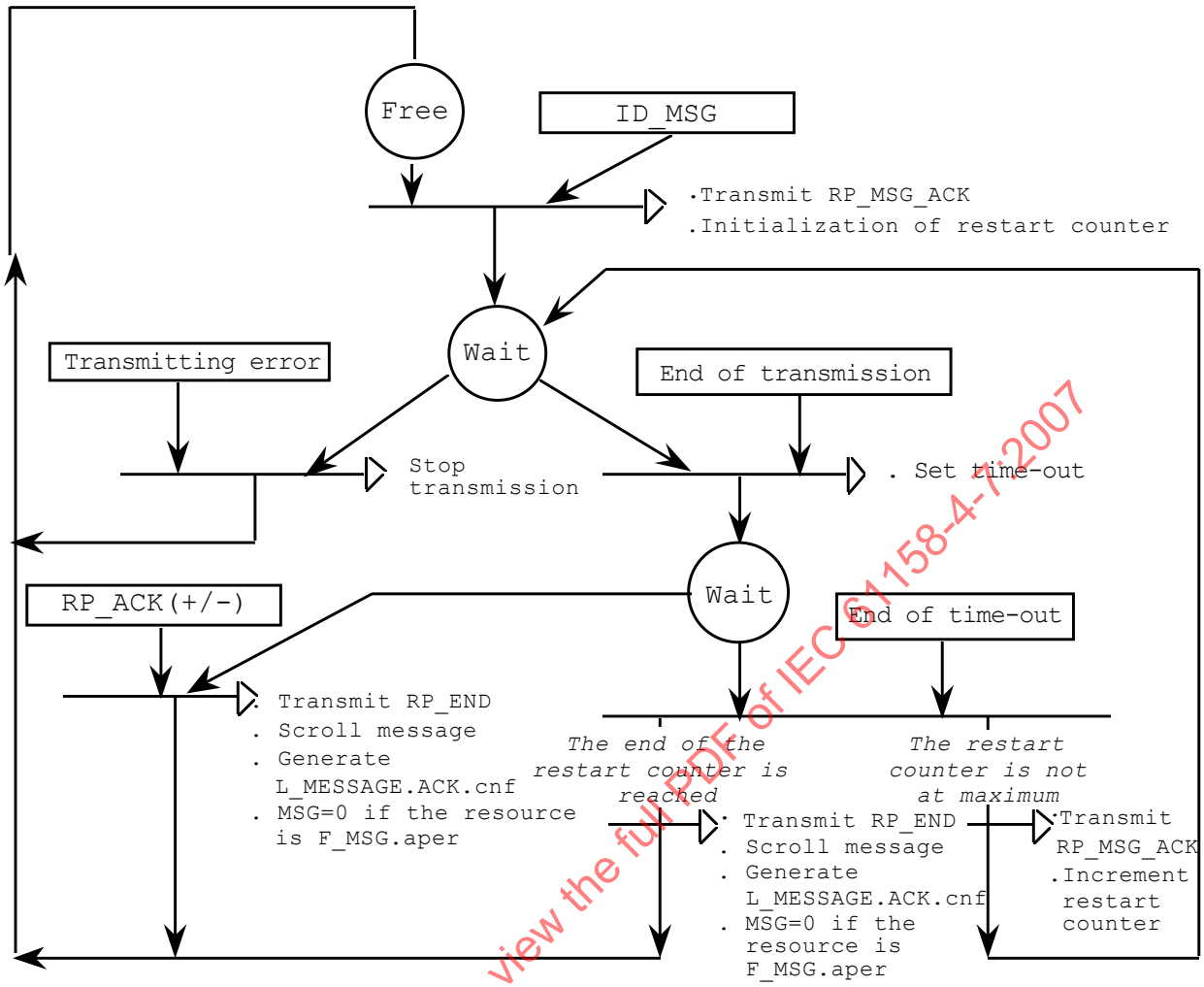


Figure D.6 – Evaluation DL-subnetwork for transmission of RP_MSG_ACK, first behavior

D.5.2 Second behavior

On detection of an error during transmission of the RP_MSG_ACK DLPDU, the actions by the producing entity are the following.

— Interruption of the DLPDU transmission.

— If the restart counter is not reached the producing entity performs a restart immediately without waiting for the expiration of time-out T6.

The evaluation DL-subnetwork of Figure D.7 describes this operation.

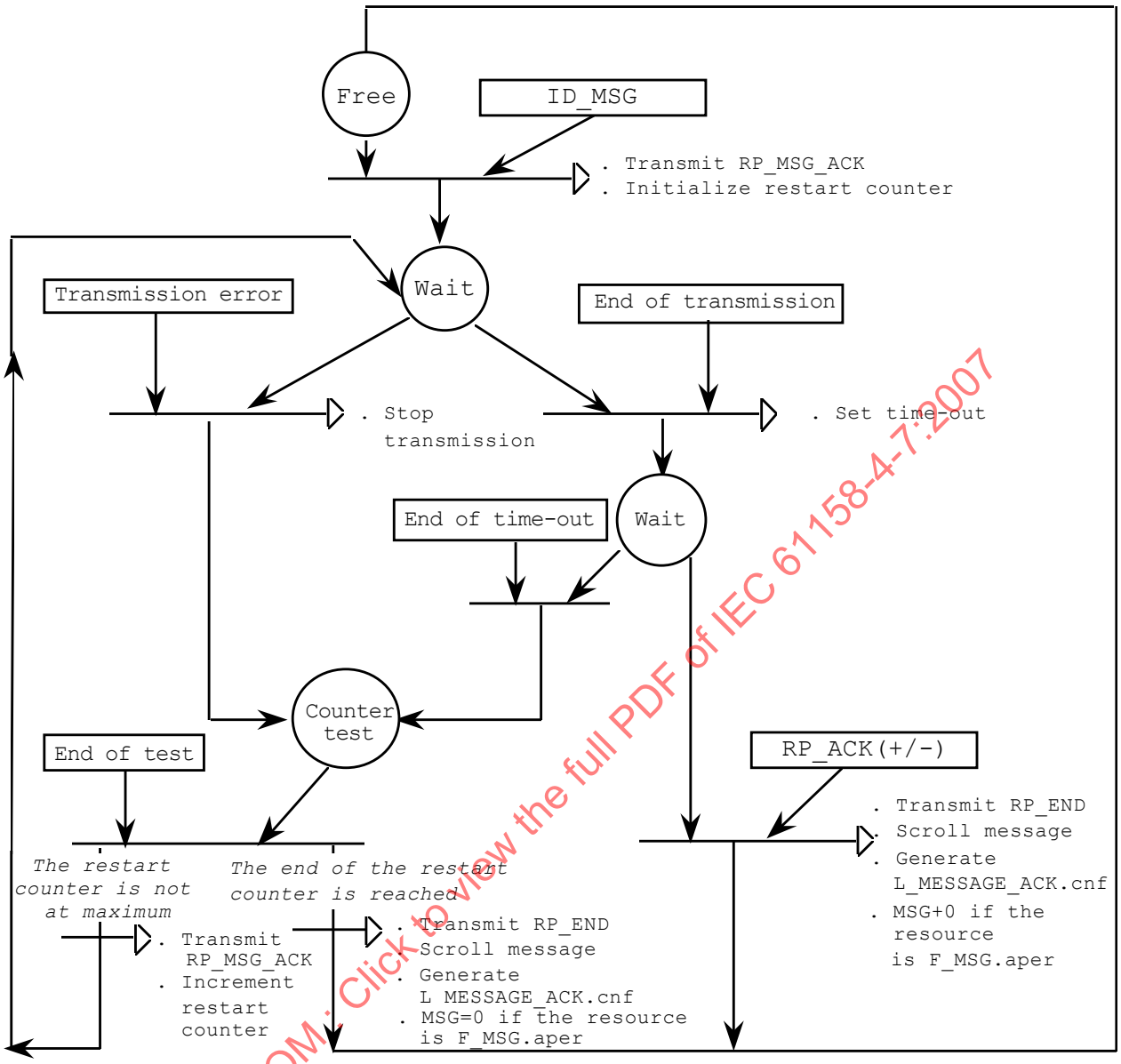


Figure D.7 – Evaluation DL-subnetwork for transmission of RP_MSG_ACK, second behavior

Bibliography

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

IEC 60847, *Characteristics of LANs*

IEC 60870-5-1, *Telecontrol equipment and systems – Part 5: Transmission protocols – Section one: Transmission frame formats*

IEC 60955, *Process data highway, Type C (PROWAY C), for distributed process control systems*

IEC 61131-2, *Programmable controllers – Part 2: Equipment requirements and tests*

IEC 61131-3, *Programmable controllers – Part 3: Programming languages*

IEC/TR 61158-1 (Ed.2.0), *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-5-7, *Industrial communication networks – Fieldbus specifications – Part 5-7: Application layer service definition – Type 7 elements*

IEC 61158-6-7, *Industrial communication networks – Fieldbus specifications – Part 6-7: Application layer protocol specification – Type 7 elements*

IEC 61784-1 (Ed.2.0), *Industrial communication networks – Profiles – Part 1: Fieldbus profiles*

ISO/IEC 2022, *Information technology – Character code structure and extension techniques*

ISO/IEC 8802 (all parts), *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks*

- ISO/IEC TR 8802-1, *Specific requirements – Part 1: Overview of Local Area Network Standards*
- ISO/IEC 8802-2, *Specific requirements – Part 2: Logical link control*
- ISO/IEC 8802-3, *Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*
- ISO/IEC 8802-4, *Information processing systems – Local area networks – Part 4: Token-passing bus access method and physical layer specifications*
- ISO/IEC 8802-5, *Specific requirements – Part 5: Token ring access method and physical layer specifications*

ISO/IEC 9314-2, *Information processing systems – Fibre Distributed Data Interface (FDDI) – Part 2: Token Ring Media Access Control (MAC)*

ISO/IEC 9646-1, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 1: General concepts*

ISO/IEC 9646-2, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 2: Abstract test suite specification*

ISO/IEC 10646-1, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*

ISO/IEC 15802-1, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Common specifications – Part 1: Medium Access Control (MAC) service definition*

ISO 1177, *Information processing – Character structure for start/stop and synchronous character oriented transmission*

ISO 3309, *Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures – Frame structure.*

ISO 8509, *Information processing systems – Open System Interconnection – Service Conventions*

ITU-T V.41, *Code-independent error-control system.*

EN 50170:1996, *General purpose field communication system*
Amendment 3-4:2002, *Data link layer definitions*
Amendment 7-4:2002, *Network management*

ANSI X3.66 (R1990), *Advanced data communication control procedures (ADCCP)*

ANSI X3.159, *Information Systems – Programming Language C*

ANSI X3J16 / ISO WG21 *committee draft working paper for a C++ standard*

Internet Engineering Task Force (IETF), *Request for Comments (RFC)*:

RFC 791, *Internet Protocol*

RFC 793, *Transmission Control Protocol*

RFC 1213, *Management Information Base for Network Management of TCP/IP-based internets : MIB-II*

RFC 1643, *Definitions of Managed Objects for the Ethernet-like Interface Types*

IETF <draft-ietf-dhc-fqdn-option-01.txt>: March 2, 2001,*The DHCP Client FQDN Option*

SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 4-7: Spécification de protocole de la couche de liaison de données – Éléments de Type 7

AVANT-PROPOS

1) La CEI (Commission Électrotechnique Internationale) 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. À 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 des questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux 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. Toute divergence entre toute Publication de la CEI et toute publication nationale ou régionale correspondante doit être indiquée en termes clairs dans cette dernière.
5) La CEI n'a prévu aucune procédure de marquage valant indication d'approbation et n'engage pas sa responsabilité pour les équipements déclarés conformes à une de ses Publications.
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.
9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de la CEI peuvent faire l'objet de droits de propriété intellectuelle ou de droits analogues. La CEI ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

NOTE L'utilisation de certains des types de protocoles associés est limitée par les détenteurs de leurs droits de propriété intellectuelle. Dans tous les cas, l'engagement à un abandon limité des droits de propriété intellectuelle pris par les détenteurs de ces droits permet d'utiliser un type particulier de protocole de couche de liaison de données avec des protocoles de couche physique et de couche d'application dans des combinaisons de types telles que spécifiées de façon explicite dans la série CEI 61784. L'utilisation des divers types de protocoles dans d'autres combinaisons peut exiger la permission donnée par les détenteurs respectifs de leurs droits de propriété intellectuelle.

La Norme internationale CEI 61158-4-7 a été établie par le sous-comité 65C: Réseaux industriels du comité d'études 65 de la CEI: Mesure, commande et automation dans les processus industriels.

Cette première édition et ses parties associées de la sous-série CEI 61158-4 annulent et remplacent la CEI 61158-4:2003. Cette édition de la présente partie constitue une révision rédactionnelle.

Cette édition de la CEI 61158-4 comporte les modifications significatives suivantes par rapport à l'édition précédente:

a) suppression du précédent bus de terrain Type 6 et du «placeholder» (réceptacle) pour une couche de liaison de données de bus de terrain Type 5, en raison du manque de pertinence commerciale;

b) ajout de nouveaux types de bus de terrain;

c) division de la présente partie en plusieurs parties numérotées -4-1, -4-2, …, -4-19.

La présente version bilingue (2013-09) correspond à la version anglaise monolingue publiée en 2007-12.

Le texte anglais de cette norme est issu des documents 65C/474/FDIS et 65C/485/RVD.

Le rapport de vote 65C/485/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 maintenance indiquée sur le site web de la CEI sous http://webstore.iec.ch dans les données relatives à la publication recherchée. À cette date, la publication sera:

- reconduite;
- supprimée;
- remplacée par une édition révisée, ou
- amendée.

NOTE La révision de la présente norme sera synchronisée avec les autres parties de la série CEI 61158.

La liste de toutes les parties de la série CEI 61158, sous le titre général *Réseaux de communication industriels – Spécifications des bus de terrain*, peut être consultée sur le site web de la CEI.

INTRODUCTION

La présente partie de la CEI 61158 est l'une d'une série produite pour faciliter l'interconnexion de composants de systèmes d'automation. Elle est liée à d'autres normes dans l'ensemble tel que défini par le modèle de référence des bus de terrain "à trois couches" décrit dans la CEI/TR 61158-1.

Le protocole de liaison de données fournit le service de liaison de données en faisant usage des services disponibles à partir de la couche physique. L'objectif principal de la présente norme est de fournir un ensemble de règles pour la communication, exprimées en termes des procédures qui doivent être effectuées par les entités de liaison de données homologues (entités DLE) au moment de la communication. Ces règles pour la communication sont destinées à fournir une base solide pour le développement afin de servir à diverses fins:

a) comme un guide pour les développeurs et les concepteurs;

b) pour utilisation en essai et approvisionnement d'équipements;

c) comme une partie d'un accord pour l'admission des systèmes dans l'environnement à systèmes ouverts;

d) comme un raffinement à la compréhension des communications à temps critiques dans le modèle OSI.

La présente norme porte, en particulier, sur la communication et l'interopérabilité des capteurs, des effecteurs et des autres dispositifs d'automation. Grâce à cette norme et à d'autres normes des modèles de référence OSI ou de bus de terrain, des systèmes par ailleurs incompatibles peuvent fonctionner ensemble, quelle que soit leur combinaison.

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 4-7: Spécification de protocole de la couche de liaison de données – Éléments de Type 7

1 Domaine d'application

1.1 Généralités

La couche de liaison de données assure les communications de messagerie de base en temps critique entre des dispositifs dans un environnement d'automation.

Ce protocole offre des opportunités de communication à toutes les entités de liaison de données participantes

a) d'une manière cyclique avec un début synchrone, conformément à un calendrier préétabli, et

b) d'une manière asynchrone cyclique ou acyclique, comme le requiert chaque cycle pour chacune de ces entités de liaison de données.

Ainsi, ce protocole peut être caractérisé comme protocole qui fournit de façon asynchrone l'accès cyclique et acyclique, mais avec un redémarrage synchrone de chaque cycle.

1.2 Spécifications

La présente norme spécifie les éléments suivants:

a) les procédures de transfert opportun des données et des informations de commande entre une entité utilisateur de liaison de données et une entité utilisateur homologue, mais aussi parmi les entités de liaison de données formant le fournisseur de service de liaison de données distribué;

b) la structure des unités DLPDU de bus de terrain utilisées par le protocole de la présente norme pour le transfert des données et des informations de commande, ainsi que leur représentation sous forme d'unités de données d'interface physique.

1.3 Procédures

Les procédures sont définies en termes

a) d' interactions entre les entités de DL (DLE) homologues à travers l'échange des DLPDU de bus de terrain;

b) d' interactions entre un fournisseur de services de DL (DLS) et un utilisateur de DLS au sein du même système par l'échange de primitives de DLS;

c) d' interactions entre un fournisseur de DLS et un fournisseur de services de Ph au sein du même système par l'échange de primitives de service de Ph.

1.4 Applicabilité

Ces procédures sont applicables aux instances de communication entre des systèmes qui prennent en charge des services de communication en temps critique dans la couche de liaison de données du modèle OSI ou des modèles de référence de bus de terrain, et qui exigent la capacité à s'interconnecter dans un environnement d'interconnexion de systèmes ouverts.

Les profils constituent un moyen simple à plusieurs attributs de récapituler les capacités d'une mise en œuvre et donc son applicabilité en fonction des différents besoins de communication en temps critique.

1.5 Conformité

La présente norme spécifie également les exigences de conformité relatives aux systèmes mettant en œuvre ces procédures. La présente norme ne contient aucun essais visant à démontrer la conformité à ces exigences.

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).

IEC 61158-2 (Ed.4.0), *Industrial communication networks – Fieldbus specifications – Part 2: Physical layer specification and service definition* (disponible en anglais uniquement)

IEC 61158-3-7, *Industrial communication networks – Fieldbus specifications – Part 3-7: Data link service definition – Type 7 elements* (disponible en anglais uniquement)

ISO/CEI 7498-1, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base: Le modèle de base*

ISO/CEI 7498-3, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base: Dénomination et adressage*

ISO/CEI 10731, *Technologie de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base – Conventions pour la définition des services OSI*

3 Termes, définitions, symboles et abréviations

Pour les besoins du présent document, les termes, définitions, symboles et abréviations suivants 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 dans l'ISO/CEI 7498-3, et utilise les termes suivants qui y sont définis.

3.1.1 entités (N) correspondantes [7498-1]
entités de DL correspondantes (N=2)
entités de Ph correspondantes (N=1)

3.1.2 adresse de DL [7498-3]

3.1.3 connexion de DL [7498-1]

3.1.4 extrémité de connexion de DL [7498-1]

3.1.5 identificateur d'extrémité de connexion de DL [7498-1]

3.1.6 nom de DL [7498-3]

3.1.7 protocole de DL [7498-1]

3.1.8 **identificateur de connexion de protocole de DL** [7498-1]

3.1.9 **Information de contrôle de protocole de DL** [7498-1]

3.1.10 **unité de données de protocole de DL** [7498-1]

3.1.11 **relais de DL** [7498-1]

3.1.12 **identificateur de connexion de service de DL** [7498-1]

3.1.13 **unité de données de service de DL** [7498-1]

3.1.14 **données d'utilisateur de DL** [7498-1]

3.1.15 **contrôle de flux** [7498-1]

3.1.16 **entité (N)** [7498-1]
entité de DL
entité de Ph

3.1.17 **unité de données d'interface (N)** [7498-1]
unité de données de service de DL (N=2)
unité de données d'interface Ph (N=1)

3.1.18 **couche (N)** [7498-1]
couche DL (N=2)
couche Ph (N=1)

3.1.19 **service (N)** [7498-1]
service de DL (N=2)
service de Ph (N=1)

3.1.20 **point d'accès au service (N)** [7498-1]
point d'accès au service de DL (N=2)
point d'accès au service de Ph (N=1)

3.1.21 **adresse de point d'accès au service (N)** [7498-1]
adresse de point d'accès au service de DL (N=2)
adresse de point d'accès au service de Ph (N=1)

3.1.22 **entités homologues** [7498-1]

3.1.23 **information de contrôle d'interface Ph** [7498-1]

3.1.24 **données d'interface Ph** [7498-1]

3.1.25 **nom de primitive** [7498-3]

3.1.26 **reset (réinitialisation)** [7498-1]

3.1.27 **adresse de DL en réponse** [7498-3]

3.1.28 **routage** [7498-1]

3.1.29 **segmentation** [7498-1]

3.1.30 **ordonnancement** [7498-1]

3.1.31 **gestion de système** [7498-1]
gestion de systèmes

3.2 Termes et définitions de convention pour les services

La présente norme utilise également les termes suivants définis dans l'ISO/CEI 10731 tels qu'ils s'appliquent à la couche de liaison de données:

3.2.1 **"confirm" (primitive);**
"requestor.deliver" (primitive)

3.2.2 **"deliver" (primitive)**

3.2.3 **primitive de service de DL**;
primitive

3.2.4 **fournisseur de service de DL**

3.2.5 **utilisateur de service de DL**

3.2.6 **fonctionnalité facultative d'utilisateur de DL**

3.2.7 **"indication" (primitive)**
"acceptor.deliver" (primitive)

3.2.8 **homologues multiples**

3.2.9 **"request" (primitive);**
"requestor.submit" (primitive)

3.2.10 **demandeur**

3.2.11 **"response" (primitive);**
"acceptor.submit" (primitive)

3.2.12 **"submit" (primitive)**

3.3 Autres termes et définitions

NOTE Plusieurs définitions sont communes à plus d'un type de protocole; elles ne sont pas forcément utilisées par tous les types de protocoles.

Pour les besoins de la présente partie de la CEI 61158, les définitions suivantes s'appliquent également:

3.3.1
DLPDU de réponse d'acquittement
information que le destinataire d'un message avec acquittement émet afin de signaler soit la réception correcte du message soit le manque de ressources disponibles pour stocker le message reçu par la DLE sur la liaison locale qui a émis le message et qui a demandé l'acquittement

3.3.2
cycle de base
séquence de balayage par l'administrateur de bus:

a) d'un ensemble d'identificateurs de DLCEP pour les variables, les demandes et les messages d'applications cycliques,
b) plus la fenêtre prévue pour les échanges apériodiques,
c) plus la fenêtre prévue pour les services de messages,
d) plus la fenêtre prévue pour la synchronisation

3.3.3
transaction de base
succession de DLPDU liées à une instance de service de DL unique

3.3.4
administrateur de bus (BA, bus-arbitrator)
DLE qui commande le droit d'accès au support de chaque producteur de données

NOTE À un instant donné, un et un seul administrateur de bus est actif dans chaque segment de DL dans un sous-réseau de DL.

3.3.5
variable identifiée consommée
variable identifiée qui correspond à un identificateur de DLCEP pour lequel l'entité en question reçoit les données

3.3.6
champ de contrôle
partie d'une DLPDU émise ou reçue qui fournit la nature des données échangées et le type de l'échange

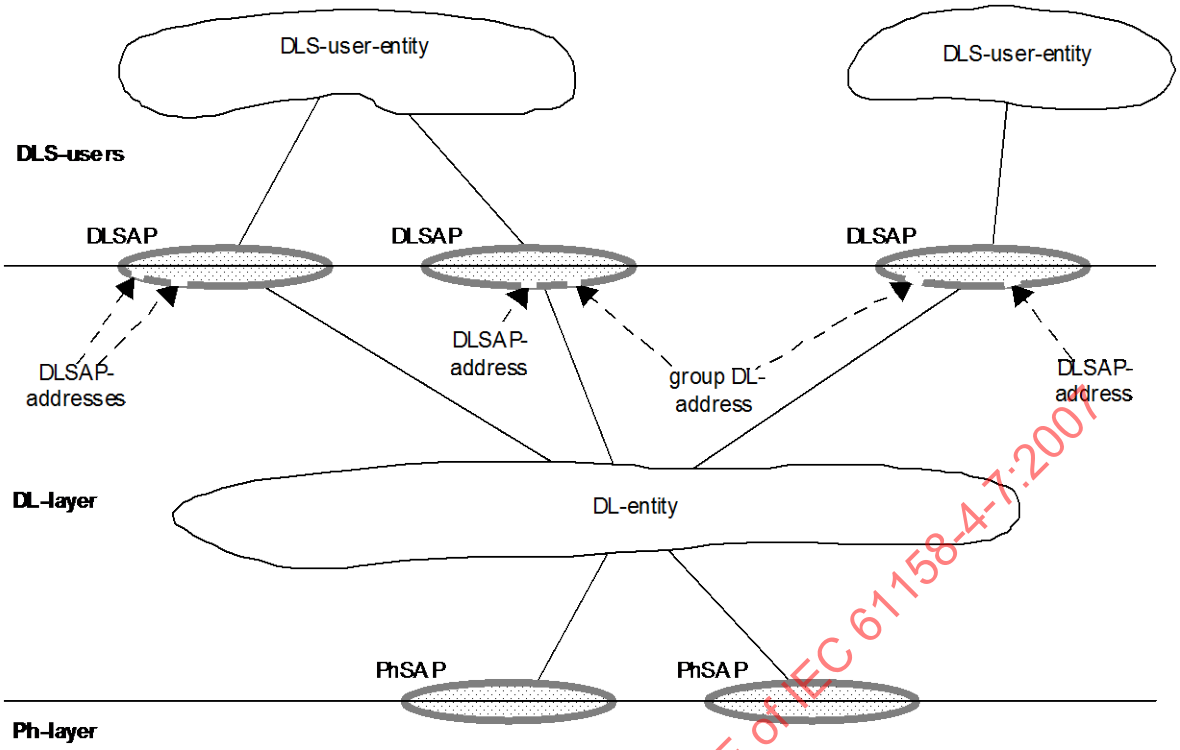
3.3.7
adresse de destination
trois octets spécifiant le segment de DL de la DLE à laquelle le message est envoyé, et la sous-adresse du DLSAP de destination dans le segment de DL de la liaison locale

3.3.8
adresse de DLCEP
information que l'administrateur de bus émet pour allouer le support à un producteur de données dans le but d'échanger une variable

3.3.9
segment de DL, liaison, liaison locale
sous-réseau de DL unique dans lequel toutes les éventuelles DLE connectées peuvent communiquer directement, sans intervention de relayage de DL, chaque fois que toutes celles des DLE qui participent à une instance de communication sont simultanément attentives au sous-réseau de DL pendant la/les période(s) de communication tentée(s)

3.3.10
DLSAP
point distinctif dans lequel des services de DL sont fournis par une entité de DL unique à une unique entité de couche supérieure

NOTE Cette définition, dérivée de l'ISO/CEI 7498-1, est reprise ici pour faciliter la compréhension de la distinction critique entre les DLSAP et leurs adresses de DL. (Voir Figure 1.)



NOTE 1 Les DLSAP et les PhSAP sont illustrés sous la forme d'ovales enjambant la frontière entre deux couches adjacentes.

NOTE 2 Les adresses de DL sont illustrées comme désignant de petits espaces (points d'accès) dans la partie DLL d'un DLSAP.

NOTE 3 Une entité de DL unique peut avoir plusieurs adresses de DLSAP et adresses de DL de groupe associées à un même DLSAP.

Légende

Anglais	Français
DLS-user-entity	Entité d'utilisateur de DL
DLS-users	Utilisateurs de DL
DLSAP address	Adresse de DLSAP
DLSAP addresses	Adresses de DLSAP
Group DL-address	Adresse de DL de groupe
DL-entity	Entité de DL
DL-layer	Couche DL (Couche de liaison de données)
Ph-layer	Couche Ph (Couche physique)

Figure 1 – Relations des DLSAP, des adresses de DLSAP et des adresses de DL de groupe

3.3.11
adresse de DL(SAP)

soit une adresse de DLSAP individuelle, désignant un unique DLSAP d'un unique utilisateur de DLS, soit une adresse de DL de groupe désignant potentiellement plusieurs DLSAP, chacun d'un unique utilisateur de DLS

NOTE Cette terminologie est choisie parce que l'ISO/CEI 7498-3 ne permet pas l'utilisation du terme "adresse de DLSAP" pour désigner plus d'un seul DLSAP au niveau d'un utilisateur de DLS unique.

3.3.12
adresse de DLSAP (individuelle)
adresse de DL qui désigne un seul DLSAP au sein de la liaison étendue

NOTE Une entité de DL unique peut avoir plusieurs adresses de DLSAP associées à un seul DLSAP.

3.3.13
DLPDU d'indication de fin de transaction de message
information que l'entité source d'un message émet afin de passer le contrôle d'accès de la liaison à l'administrateur de bus à la fin de transaction de message

3.3.14
liaison étendue
sous-réseau de DL, constitué de l'ensemble maximal de liaisons interconnectées par des relais de DL, partageant un seul espace de noms de DL (d'adresses de DL), dans lequel toute entité de DL connectée peut communiquer, l'une avec l'autre, soit directement, soit à l'aide d'une ou de plusieurs entités relais de DL intermédiaires

NOTE Une liaison étendue peut être composée juste d'une seule liaison.

3.3.15
trame
synonyme déconseillé de DLPDU

3.3.16
adresse de DL de groupe
adresse de DL qui désigne potentiellement plus d'un DLSAP au sein de la liaison étendue Une entité de DL unique peut avoir plusieurs adresses de DL de groupe associées à un DLSAP unique. Une entité de DL unique peut avoir aussi une unique adresse de DL de groupe associée à plus d'un DLSAP

3.3.17
variable identifiée (ou simplement "variable")
variable de DLL (mémoire tampon) pour laquelle un identificateur de DLCEP associé a été défini

3.3.18
identificateur
mot de 16 bits associé à une variable système. Un identificateur de DLCEP désigne de manière unique une seule variable au sein du sous-réseau de DL

3.3.19
identificateur de DLCEP non valide
identificateur non reconnu localement

3.3.20
liaison locale
ensemble de dispositifs qui respectent le protocole de DL et qui sont interconnectés à travers un support. Un seul administrateur de bus est actif sur une seule et même liaison locale

3.3.21
macrocycle
ensemble de cycles de base nécessaires pour tous les identificateurs de DLCEP cycliques qui doivent être balayés

3.3.22
adresse de DL de message
information que l'administrateur de bus émet pour allouer le support à une entité source pour un transfert de messages

3.3.23
DLPDU de réponse de message
information qu'un producteur de données émet en réponse à une DLPDU d'identificateur de DLCEP de message

NOTE L'entité ou les entités de destination désirée(s) reçoivent cette information.

3.3.24
nœud
simple entité de DL telle qu'elle apparaît sur une liaison locale

3.3.25
balayage périodique de variables
action par l'administrateur de bus qui garantit l'échange cyclique de variables

NOTE C'est le principe de base du protocole et du service de DL de Type 7.

3.3.26
variable identifiée produite
variable identifiée qui correspond à un identificateur de DLCEP pour lequel la DLE émet des données

3.3.27
utilisateur de DLS destinataire
utilisateur de service de DL qui agit comme un destinataire de données d'utilisateur de DL

NOTE Un utilisateur de service de DL peut être simultanément un utilisateur de DLS expéditeur et destinataire.

3.3.28
adresse de DLCEP de demande
information que l'administrateur de bus émet pour allouer le support à l'émetteur d'une demande explicite pour un échange de variables

3.3.29
DLPDU de réponse de demande
information que l'émetteur d'une demande explicite pour un échange de variable émet en réponse à une DLPDU d'identificateur de DLCEP de demande, et auquel envoi l'administrateur de bus répond également

3.3.30
utilisateur de DLS expéditeur
utilisateur de service de DL qui agit comme une source de données d'utilisateur de DL

3.3.31
adresse source
mot de 24 bits comportant le numéro de segment de DL de l'entité envoyant le message, et la sous-adresse de l'entité dans le segment de DL

3.3.32
temporisateurs
voir 5.6.

3.3.33
balayage déclenché de messages
fonction d'un administrateur de bus qui permet de transférer des messages

3.3.34
balayage déclenché de variables
fonction d'un administrateur de bus qui permet l'échange non cyclique de variables

3.3.35
balayage périodique déclenché de messages
fonction d'un administrateur de bus qui permet de demander cycliquement des échanges de messages déclenchés

3.3.36
balayage périodique déclenché de variables
fonction d'un administrateur de bus qui permet de demander cycliquement des transferts déclenchés de variables

3.3.37
temps de changement
intervalle de temps entre la réception ou l'émission du dernier symbole MAC d'une DLPDU, signalée par une indication SILENCE de la couche physique (PhL), et la réception ou l'émission du premier symbole MAC de la DLPDU suivante, signalée par une indication ACTIVITY de la couche physique (PhL), les deux telles que mesurées dans une station donnée

3.3.38
DLPDU de réponse variable
information qu'un producteur de données émet en réponse à une DLPDU d'identificateur de DLCEP, qui alerte aussi les consommateurs de données de la pertinence de la DLPDU immédiatement la plus proche en terme de temps

3.4 Symboles et abréviations

3.4.1	**BA**	bus-arbitrator (administrateur de bus)
3.4.2	**B_Dat_Cons**	mémoire tampon qui contient la valeur de la donnée consommée
3.4.3	**B_Dat_Prod**	mémoire tampon qui contient la valeur de la donnée produite
3.4.4	**B_Req1/2**	mémoire tampon contenant la liste d'identificateurs de DLCEP qui font l'objet d'une demande explicite spécifiée pour un transfert à la priorité 1 (urgent) ou à la priorité 2 (normal)
3.4.5	**ID_DAT**	DLPDU utilisée pour allouer le support à un transfert de mémoire tampon
3.4.6	**ID_MSG**	DLPDU utilisée pour allouer le support à un échange de message
3.4.7	**ID_RQ1/2**	DLPDU utilisée pour allouer le support à une demande pour un transfert de mémoire tampon
3.4.8	**PRT**	physical reaction time (temps de réaction physique)
3.4.9	**Q_IDMSG**	file d'attente d'identificateurs de DLCEP demandés reçus par l'administrateur de bus pour un transfert de message
3.4.10	**Q_IDRQ1/2**	file d'attente d'identificateurs de DLCEP demandés reçus par l'administrateur de bus à la priorité 1 (urgent) ou à la priorité 2 (normal)
3.4.11	**Q_Msg_Cyc**	file d'attente contenant des messages à émettre qui sont associés aux échanges cycliques
3.4.12	**Q_Msg_Aper**	file d'attente contenant des messages à émettre qui sont associés aux échanges apériodiques

3.4.13 Q_Req1/2 file d'attente contenant la liste d'identificateurs de DLCEP qui font l'objet d'une demande explicite libre pour un transfert à la priorité 1 (urgent) ou à la priorité 2 (normal)

3.4.14 Q_RPRQ file d'attente pour les transferts apériodiques en cours

3.4.15 RQ_Inhibit Indicateur utilisé pour gérer une demande explicite pour des échanges de variables

3.4.16 RP_ACK DLPDU utilisée pour transférer un acquittement d'échange de messages, Transfert OK

3.4.17 RP_DAT DLPDU utilisée pour transporter la valeur de la variable identifiée précédemment demandée

3.4.18 RP_DAT_MSG DLPDU utilisée pour transporter la valeur de la variable identifiée précédemment demandée avec une demande pour un échange de messages

3.4.19 RP_DAT_RQ1/2 DLPDU utilisée pour transporter la valeur de la variable identifiée précédemment demandée avec une demande pour un transfert explicite de variables

3.4.20 RP_DAT_RQ1/2_MSG DLPDU utilisée pour transporter la valeur de la variable identifiée précédemment demandée avec une demande pour un transfert explicite de variables et une demande pour un transfert de messages

3.4.21 RP_MSG_ACK DLPDU utilisée pour transporter le message à échanger avec une demande d'acquittement de cet échange

3.4.22 RP_MSG_NOACK DLPDU utilisée pour transporter le message à échanger sans une demande d'acquittement de cet échange

3.4.23 RP_NAK DLPDU utilisée pour transférer un acquittement d'échange de messages, que le message a correctement été reçu, mais n'a pas été stocké par la DLE

3.4.24 RP_END DLPDU utilisée pour terminer un échange de messages

3.4.25 STT station turnaround time (temps de changement de station)

3.4.26 TI temps de latence

4 Vue d'ensemble du protocole de DL

4.1 Description générale d'allocation de support

Un élément connu sous le nom de «**administrateur de bus» (BA)** commande le droit de chaque producteur de données à accéder au support en émettant une DLPDU contenant un identificateur de DLCEP. À un instant donné, il convient de trouver un seul administrateur de bus actif dans chaque liaison locale.

NOTE Le terme "producteur de données" désigne la station unique connectée à la liaison locale qui est reconnue comme ayant la responsabilité d'émettre les données associées à la DLPDU identificatrice circulant sur le support.

Chaque transaction appartient à l'une des trois classes d'allocation du support définies ci-dessous:

— échange cyclique de variables, demandes ou messages,

— demande explicite pour échange de variables,

— demande explicite pour transfert de messages.

Pour les échanges de variables cycliques, une transaction de base est composée des phases suivantes. L'administrateur de bus diffuse une DLPDU d'identificateur de variable. Le seul producteur de l'information requise diffuse alors une DLPDU de réponse de variable. Durant cette phase, les consommateurs prennent l'information à partir de la liaison locale. La Figure 2 montre les différentes phases d'une transaction d'échanges de variables. Lorsqu'une transaction a été achevée, l'administrateur de bus commence la transaction suivante conformément aux directives définies lorsque le système est configuré.

Au cours d'un échange de variables cyclique, le producteur d'informations peut, en utilisant la DLPDU de réponse, émettre à l'administrateur de bus une demande explicite pour l'échange de variables ou de messages.

Pour une demande explicite pour un échange de variables, une transaction de base est composée des phases suivantes: L'administrateur de bus diffuse une DLPDU d'identificateur de demande. Ensuite l'émetteur de la demande diffuse une DLPDU de réponse à une demande. Puis se suivent une ou plusieurs transactions identiques à la transaction d'échange de variables cyclique.

Pour des transferts de message, une transaction de base est constituée des phases suivantes: L'administrateur de bus diffuse une DLPDU d'identificateur de message. Ensuite, la DLPDU de réponse de message est échangée entre les entités communicantes. Cette DLPDU peut ou peut ne pas être suivie par une DLPDU d'acquittement. Puis, l'entité source émet à l'administrateur de bus une DLPDU d'indication de fin de transaction.

L'intervalle de temps séparant la réception ou l'émission de la fin d'une DLPDU et l'émission ou la réception de la DLPDU suivante est connu sous le nom de "temps de changement" de la station, que la fonction de la station soit celle d'un producteur/consommateur ou d'un administrateur de bus.

Une définition plus détaillée du temps de changement est donnée en 3.3.37. L'impact du temps de changement sur les temporisateurs de liaison de données est décrit en 5.6.2.

Le rôle d'un administrateur de bus est de "donner la parole" à chaque producteur de données, en tenant compte des services requis pour le type de données (selon les trois classes d'allocation du support définies ci-dessus).

L'administrateur de bus a ainsi trois types de fonctions:

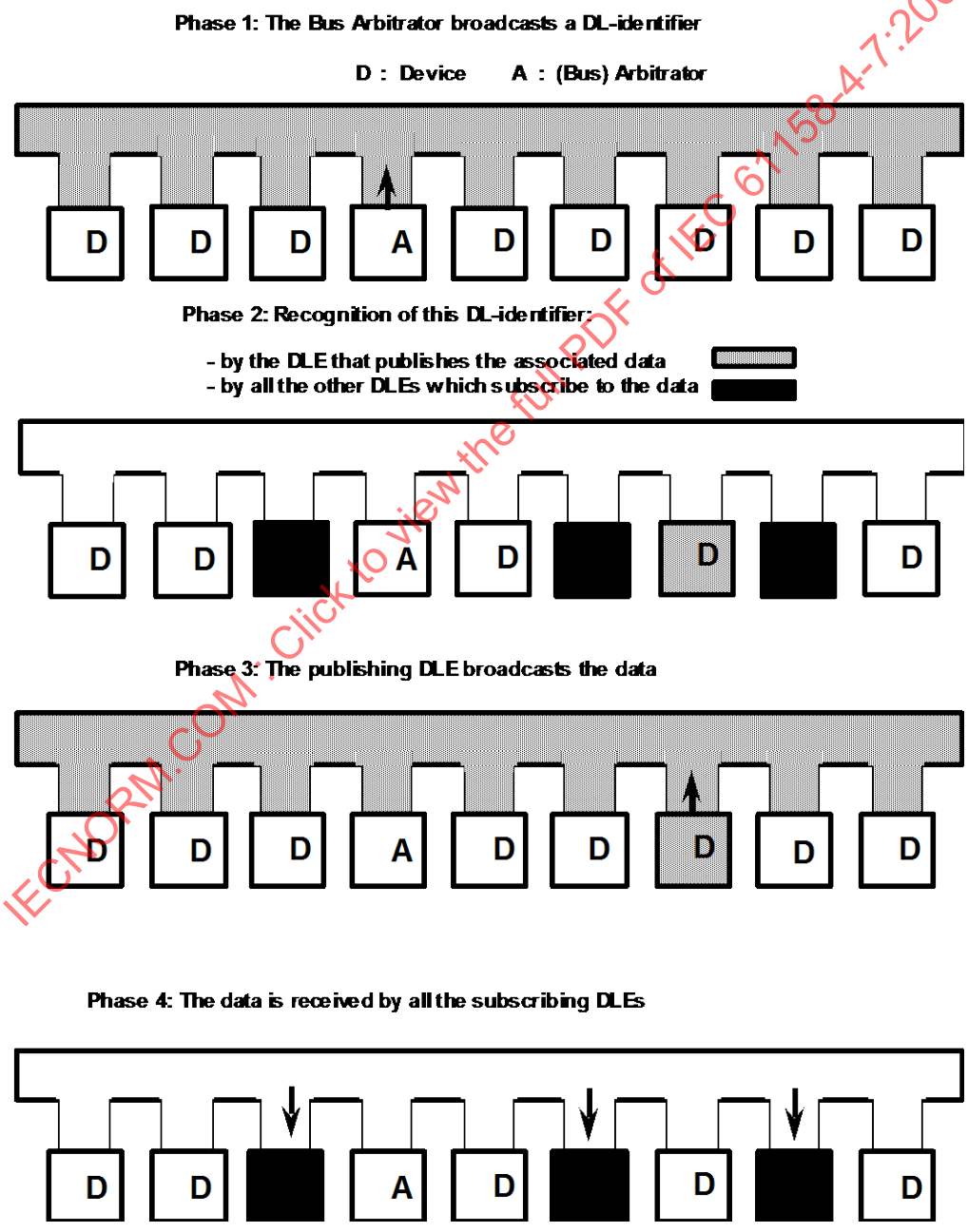
— balayage périodique de variables ou déclenchement périodique d'échanges de variables et de messages,

— balayage déclenché de variables,

— balayage déclenché de messages.

De plus, l'administrateur de bus peut fournir une fonction de synchronisation afin de garantir la longueur constante des cycles de balayage.

Chaque type de balayage a lieu dans une fenêtre spécifique, qui est, respectivement dans une fenêtre périodique, dans une fenêtre apériodique de variables, dans une fenêtre apériodique de messages ou dans une fenêtre de synchronisation, comme montré à la Figure 2. Ces quatre fenêtres constituent un cycle de balayage de base.



Légende

Anglais	Français
Phase 1: The Bus Arbitrator broadcasts a DL	Phase 1: L'administrateur de bus diffuse un

Anglais	Français
identifier	identificateur de DL
D : Device	D : Dispositif
A : (Bus) Arbitrator	A : Administrateur (de bus)
Phase 2: Recognition of this DL identifier	Phase 2: Reconnaissance de cet identificateur de DL
by the DLE that publishes the associated data	par la DLE qui publie les données associées
by all the other DLEs which subscribe to the data	par toutes les autres DLE qui s'abonnent aux données
Phase 3: The publishing DLE broadcasts the data	Phase 3: La DLE éditrice diffuse les données
Phase 4: The data is received by all the subscribing DLEs	Phase 4: Les données sont reçues par toutes les DLE abonnées

Figure 2 – Description générale d'allocation de support

La technique d'accès au support a les caractéristiques suivantes:

— diffusion de variables identifiées,

— efficacité croissante dans les échanges de variables cycliques,

— les paramètres relatifs au partage de support peuvent être configurés par l'utilisateur lorsque le système est configuré,

— un temps d'accès garanti pour les échanges cycliques de variables, en toutes circonstances et indépendamment du nombre de demandes pour les échanges de variables et/ou messages déclenchés,

— possibilité de déclencher une transaction, conformément à une horloge globale, qui est une horloge qui indique le même temps pour toutes les stations.

De plus, la technique d'accès au support permet de:

— donner aux échanges cycliques la plus haute priorité,

— respecter la période de balayage associée à chaque variable,

— donner différentes priorités aux transferts de messages déclenchés et aux échanges de variables. Ces transactions sont déclenchées dans des fenêtres adaptables: les longueurs des fenêtres de "variables apériodiques" et "messages apériodiques" sont définies en termes de limites maximales fixées par l'utilisateur lorsque le système est configuré,

— changer la priorité des transactions apériodiques en les insérant dans la fenêtre périodique.

4.2 Types d'entités

4.2.1 Entité productrice/consommatrice

Les services DLL utilisent plusieurs types de DLPDU. Chaque type de DLPDU est défini dans le champ de contrôle de la DLPDU. Les interactions entre un service spécifique et les types de DLPDU seront décrites dans les spécifications particulières de chaque service.

NOTE Le paragraphe 5.5 décrit tous les types de DLPDU et explique aussi l'utilisation de divers champs.

La fonction d'une entité appartenant à la plus haute classe de conformité exige:

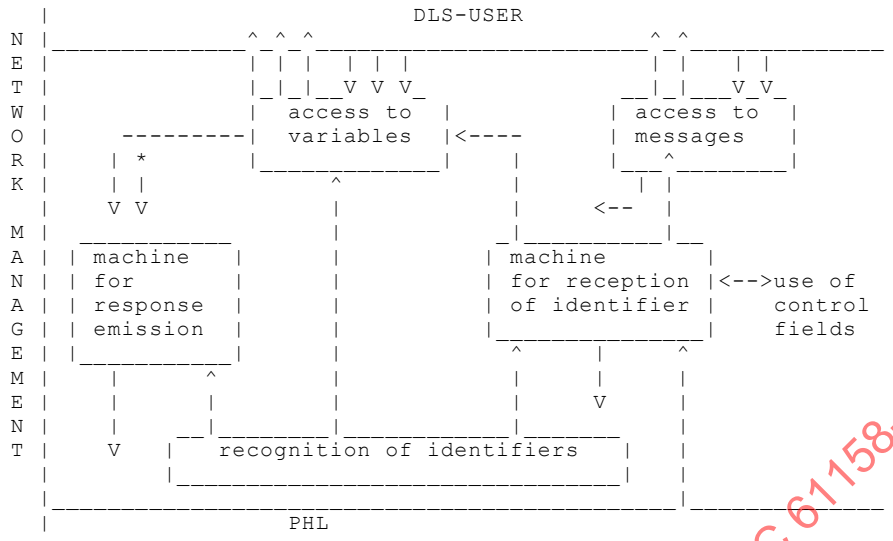
— un mécanisme pour analyser le type de DLPDU (utilisation du champ de contrôle),

— une table d'identificateurs reconnus lors de l'émission,

— une table d'identificateurs reconnus lors de la réception (les deux tables sont définies à l'interface entre la DLL et la gestion de système),

un mécanisme qui fournit un accès en lecture/écriture aux variables et aux messages.

Le modèle conceptuel d'une entité productrice/consommatrice de liaison de données inclut plusieurs tampons: les files d'attente nécessaires pour fournir les services offerts par la DLL, comme montré dans la Figure 3.



Légende

Anglais	Français
DLS-USER	Utilisateur de DLS
Network management	Gestion du réseau
access to variables	Accès aux variables
access to messages	Accès aux messages
Machine for response emission	Diagramme pour émission de réponse
Machine for reception of identifier	Diagramme pour réception d'identificateur
Use of control fields	Utilisation des champs de contrôle
recognition of identifiers	Reconnaissance d'identificateurs

Figure 3 – Structure interne d'une entité productrice/consommatrice

Les éléments suivants sont associés à chaque identificateur produit:

— une mémoire tampon appelée B_DATprod, qui contient la valeur de la variable produite,

— une mémoire tampon appelée B_REQ, contenant la liste d'identificateurs qui sont l'objet d'une demande explicite pour un transfert de mémoire tampon, si l'identificateur a été réservé pour la demande explicite pour le service transfert de mémoire tampon,

— une file d'attente appelée Q_MSGcyc, contenant des messages à émettre qui sont associés au transfert cyclique de messages, si l'identificateur a été réservé pour le transfert cyclique de messages,

— un indicateur de validité de l'identificateur produit (gestion de système),

— pour chacun des tampons B_DATprod et B_REQ , un indicateur de disponibilité,

— un indicateur associé à Q_MSGcyc: file d'attente pleine ou pas,

— indicateurs utilisés pour gérer des demandes explicites pour des demandes d'échanges de variables et de transferts de messages:

— demande explicite pour le transfert de mémoire tampon en cours (RQ),

— priorité associée à une demande explicite (PR),

— domaine d'application de la demande explicite (RQ_INHIBIT),

— demande de transfert de message en cours (MSG) si l'identificateur a été réservé pour le transfert apériodique de messages,

— référence à la file d'attente des messages à émettre associée au transfert apériodique de messages.

Les éléments suivants sont associés à chaque identificateur consommé:

— une mémoire tampon appelée B_DATcons, qui contient la valeur de la variable consommée,

— un indicateur de validité de l'identificateur consommé (gestion de système),

— un indicateur lié à la gestion de B_DATcons: disponibilité de la mémoire tampon (conflit d'accès).

NOTE L'indicateur de disponibilité de la mémoire tampon fournit des informations relatives à l'intégrité des données manipulées par des primitives de service.

Chaque entité qui prend en charge un service de demande de transfert de message utilise aussi:

— une file d'attente appelée Q_MSGaper, qui est associée au transfert apériodique de messages et elle contient des messages à émettre,

— une file d'attente appelée Q_MSGrec qui contient les messages reçus,

— un indicateur du statut de la file d'attente (pleine ou pas) est associé à la file d'attente réservée pour le transfert apériodique de messages.

De plus, chaque entité qui prend en charge un service de message avec acquittement a les caractéristiques suivantes:

— valeur du bit pair/impair de l'entité source,

— valeur maximale du compteur de redémarrage,

— valeur courante du compteur de redémarrage.

Les paramètres suivants sont associés à la file d'attente Q_MSGrec:

— un indicateur du statut de la file d'attente (pleine ou pas),

— valeur du bit pair/impair de l'entité destination,

— valeur de l'adresse source stockée.

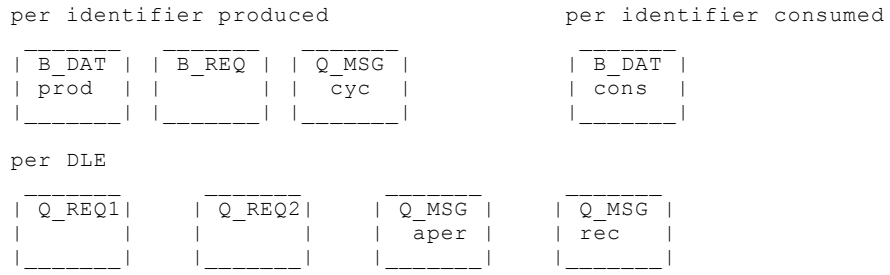
Si l'entité prend en charge aussi le service de demande explicite libre pour un transfert de mémoire tampon, il y a deux files d'attente générales pour contenir les demandes explicites libres pour un transfert de mémoire tampon:

— Q_REQ1 est associé à des demandes urgentes,

— Q_REQ2 est associé à des demandes normales.

Un indicateur de statut (pleine ou pas) est associé à chaque file d'attente.

La Figure 4 montre plusieurs de ces mémoires tampons et files d'attente associées.



Légende

Anglais	Français
per identifier produced	par identificateur produit
per identifier consumed	par identificateur consommé
per DLE	par DLE

Figure 4 – Mémoires tampons et files d'attente associées

Les temporisateurs associés au protocole de liaison de données sont aussi gérés par l'entité productrice/consommatrice (voir 5.6).

L'Annexe B présente un modèle d'objet qui définit les attributs d'un identificateur de DLCEP.

4.2.2 Entité administrateur de bus

La fonction fournie par l'administrateur de bus consiste à "accorder la permission de parler" à chaque producteur de données. Cette permission est accordée pour trois types de balayage:

— balayage périodique ou balayage périodique déclenché de variables et de messages,
— balayage déclenché de variables,
— balayage déclenché de messages.

L'administrateur de bus fournit aussi les fonctions suivantes:

— chaînage des transactions de base,
— chaînage des différents types de balayage,
— analyse des champs de contrôle des DLPDU (le champ de contrôle porte les demandes de messages et de variables apériodiques),
— le remplissage des fenêtres apériodiques conformément à ces demandes.

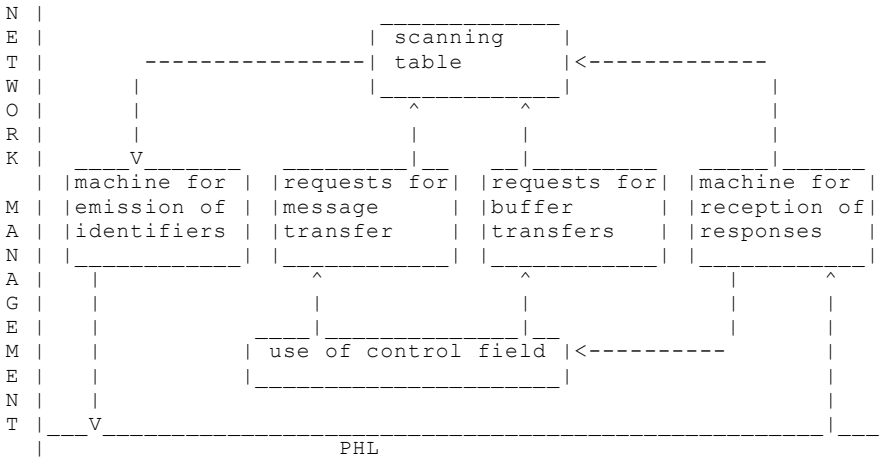
NOTE Une transaction de base est composée de la succession de DLPDU liées à un service unique.

De plus, l'administrateur de bus peut fournir une fonction de synchronisation pour garantir la longueur constante d'un cycle de balayage de base.

Chaque type de balayage a lieu dans une fenêtre particulière: respectivement dans une fenêtre périodique, dans une fenêtre apériodique de variables, dans une fenêtre apériodique de messages, ou dans une fenêtre de synchronisation.

Les quatre fenêtres constituent un cycle de balayage de base.

La Figure 5 donne une vue d'ensemble de la structure de l'administrateur de bus. Une description plus complète de l'administrateur de bus est donnée en 7.4.



Légende

Anglais	Français
scanning table	Table de balayage
Network management	Gestion du réseau
Machine for emission of identifiers	Diagramme pour émission d'identificateurs
requests for message transfer	Demandes pour transfert de messages
requests for buffer transfers	Demandes pour transferts de mémoires tampon
Machine for reception of responses	Diagramme pour réception de réponses
Use of control fields	Utilisation des champs de contrôle
PHL	PHL (couche physique)

Figure 5 – Structure interne d'un administrateur de bus

Le modèle conceptuel de l'administrateur de bus détaille les différentes tables et files d'attente nécessaires à fournir les services offerts par la DLL.

L'administrateur de bus gère conformément à la configuration de système, comme montré à la Figure 6:

— une table de balayage avec des parties statiques et des parties qui sont modifiées dynamiquement pendant le balayage,

— une mémoire tampon de rétablissement pour le balayage périodique déclenché de variables,

— une file d'attente appelée Q_IDRQ1 pour les demandes explicites urgentes pour le transfert de mémoire tampon,

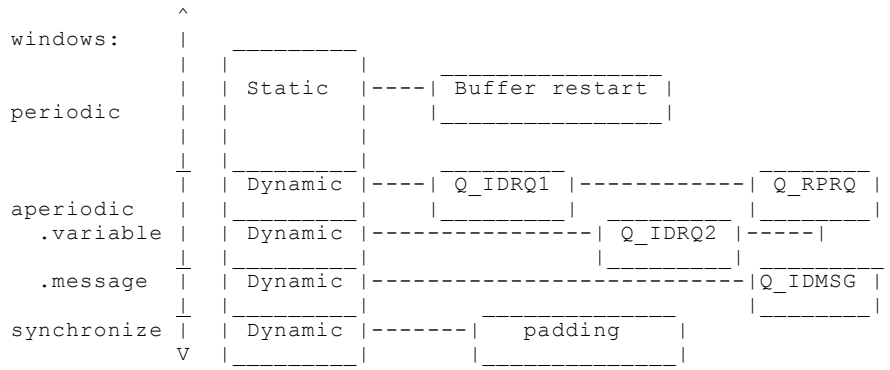
— une file d'attente appelée Q_IDRQ2 pour les demandes explicites normales pour le transfert de mémoire tampon,

— une file d'attente pour les transferts apériodiques en cours (Q_RPRQ),

— une file d'attente appelée Q_IDMSG pour les demandes pour le transfert de messages,

— une séquence de remplissage de base utilisée pour remplir la fenêtre de synchronisation.

NOTE Il convient que l'algorithme et/ou le calcul de gestion de ces tables fournisse(nt) des droits d'accès égaux pour les différentes entités, car il convient d'éviter la famine de certaines entités.



Légende

Anglais	Français
windows	fenêtre
periodic	périodique
aperiodic	apériodique
variable	variable
message	message
synchronize	synchroniser
Static	Statique
Dynamic	Dynamique
Buffer restart	Redémarrage de mémoire tampon
padding	remplissage/bourrage

Figure 6 – Table d'échange sur interrogation de l'administrateur de bus

4.3 Adressage

4.3.1 Relation d'utilisateur de DLSAP

Le rôle de la DLL est d'émettre les informations d'une manière fiable et conformément au protocole d'émission.

Dans une seule station physique, plusieurs entités d'utilisateur ont accès aux services de DLL. Chacune de ces entités utilise un point d'accès au service (SAP) appartenant à la DLL, en conformité avec le modèle statique de l'ISO dans lequel chaque entité (N+1) est reconnue par l'adresse du point d'accès au service (N) auquel l'entité est statiquement attachée.

Plusieurs associations entre les entités d'utilisateur sont possibles au niveau de points d'accès aux services de DLL.

Ainsi, conformément au modèle ISO, une entité d'utilisateur (N+1) demande l'établissement d'une connexion (N) afin de communiquer avec une autre entité d'utilisateur (N+1). Les (N) entités, dont chacune est liée à l'une des deux (N+1) entités, créent et gèrent cette connexion après qu'elles la négocient.

Tous les échanges entre les entités (N+1) ainsi associées ont lieu en utilisant cette connexion. La connexion est identifiée localement à chaque SAP par un identificateur de point d'extrémité de connexion spécifique («connection end point identifier» (CEPi)).

Le nombre de DLSAP correspond au nombre d'entités dans une station qui sont les utilisateurs potentiels des services de DLL.

Le nombre de CEPi par DLSAP correspond au nombre de liaisons de communication potentielles allouées aux (N+1) entités liées au DLSAP.

4.3.2 Relation avec le modèle OSI

Cette DLL suit les mêmes règles que l'ISO/CEI 7498. Chaque entité d'utilisateur potentielle appelle auprès des services de la DLL à travers un DLSAP.

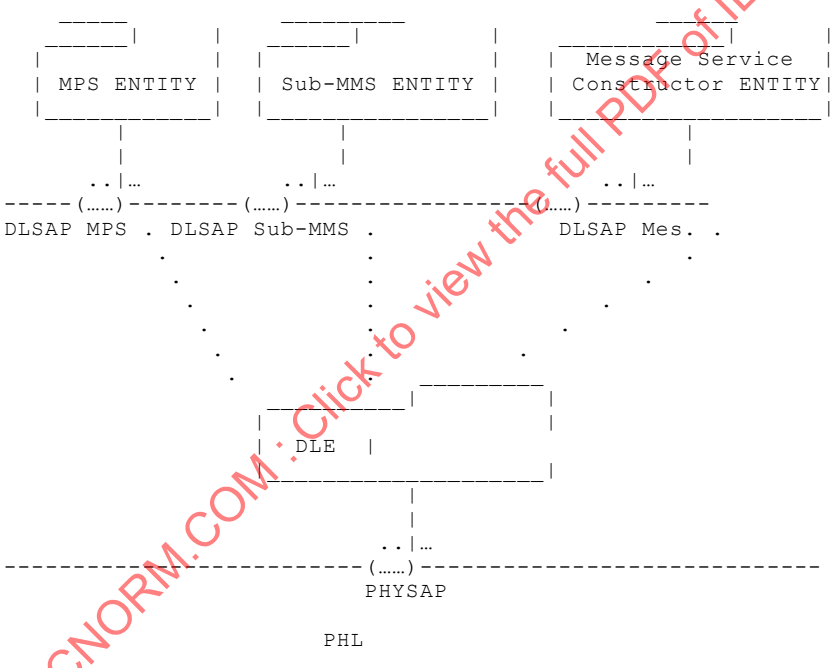
La DLL place deux espaces d'adressage distincts comme la disposition de l'utilisateur de DLS:

— l'espace d'adressage relatif au transfert de mémoire tampon, chaque identificateur étant codé sur 16 bits,

— l'espace d'adressage relatif aux échanges de messages, ce qui permet l'adressage des utilisateurs de DLS de messagerie, chaque adresse étant codée sur 24 bits.

Les identificateurs, codés sur 16 bits, permettent l'adressage des mémoires tampons au sein d'une liaison locale, alors que les adresses de messagerie, codées sur 24 bits sont destinées à identifier tous les utilisateurs de DLS de messagerie du sous-réseau de DL, qui est celles de tous les segments de DL du sous-réseau de DL. Par conséquent, un système de communication composé de n liaisons locales aura un seul espace d'adresses de messagerie, mais aura n espaces d'identificateurs pour attribuer les adresses aux tampons.

La Figure 7 illustre cet usage.



Légende

Anglais	Français
MPS ENTITY	Entité MPS
Sub-MMS ENTITY	Entité Sub-MMS
Message Service Constructor ENTITY	Entité constructrice de service de message

Figure 7 – Plan d'adressage

4.3.2.1 Adressage de transfert de mémoire tampon

Pour des raisons liées à la performance et au déterminisme, toutes les associations entre les différentes entités des utilisateurs DLL sont connues et définies lorsque le système est configuré.

Ainsi, lors de la configuration, le nombre de DLSAP est connu, ainsi que le nombre de connexions dans le DLSAP.

Cette DLL utilise la diffusion comme son mode d'émission sur le support physique. Il y a un seul producteur et un ou plusieurs consommateurs de chaque information émise.

Les connexions dans chaque DLSAP sont des connexions multipoint permettant une communication unilatérale (les connexions ont plus de deux extrémités, et via ces connexions, la communication de données se produit dans un seul sens défini à l'avance: du producteur vers les consommateurs).

Les identificateurs des points d'extrémité de connexion (CEPi) qui sont connus sous le nom d'identificateurs et qui sont codés en utilisant 16 bits identifient les associations entre les entités d'utilisateurs.

Ainsi, les CEPiS ne sont pas reconnus localement dans un DLSAP, mais de façon globale dans l'ensemble du sous-réseau de DL, et le rôle de l'adresse de DLSAP est moins important dans le modèle d'adressage:

Ainsi, le concept d'un DLSAP n'est pas le même que dans le modèle ISO général. Cette DLL représente un groupe compris entre 1 et n identificateurs utilisés par une entité d'utilisateur attachée à la DLL. Par ailleurs, un DLSAP n'attribue pas d'adresse à l'entité d'application directement, mais à travers l'utilisation d'identificateurs.

Chaque connexion (identificateur) peut être utilisée différemment (selon la configuration et les services périodiques ou apériodiques choisis) afin de communiquer avec d'autres stations (transferts de variables) ou avec l'administrateur de bus (demandes pour les transferts apériodiques).

4.3.2.2 Adressage d'échange de messages

4.3.2.2.1 Exigences relatives à l'adressage

L'adressage des utilisateurs de DLS de messagerie doit permettre de satisfaire aux exigences de communication habituelles telles qu'elles sont indiquées en ISO et qui sont comme suit:

— communication point-à-point entre deux utilisateurs de DLS, qu'ils se trouvent ou pas dans la même liaison locale,

— communication multipoint entre une entité et toutes les entités d'un groupe, celui-ci étant localisé dans le même équipement ou la même liaison locale ou la même liaison étendue,

— communication par distribution entre un utilisateur de DLS et tous les autres utilisateurs de DLS d'une DLE, d'une liaison locale ou de la liaison étendue.

NOTE Ceci correspond à la réservation d'un groupe particulier contenant toutes les entités d'un dispositif, d'une liaison locale ou de la liaison étendue. L'adressage des utilisateurs de DLS de messagerie doit distinguer entre deux types d'adresses:

— l'adresse physique, afin d'identifier une entité d'application par rapport à son emplacement physique, par le biais du numéro de segment de DL et le numéro de la station où elle se situe,

— l'adresse logique, afin d'identifier une entité d'application dont l'emplacement physique ne sera pas nécessaire.

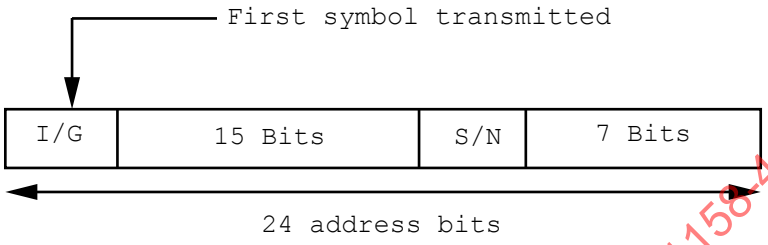
Les notions de groupes restreints via le sous-réseau de DL à l'égard de groupes d'utilisateurs de DLS dans un dispositif ou dans un segment de DL, ont été définies avec un objectif d'optimiser le flux sur le bus. Ainsi, lorsque l'on attribue des adresses à un groupe, il est nécessaire d'avoir la possibilité d'un filtre au niveau de chaque pont. La fonction de ce filtre est d'empêcher la distribution sur tous les segments de DL de la liaison étendue lorsque cela n'est pas nécessaire.

L'adressage des tampons limite le nombre de dispositifs sur un segment de DL à 256. Ceci doit être pris en compte dans le mode d'adressage de messagerie.

4.3.2.2.2 Codage des adresses

Les messages d'échange entre utilisateurs de DLS source et destination sont identifiés dans un sous-réseau de DL par les adresses qui se trouvent dans les unités de données de protocole de liaison de données (RP_MSG_NOACK et RP_MSG_ACK).

Afin de satisfaire aux exigences d'adressage individuel des utilisateurs de DLS dans un dispositif, dans un groupe de dispositifs d'une liaison locale ou dans un groupe de dispositifs de la liaison étendue, l'espace d'adressage offert par les 24 bits est divisé comme le montre la Figure 8.



I/G désigne un utilisateur de DLS individuel (I) ou un groupe d'utilisateurs de DLS (G)

S/N désigne un groupe d'utilisateurs de DLS sur un segment de DL (S) ou sur un sous-réseau de DL (N)

Légende

Anglais	Français
First symbol transmitted	Premier symbole émis
24 address bits	24 bits d'adresse

Figure 8 – Partitionnement d'adresses

Le codage d'adresses individuelles et des groupes d'adresses est montré dans le Tableau 1.

Tableau 1 – Codage d'adresses individuelles et d'adresses de groupe

I/G	S/N	Type d'adressage
0	0	Adresse individuelle
1	0	Adresse d'un groupe dans un segment de DL
non significatif	1	Adresse d'un groupe dans un sous-réseau de DL

La couche de liaison de données offre ainsi un espace d'adressage qui est divisé en deux sous-espaces:

— le premier contenant les adresses de liaison de données destinées à identifier les utilisateurs de DLS individuellement,

— l'autre vise à identifier les groupes d'utilisateurs de DLS.

Les adresses de groupe sont elles-mêmes divisées en deux sous-espaces:

— le sous-espace définissant les adresses de groupes restreints, ce qui permet l'identification des utilisateurs de DLS qui appartiennent à la même liaison locale,

— l'autre espace définissant les adresses de groupes généralisés, permettant ainsi l'identification des identités d'applications qui appartiennent toutes au sous-réseau de DL de manière globale.

Chacun de ces sous-espaces précédemment définis (adresses individuelles et adresses de groupes restreints) est divisé en:

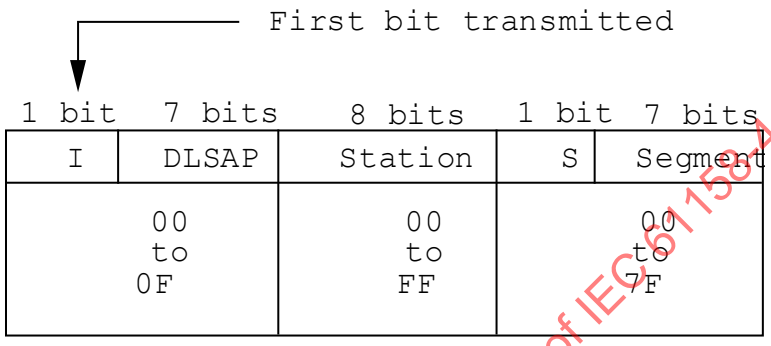
— adresses physiques,

— adresses logiques.

4.3.2.2.2.1 Adressage individuel

Les adresses individuelles permettant à un utilisateur de DLS de correspondre avec un et un seul autre utilisateur de DLS sont structurées comme le montrent la Figure 9 et la Figure 10:

a) Adresses physiques individuelles:



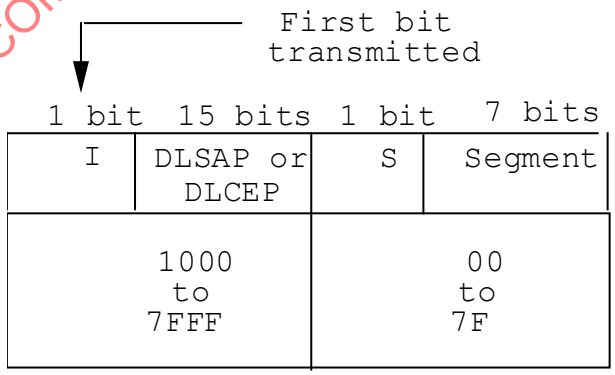
Légende

Anglais	Français
First bit transmitted	Premier bit émis
Station	Station
Segment	Segment
DLSAP	DLSAP

Figure 9 – Structure d'une adresse physique individuelle

Cette structure permet d'identifier 16 DLSAP physiques par dispositif.

b) Adresses logiques individuelles:



Légende

Anglais	Français
First bit transmitted	Premier bit émis
DLSAP or DLCEP	DLSAP ou DLCEP
segment	Segment
1000 to 7FFF	1000 à 7FFF
00 to 7F	00 à 7F

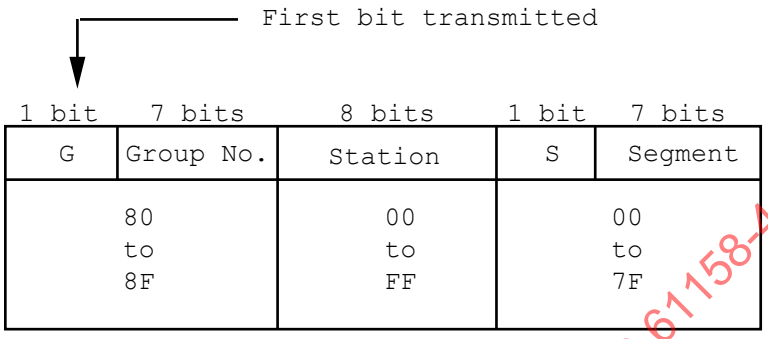
Figure 10 – Structure d'une adresse logique individuelle

La capacité d'adressage est un total de 28K DLSAP et DLCEP logiques par segment de DL.

4.3.2.2.2.2 Adressage de groupes restreints

Les adresses de groupes restreints qui permettent à un utilisateur de DLS de correspondre avec un groupe d'utilisateurs de DLS du même dispositif ou du même segment de DL sont représentées par leurs codes comme montrés à la Figure 11 et à la Figure 12:

a) Adresses de groupes physiques restreints:



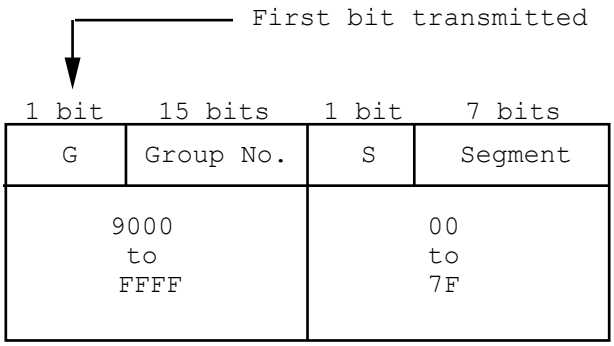
Légende

Anglais	Français
First bit transmitted	Premier bit émis
Group No.	Numéro de groupe
Station	Station
Segment	Segment
80 to 8F	80 à 8F
00 to FF	00 à FF
00 to 7F	00 à 7F

Figure 11 – Structure d'une adresse de groupe physique restreint

La capacité d'adressage comporte ainsi 16 groupes physiques restreints dans une DLE. La distribution dans un dispositif est assurée par le numéro de groupe spécifique, 8F.

b) Adresses de groupes logiques restreints:



Légende

Anglais	Français
First bit transmitted	Premier bit émis
Group No.	Numéro de groupe
segment	Segment

Anglais	Français
9000 to FFFF	9000 à FFFF
00 to 7F	00 à 7F

Figure 12 – Structure d'une adresse de groupe logique restreint

Cette structure permet d'attribuer des adresses à 28K de groupes logiques restreints dans une liaison locale. FFFF correspond au groupe de distribution dans un segment de DL.

4.3.2.2.2.3 Adressage de groupes généralisés

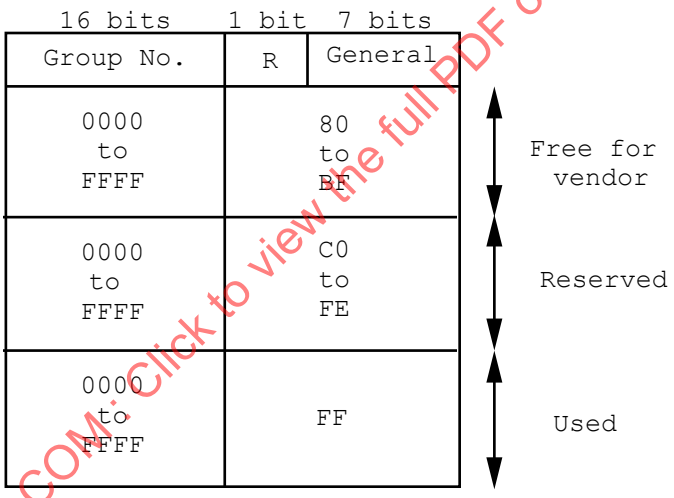
Les adresses de groupes généralisés permettent à un utilisateur de DLS de correspondre avec un groupe d'utilisateurs de DLS sur la liaison étendue. Ces adresses sont divisées en trois sous-zones:

— une sous-zone d'adressage qui peut être utilisée par tous les équipements de la liaison étendue,

— une sous-zone réservée,

— une sous-zone d'adressage du vendeur.

La structure de codage de ces adresses est montrée à la Figure 13:



Légende

Anglais	Français
General	Général
Group No.	Numéro de groupe
0000 to FFFF	0000 à FFFF
80 to BF	80 à BF
C0 to FE	C0 à FE
Free for vendor	Libre au vendeur
reserved	Réservé
Used	Utilisé

Figure 13 – Structure d'une adresse de groupe généralisé

Cette zone d'adressage permet d'identifier 64K groupes généralisés par élément de sous-zone.

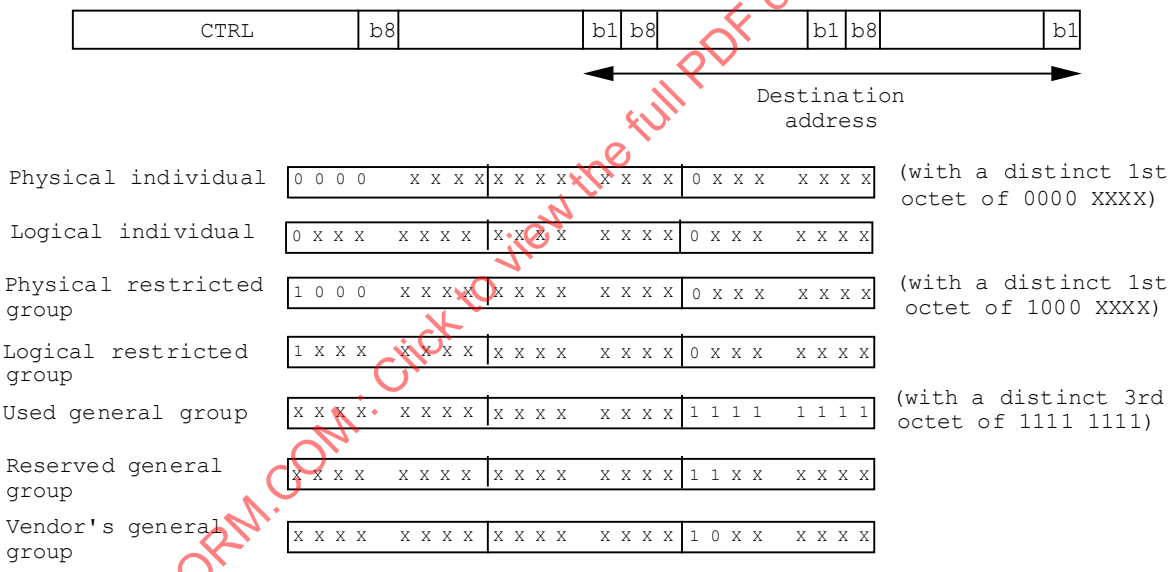
La distribution à toute la liaison étendue est assurée par le numéro de groupe spécifique FFFF.

4.3.2.3 Capacité d'adressage

Les règles de codage offrent les possibilités suivantes pour le sous-réseau de DL en termes de capacité maximale:

— 126 segments de DL dans un sous-réseau DL;
— le numéro de segment "00" est le numéro de segment de DL par défaut;
— 256 DLE par segment de DL dans un sous-réseau DL;
— 16 DLSAP physiques dans une DLE;
— 16 adresses de DL de groupes physiques restreints dans une DLE;
— 28K DLSAP logiques dans une liaison locale;
— 28K adresses de DL de groupes logiques restreints dans une liaison locale;
— 64K adresses de DL de groupes généralisés dans un sous-réseau DL dans la sous-zone en utilisation.

La structure exacte du codage du début d'une DLPDU de réponse de message RP_MSG_XX est comme montrée à la Figure 14:



CTRL désigne le champ de contrôle.

Légende

Anglais	Français
Destination address	Adresse destination
Physical individual	Physique individuelle
Logical individual	Logique individuelle
Physical restricted group	Groupe physique restreint
Logical restricted group	Groupe logique restreint
Used general group	Groupe général utilisé
Reserved general group	Groupe général réservé
Vendor's general group	Groupe général du vendeur
(with a distinct 1st octet of 0000 XXXX)	(avec un premier octet distinct de 0000 XXXX)
(with a distinct 1st octet of 1000 XXXX)	(avec un premier octet distinct de 1000 XXXX)

Anglais	Français
(with a distinct 3rd octet of 1111 1111)	(avec un troisième octet distinct de 1111 1111)

Figure 14 – Résumé de la structure d'adresses

L'adresse source suit l'adresse destination conformément au même codage.

4.4 Contrôle de flux

4.4.1 Échanges de variables

Pour le transfert cyclique de variables identifiées, le principe de balayage périodique implique le remplacement de valeur d'une ancienne variable avec une nouvelle variable indépendante de tout accès en lecture par le consommateur de données.

La gestion de contrôle de flux pour ces échanges est définitivement établie à la configuration:

— l'administrateur de bus gère le contrôle de flux en respectant la période de balayage associée à la variable,

— les stations gèrent le contrôle de flux en associant une mémoire tampon avec chaque variable identifiée.

Ce principe s'applique également aux transferts déclenchés de variables puisque l'utilisateur est seulement intéressé par la valeur la plus récente validée et puisque l'utilisateur associe une mémoire tampon avec chaque variable identifiée. De plus, à la configuration d'un système DLL, un délimiteur de temps est défini pour la fenêtre apériodique pour les transferts déclenchés de variables pour chacun des cycles de balayage de base de l'administrateur de bus.

4.4.2 Transfert de messages

Le contrôle de flux pour les transferts de messages est traité par le mécanisme d'acquittement et par les files d'attente pour les messages à émettre et les messages reçus.

Lorsqu'un système DLL est configuré, un délimiteur de temps est défini pour la fenêtre apériodique pour les transferts déclenchés de messages pour chacun des cycles de balayage de base de l'administrateur de bus.

Pour les transferts de messages avec acquittement, une partie du contrôle de flux est fournie à l'émission puisque la DLL accepte seulement les demandes pour les transferts de messages avec acquittement s'il existe un espace disponible dans la file d'attente pour les messages à émettre et qui est spécifié dans la demande. Avec ce type de transfert, il existe aussi un contrôle de flux à la réception par la file d'attente des messages reçus et par l'émission d'une DLPDU d'acquittement.

4.4.3 Détection de duplication ou de perte de DLPDU

Les mécanismes de détection fournis s'appliquent aux erreurs causées par des problèmes de communication (DLPDU interrompues, erreur FCS) ou par des stations hors service.

Ces erreurs comportent:

— perte d'une DLPDU d'identificateur de DLCEP,

— absence d'une DLPDU de réponse (DLPDU perdue ou station absente),

— perte d'une DLPDU de réponse à une demande,

— perte d'une DLPDU de réponse de message,

— perte d'une DLPDU d'acquittement de message DLPDU,

— perte d'une DLPDU indiquant la fin d'une transaction de message.

et peuvent engendrer soit la perte, soit la duplication de données.

Les pertes de DLPDU sont tenues en compte dans les graphes du diagramme d'états final définissant les protocoles de liaison de données.

La perte ou la duplication des DLPDU peut avoir l'impact suivant sur les services:

— la perte d'une DLPDU d'identificateur de DLCEP ou d'une réponse relative aux variables échangées périodiquement n'est pas généralement néfaste puisque la valeur de la variable sera diffusée encore une fois durant le prochain cycle de la liaison locale;

— la perte d'une DLPDU de message sans acquittement ou d'une DLPDU associée à un échange déclenché de variables sera seulement traitée par les couches supérieures;

— la perte d'une DLPDU d'acquittement engendre une confirmation négative envoyée à l'utilisateur de DLS émetteur;

— le concept de duplication de DLPDU ne s'applique pas aux variables échangées périodiquement puisque le principe de rafraîchissement cyclique de variables implique que la même variable sera envoyée de façon répétitive;

— les DLPDU de messages avec acquittement sont protégées contre la duplication par un mécanisme de numérotation paire/impaire destiné aux messages;

— la duplication d'une DLPDU associée à une demande explicite pour un transfert de mémoire tampon engendrera une substitution supplémentaire de la variable. Cette duplication n'est pas néfaste puisque le concept d'échange de variables cycliquement ou à la suite d'une demande explicite se base sur un rafraîchissement asynchrone qui est traité par la liaison locale et que le consommateur ne peut pas commander. La sémantique de données émises doit être compatible avec ce principe de fonctionnement, qui est, le service d'échange de variables est désigné pour l'émission de statuts, et non pour l'émission de séquences d'événements.

4.5 Représentation graphique

Pour des raisons d'utilité pour le lecteur, certains mécanismes sont expliqués par une représentation graphique. Ces représentations sont basées sur un réseau d'évaluation construit sur un graphe orienté avec trois types de nœuds:

— les états, représentés par un cercle,

— les demandes représentées par un rectangle,

— les transitions, représentées par une ligne horizontale.

Le réseau d'évaluation est construit conformément aux principes suivants, tel qu'illustré dans la Figure 15:

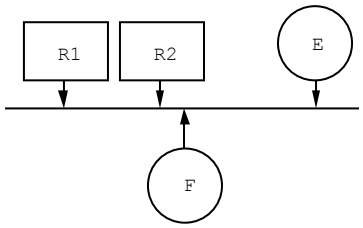


Figure 15 – Exemple d'un réseau d'évaluation

Lorsque le système décrit est dans l'état E, l'arrivée de la demande R1 ou R2 entraîne une transition qui le bascule vers l'état F.

Par ailleurs, le franchissement d'une transition a lieu, par convention, à l'instant zéro.

4.5.1 Actions simples

Des actions simples peuvent être associées à chaque transition, correspondant soit à l'émission des demandes soit à l'exécution des actions locales.

Elles sont schématiquement représentées par un triangle, étiqueté par le nom de la demande ou de la description de l'action exécutée.

4.5.2 Conditions

Le franchissement d'une transition peut concerner une condition. Dans ce cas, le graphe orienté associé à la représentation montre une condition associée à l'arc qui précède la transition.

Un corollaire de cette représentation permet de définir les choix pour le franchissement d'une transition à partir du même état initial.

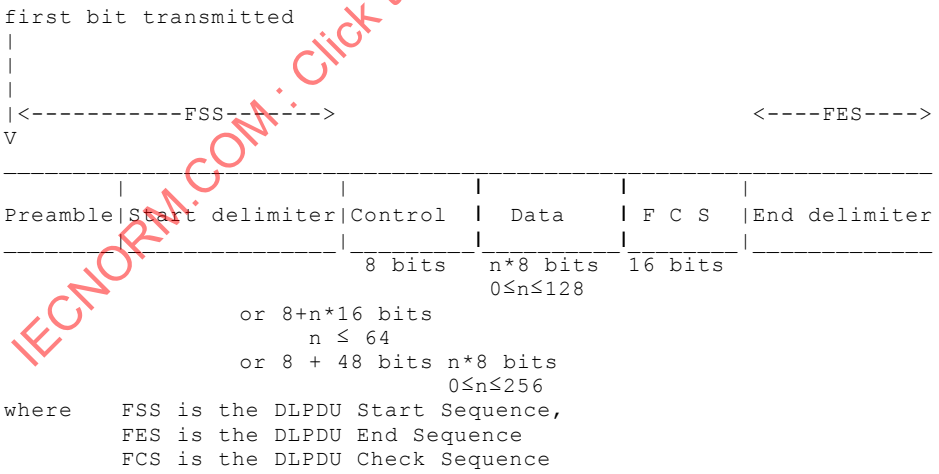
De la même manière que pour une transition simple, l'émission d'une action peut être associée au franchissement d'une transition conditionnelle.

5 Structure générale et codage de PhIDU et de DLPDU et éléments de procédure correspondants

5.1 Formats et composantes de DLPDU

Le présent paragraphe décrit la structure de DLPDU et l'utilisation des divers champs de contrôle dans la DLPDU. Le terme "DLPDU" se réfère aux unités de données de protocole échangées par les entités de liaison de données.

La structure de base d'une DLPDU est telle que montrée dans la Figure 16. L'ordre d'émission et de réception est montré dans la Figure 17:

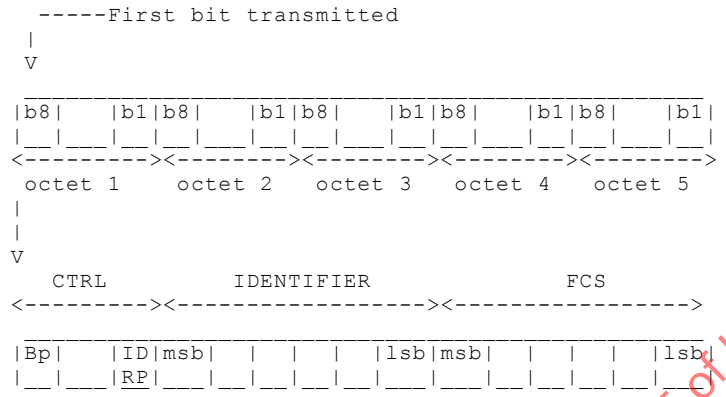


Légende

Anglais	Français
First bit transmitted	Premier bit émis
Preamble	Préambule
Start delimiter	Délimiteur de début
Control	Contrôle
Data	Données
End delimiter	Délimiteur de fin
where	où

Anglais	Français
FSS is the DLPDU Start Sequence,	FSS est la séquence de début de DLPDU
FES is the DLPDU End Sequence	FES est la séquence de fin de DLPDU
FCS is the DLPDU Check Sequence	FCS est la séquence de contrôle de DLPDU
or 8+n*16 bits or 8 + 48 bits n*8 bits	ou 8+n*16 bits ou 8 + 48 bits n*8 bits

Figure 16 – Structure de base d'une DLPDU



Légende

Anglais	Français
First bit transmitted	Premier bit émis
IDENTIFIER	Identificateur
CTRL	CTRL (contrôle)

Figure 17 – Ordre d'émission et de réception d'une DLPDU

Les octets sont émis dans le même ordre dans lequel ils sont reçus (à partir de l'utilisateur de DLS ou de la PhL).

5.2 Description de chaque composante de DLPDU

5.2.1 Préambule

(Voir CEI 61158-2)

5.2.2 Délimiteurs de DLPDU

(Voir CEI 61158-2)

5.2.3 Champ de contrôle et d'adressage

Le champ de contrôle et d'adressage spécifie le type de DLPDU échangée:

— le bit 1 du champ de contrôle indique si la DLPDU est une DLPDU d'identificateur de DLCEP (bit 1 = 1) ou une DLPDU de réponse (bit 1 = 0)

— les bits 2 à 7 du champ de contrôle, où chaque bit a une fonction particulière, comme montré dans le Tableau 2:

  – si bit 2 est mis à 1, la DLPDU est liée à un transfert de mémoire tampon

  – si bit 3 est mis à 1, la DLPDU est liée à un transfert de messages

  - si bit 4 est mis à 1, la DLPDU est liée à une demande explicite pour un transfert de mémoire tampon
  - si bit 5 est mis à 1, la DLPDU est liée à un acquittement
  - le bit 6 indique la priorité ou le résultat de l'acquittement
  - le bit 7 mis à 1 est utilisé seulement pour indiquer une DLPDU de fin de transaction de message;
- le bit 8 du champ de contrôle (le premier bit émis) est utilisé seulement pour une DLPDU de réponse de message ou une DLPDU de réponse d'acquittement. Le bit 8 fournit une numérotation paire/impaire de messages pour prendre en charge la détection de messages dupliqués.

Pour une DLPDU d'identificateur de DLCEP, ces bits spécifient le type d'échange en cours. Pour une DLPDU de réponse, ces bits indiquent le type d'échange en cours ainsi que les demandes pour les transferts de messages et les demandes explicites pour les échanges de variables, avec leurs priorités.

Tableau 2 – Codage du champ de contrôle d'une DLPDU

Bits	DLPDU identificatrice
2 3 4 5 6 7	
1 0 0 0 0 x	allocation pour une variable identifiée
0 1 0 0 0 x	allocation pour message
0 0 1 0 1 x	allocation pour demande explicite urgente
0 0 1 0 0 x	allocation pour demande explicite normale
	DLPDU de réponse
1 0 0 0 0 x	réponse de variable identifiée
1 1 0 0 0 x	réponse de variable identifiée + demande de message
1 0 1 0 1 x	réponse de variable identifiée + demande urgente explicite
1 0 1 0 0 x	réponse de variable identifiée + demande normale explicite
1 1 1 0 1 x	réponse de variable identifiée + demande urgente explicite + demande de message
1 1 1 0 0 x	réponse de variable identifiée + demande normale explicite + demande de message
0 1 0 1 0 x	message avec acquittement
0 1 0 0 0 x	message sans acquittement
0 0 0 1 1 x	acquittement positif
0 0 0 1 0 x	acquittement négatif
0 0 1 0 1 x	liste d'identificateurs pour une demande urgente
0 0 1 0 0 x	liste d'identificateurs pour une demande normale
0 0 0 0 1 x	fin de transaction de message

Pour une DLPDU identificatrice, le champ de contrôle et d'adressage est étendu à 8 + 16 bits. Dans ce cas, il inclut un seul identificateur de DL qui représente une adresse de DL d'une liaison locale, codée sur 16 bits.

Pour les DLPDU de réponse à une demande, le champ de contrôle et d'adressage est étendu à 8 + × fois 16 bits. Dans ce cas, il inclut la liste d'identificateurs de DL associés aux variables demandées.

Pour les DLPDU de réponse de message, le champ de contrôle et d'adressage est étendu à 8 + 24 + 24 bits. Dans ce cas, il inclut deux adresses de DL de liaison étendue — respectivement l'adresse destination et l'adresse source du message.

Le Tableau 3 donne la correspondance entre le nom et le codage des bits du champ de contrôle.

Tableau 3 – Correspondance entre le nom et le codage de 8 bits dans le champ de contrôle

Bits	Nom associé
1 2 3 4 5 6 7 8	
1 1 0 0 0 0 × x	ID_DAT
1 0 1 0 0 0 × x	ID_MSG
1 0 0 1 0 1 × x	ID_RQ1
1 0 0 1 0 0 × x	ID_RQ2
0 1 0 0 0 0 × x	RP_DAT
0 1 1 0 0 0 × x	RP_DAT_MSG
0 1 0 1 0 1 × x	RP_DAT_RQ1
0 1 0 1 0 0 × x	RP_DAT_RQ2
0 1 1 1 0 1 × x	RP_DAT_RQ1_MSG
0 1 1 1 0 0 × x	RP_DAT_RQ2_MSG
0 0 1 0 1 0 × N	RP_MSG_ACK
0 0 1 0 0 0 × x	RP_MSG_NOACK
0 0 0 0 1 1 × N	RP_ACK+
0 0 0 0 1 0 × N	RP_ACK-
0 0 0 1 0 1 × x	RP_RQ1
0 0 0 1 0 0 × x	RP_RQ2
0 0 0 0 0 0 1 x	RP_END
NOTE N prend la valeur 0 ou 1; les valeurs x sont ignorées, mais il convient qu'elles soient 0.	

5.2.4 Champ de données d'utilisateur de DLS

Seulement les DLPDU de réponse incluent un champ de données d'utilisateur de DLS. Le champ de données d'utilisateur de DLS contient:

— soit la valeur d'une variable,

— soit un message.

5.2.5 Séquence de contrôle de trame (FCS, frame check sequence) d'une DLPDU

Dans le présent paragraphe, toute référence à un bit *K* d'un octet est une référence au bit dont le poids dans un entier non signé d'un octet est 2^K.

NOTE Ceci est parfois appelé numérotage de bit "little endian" (petit-boutiste).

Pour le type de protocole de la présente norme, comme dans d'autres normes internationales (par exemple, ISO/CEI 3309, ISO/CEI 8802 et ISO/CEI 9314-2), la détection d'erreur de niveau DLPDU est fournie en calculant et en ajoutant aux autres champs de la DLPDU une séquence de contrôle de trame (FCS), multi-bits pendant l'émission afin de former un "mot de code systématique"[1)] de longueur ***n*** composé de ***k*** bits de message DLPDU suivis par ***n - k*** bits redondants (nombre égal à 16), et en calculant lors de la réception que le message et la

1) W. W. Peterson et E. J. Weldon, Jr., *Error Correcting Codes* (deuxième édition), MIT Press, Cambridge, 1972.

FCS concaténée forment un mot de code légal (*n*,*k*). Le mécanisme pour ce contrôle est comme suit:

La forme générique du polynôme générateur pour la construction de cette FCS est spécifiée dans l'équation (6) et le polynôme pour le résidu prévu du récepteur est spécifié dans l'équation (11). Les polynômes spécifiques pour la présente norme sont spécifiés dans le Tableau 4. Un exemple d'une mise en œuvre est discuté en Annexe A.

Tableau 4 – Longueur, polynôme et résidu prévu de la FCS

Élément	Valeur	
n-k	16	
G(x)	$X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^6 + X^3 + X^2 + X + 1$	(notes 1, 2, 3)
R(x)	$X^{15} + X^{14} + X^{13} + X^9 + X^8 + X^7 + X^4 + X^2$	(note 4)

NOTE 1 Les mots de code D(X) construits à partir du polynôme G(X) ont une distance de Hamming de 4 pour les longueurs ≤ 344 octets et une distance de Hamming de 5 pour les longueurs ≤ 15 octets.

NOTE 2 Ce polynôme G(X)prime, relativement, sur les autres, et il n'est pas compromis par les polynômes fréquemment utilisés dans les équipements de transmission de données (DCE, Data Communication Equipment) (modems): le polynôme de codage différentiel $1 + X^{-1}$ ainsi que tous les polynômes primitifs d'embrouillage de la forme $1 + X^{-j} + X^{-k}$.

NOTE 3 Ce polynôme G(X) est le polynôme à 16-bit optimal pour la détection d'erreur de salve sur des DLPDU de 300 octets ou moins, lorsque les statistiques de la salve d'erreur ont une distribution de Poisson (comme c'est le cas usuel).

NOTE 4 Il convient que le reste R(x) soit 1110 0011 1001 0100 (X^{15} à X^0, respectivement) en absence d'erreurs.

5.2.5.1 À la DLE expéditrice

Le message d'origine (à savoir la DLPDU sans une FCS), la FCS, et le mot de code du message composite (la DLPDU et la FCS concaténés) doivent être considérés comme des vecteurs M(X), F(X) et D(X), de dimensions respectives *k*, *n* - *k* et *n*, dans un champ d'extension sur GF(2). Si les bits du message sont $m_1 \ldots m_k$ et si les bits de la FCS sont $f_{n-k-1} \ldots f_0$, où

$m_1 \ldots m_8$ à partir du premier octet envoyé,

$m_{8N-7} \ldots m_{8N}$ à partir du $N^{ième}$ octet envoyé,

$f_7 \ldots f_0$ à partir du dernier octet envoyé, et

m_1 est envoyé par le(s) premier(s) symbole(s) PhL du message et f_0 est envoyé par le(s) dernier(s) symbole(s) PhL du message (sans compter les informations de mise en trame de PhL),

NOTE Cet ordonnancement “tel qu'émis” est critique pour les propriétés de la FCS relatives à la détection d'erreur.

alors le vecteur message M(X) doit être considéré comme

$$M(X) = m_1X^{k-1} + m_2X^{k-2} + \ldots + m_{k-1}X^1 + m_k \quad (1)$$

et le vecteur FCS F(X) doit être considéré comme

$$F(X) = f_{n-k-1}X^{n-k-1} + \ldots + f_0 = f_{15}X^{15} + \ldots + f_0 \quad (2)$$

Le vecteur composite D(X), pour la DLPDU entière, doit être construit comme étant la concaténation des vecteurs message et FCS

$$D(X) = M(X)\,X^{n-k} + F(X) = m_1X^{n-1} + m_2X^{n-2} + \ldots + m_kX^{n-k} + f_{n-k-1}X^{n-k-1} + \ldots + f_0 \quad (3)$$

$$= m_1X^{n-1} + m_2X^{n-2} + \ldots + m_kX^{16} + f_{15}X^{15} + \ldots + f_0 \quad \text{(pour le cas de k = 15)}$$

La DLPDU présentée à la PhL doit se composer d'une séquence d'un octet dans l'ordre spécifié.

Les bits de contrôle redondant $f_{n-k-1} \ldots f_0$ de la FCS doivent être les coefficients du reste F(X), après division par G(X), de L(X) $(X^k + 1)$ + M(X) X^{n-k}

où G(X) est le polynôme générateur de degré ***n-k*** pour les mots de code

$$G(X) = X^{n-k} + g_{n-k-1}X^{n-k-1} + \ldots + 1 \quad (4)$$

et L(X) est le polynôme de poids maximal (tout à un) de degré ***n-k-1***

$$L(X) = \frac{X^{n-k}+1}{X+1} = X^{n-k-1} + X^{n-k-2} + \ldots + X + 1 \quad (5)$$
$$= X^{15} + X^{14} + X^{13} + X^{12} + \ldots + X^2 + X + 1 \quad \text{(pour le cas de k = 15)}$$

D'où,

$$F(X) = L(X)\,(X^k + 1) + M(X)\,X^{n-k} \ \text{(modulo G(X))} \quad (6)$$

NOTE 1 Les termes L(X) sont inclus dans le calcul afin de détecter la troncature ou l'extension de message initial ou final, en ajoutant à la FCS un facteur dépendant de la longueur.

NOTE 2 En tant qu'une mis en œuvre typique, lorsque n-k = 16, le reste initial de la division est préconfiguré avec tous les bits mis à un. Le flux binaire du message émis est multiplié par X^{n-k} et divisé (modulo 2) par le polynôme générateur G(X), spécifié en équation (7). Le complément de uns du reste résultant est émis en tant que FCS de (n-k) bits, avec le coefficient de X^{n-k-1} émis en premier.

5.2.5.2 À la DLE destinataire

La séquence d'octets indiquée par la PhE doit être concaténée dans la DLPDU et la FCS reçues et considérée comme un vecteur V(X) de dimension ***u***

$$V(X) = v_1X^{u-1} + v_2X^{u-2} + \ldots + v_{u-1}X + v_u \quad (7)$$

NOTE 1 À cause d'erreurs, **u** peut être différent de **n**, la dimension du vecteur de code émis.

Un reste R(X) doit être calculé pour V(X), la DLPDU et la FCS reçues, par une méthode similaire à celle utilisée par la DLE expéditrice (voir 5.2.5.1) dans le calcul de F(X)

$$R(X) = L(X)\,X^u + V(X)\,X^{n-k} \ \text{(modulo G(X))} \quad (8)$$
$$= r_{n-k-1}X^{n-k-1} + \ldots + r_0$$

Définir E(X) comme étant le vecteur de code d'erreur des différences additives (modulo-2) entre le vecteur de code D(X) émis et le vecteur V(X) reçu résultant des erreurs rencontrées (dans le fournisseur de PhS et dans les ponts) entre la DLE émettrice et la DLE destinataire.

$$E(X) = D(X) + V(X) \quad (9)$$

Si aucune erreur ne s'est produite, ainsi E(X) = 0, alors R(X) sera égal à un polynôme de reste constant non nul

$$R_{ok}(X) = L(X)\,X^{n-k} \ \text{(modulo G(X))} \quad (10)$$

dont la valeur est indépendante de D(X). Malheureusement, R(X) sera aussi égal à $R_{ok}(X)$ dans les cas où E(X) est un multiple non nul exact de G(X), auquel cas, il existe des erreurs "non détectables". Dans tous les autres cas, R(X) ne sera pas égal à $R_{ok}(X)$; de telles DLPDU sont erronées et doivent être rejetées sans analyse supplémentaire.

NOTE 2 En tant qu'une mise en œuvre typique, le reste initial de la division est préconfiguré avec tous les bits mis à un. Le flux binaire du message reçu est multiplié par X^{n-k} et divisé (modulo 2) par le polynôme générateur G(X), spécifié en équation (7).

5.3 Structure et codage de PhIDU

Chaque PhIDU est constituée d'informations de contrôle d'interface physique (PhICI) et dans certains cas, un octet de données d'interface physique. Lorsque la DLE émet une DLPDU, elle calcule une séquence de contrôle de DLPDU pour la DLPDU comme spécifié en 5.2.5, concatène la DLPDU et la séquence de contrôle de DLPDU, et émet la paire concaténée en tant qu'une séquence de PhIDU comme suit:

a) La DLE émet une seule primitive Ph-DATA request avec un PhICI spécifiant le début d'activité (start-of-activity), et attend la primitive résultante Ph-DATA confirm.

b) La DLE émet une séquence de primitives Ph-DATA request avec un PhICI spécifiant LES DONNÉES, chaque primitive étant accompagnée par un octet de la DLPDU comme une donnée de l'interface Ph, du premier au dernier octet de la DLPDU et puis, chaque primitive Ph-DATA request attend la primitive résultante Ph-DATA confirm.

c) La DLE émet une séquence de deux primitives Ph-DATA request avec un PhICI spécifiant LES DONNÉES, chaque primitive étant accompagnée par un octet de la FCS comme une donnée de l'interface Ph, du premier au dernier octet de la FCS et puis, chaque primitive Ph-DATA request attend la primitive résultante Ph-DATA confirm.

d) La DLE émet une seule primitive Ph-DATA request avec un PhICI spécifiant LA FIN DE DONNÉES ET D'ACTIVITÉ (end-of-data-and-activity), et attend la primitive résultante Ph-DATA confirm.

La DLE forme une DLPDU reçue en concaténant la séquence d'octets reçus en tant qu'informations de contrôle d'interface physique d'indications Ph-DATA consécutives, en calculant une séquence de contrôle de DLPDU pour ces octets reçus comme spécifié en 5.2.5, et contrôle le syndrome de la FCS calculé pour l'exactitude comme suit:

e) La DLE reçoit une seule primitive Ph-DATA indication avec un PhICI spécifiant LE DÉBUT D'ACTIVITÉ (START-OF-ACTIVITY), et initialise le calcul d'une FCS pour la DLPDU reçue.

f) La DLE reçoit une séquence de primitives Ph-DATA indication avec un PhICI spécifiant LES DONNÉES, chaque primitive étant accompagnée par un octet de la DLPDU reçue comme une donnée de l'interface Ph, calcule d'une façon incrémentale une FCS sur l'octet reçu, et concatène le tout sauf les deux derniers des octets reçus afin de former la DLPDU reçue.

g) La DLE reçoit une seule primitive Ph-DATA indication avec un PhICI spécifiant LA FIN DE DONNÉES (END-OF-DATA), LA FIN DE DONNÉES ET D'ACTIVITÉ (END-OF-DATA-AND-ACTIVITY) ou LA FIN D'ACTIVITÉ (END-OF-ACTIVITY), et vérifie le syndrome de la FCS calculé dans un but d'exactitude:

   1) Si le PhICI spécifiait LA FIN DE DONNÉES (END-OF-DATA) ou LA FIN DE DONNÉES ET D'ACTIVITÉ (END-OF-DATA-AND-ACTIVITY), et le syndrome de la FCS calculé était correct, alors la DLE rapporte la DLPDU reconstruite et les deux octets de la FCS reçue comme une DLPDU bien reçue, appropriée pour des analyses supplémentaires.

   2) Sinon, la DLE incrémente ses statistiques de gestion pour refléter la DLPDU reçue avec erreur.

5.4 Structure commune de DLPDU, codage et éléments de procédure

Chaque DLPDU est constituée d'un champ de contrôle de DLPDU qui spécifie le type de DLPDU et transporte les paramètres de petite taille (octet fractionnaire) de la DLPDU; zéro à deux expriment les champs d'adresse, chacun contenant une adresse de DL, tous de la même longueur; et pour la plupart des DLPDU, un champ de données d'utilisateur transportant toute ou une partie d'une DLSDU. À cela est ajouté avant l'émission, et enlevé après la réception, un champ FCS utilisé pour contrôler l'intégrité de la DLPDU reçue.

5.5 Types de DLPDU valides

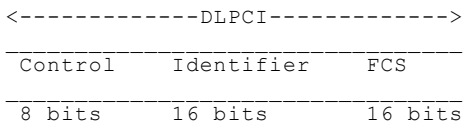
Pour une lecture plus facile, les DLPDU sont décrits sans leurs séquences de début de DLPDU et de fin de DLPDU.

Pour chaque type de DLPDU, un nom est donné pour le champ de contrôle. Ce nom est montré entre parenthèses.

La correspondance entre ce nom et le code de champ de contrôle à 8 bits est donnée dans le tableau ci-dessous.

Pour chaque type de DLPDU, nous avons indiqué la séparation entre les champs qui contiennent les informations de contrôle du protocole de liaison de données (DLPCI) et les champs contenant les données (DLSDU).

5.5.1 DLPDU identificatrice



Légende

Anglais	Français
DLCPI	DLCPI
Identifier	Identificateur
Control	Contrôle
FCS	FCS

Figure 18 – DLPDU identificatrice

Il y a quatre types de DLPDU identificatrice, dont la structure de base est montrée à la Figure 18:

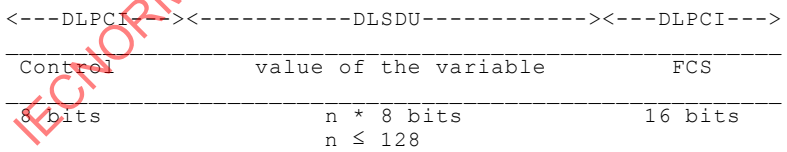
— allocation du support pour variables identifiées (ID_DAT)

— allocation du support pour message (ID_MSG)

— allocation du support pour demande explicite urgente (ID_RQ1)

— allocation du support pour demande explicite normale (ID_RQ2).

5.5.2 DLPDU de réponse de variable



Légende

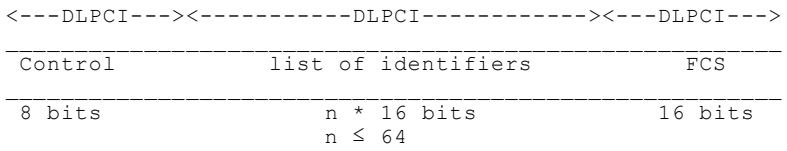
Anglais	Français
DLCPI	DLCPI
DLSDU	DLSDU
Control	Contrôle
value of the variable	Valeur de la variable
FCS	FCS

Figure 19 – DLPDU de réponse variable

Il y a six types de DLPDU de réponse à valeur variable, dont la structure de base est montrée à la Figure 19:

— variables identifiées (RP_DAT)
— variables identifiées + demande de message (RP_DAT_MSG)
— variables identifiées + demande explicite urgente (RP_DAT_RQ1)
— variables identifiées + demande explicite normale (RP_DAT_RQ2)
— variables identifiées + demande explicite urgente + demande de message (RP_DAT_RQ1_MSG)
— variables identifiées + demande explicite normale + demande de message (RP_DAT_RQ2_MSG).

5.5.3 DLPDU de réponse de demande



Légende

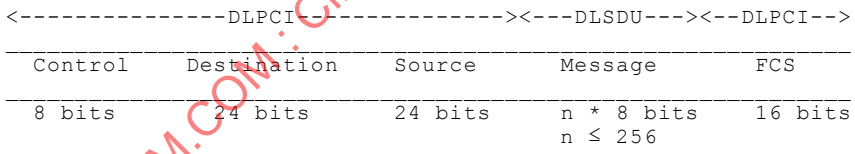
Anglais	Français
DLCPI	DLCPI
Control	Contrôle
List of identifiers	Liste d'identificateurs
FCS	FCS

Figure 20 – DLPDU de réponse de demande

Il y a deux types de DLPDU de réponse à une demande, dont la structure de base est montrée à la Figure 20:

— liste d'identificateurs urgents (RP_RQ1)
— liste d'identificateurs normaux (RP_RQ2).

5.5.4 DLPDU de réponse de message



Légende

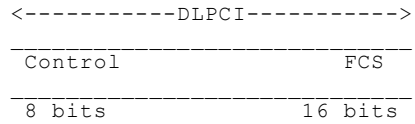
Anglais	Français
DLCPI	DLCPI
DLSDU	DLSDU
Control	Contrôle
Destination	Destination
Source	Source
Message	Message
FCS	FCS

Figure 21 – DLPDU de réponse de message

Il y a deux types de DLPDU de réponse de message, dont la structure de base est montrée à la Figure 21:

— message avec acquittement (RP_MSG_ACK)
— message sans acquittement (RP_MSG_NOACK).

5.5.5 DLPDU de réponse d'acquittement



Légende

Anglais	Français
DLCPI	DLCPI
Control	Contrôle
FCS	FCS

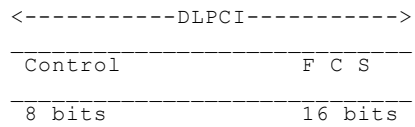
Figure 22 – DLPDU de réponse d'acquittement

Il y a deux types de DLPDU de réponse d'acquittement, dont la structure de base est montrée à la Figure 22:

— un acquittement positif (RP_ACK+) indique que le message a correctement été reçu par la DLE distante,

— un acquittement négatif (RP_ACK-) indique que la file d'attente de l'entité distante pour le message reçu est remplie.

5.5.6 DLPDU de réponse de fin de transaction de message



Légende

Anglais	Français
DLCPI	DLCPI
Control	Contrôle
FCS	FCS

Figure 23 – DLPDU de réponse de fin de transaction de message

Il y a un seul type de DLPDU de réponse de fin de transaction de message, dont la structure de base est montrée à la Figure 23:

NOTE Le chaînage des DLPDU de base est indiqué dans les spécifications pour chaque service.

La DLPDU RP_END implique deux entités: l'entité source et l'administrateur de bus. L'entité source passe le contrôle de la liaison locale à l'administrateur de bus à la fin de transaction de message.

5.6 Temporisateurs de DLL

Le présent paragraphe décrit brièvement les temporisateurs (timers) impliqués dans la description des dispositifs de liaison de données à état final, propose des relations entre les divers temporisateurs et décrit l'importance du temps de changement d'une station dans la définition de ses temporisateurs.

5.6.1 Critères communs

Le critère commun à toutes les stations pour le lancement des temporisateurs est l'absence d'activité sur la liaison locale.

Deux signaux sont utilisés pour évaluer ce critère: "Absence d'activité" (SILENCE) et "Signal présent" (BUSY).

Un temporisateur de référence T0 est défini comme correspondant au temps de silence maximal permis sur une liaison locale.

Le temporisateur T0 est un paramètre global de systèmes.

T0 est activée à la réception d'une indication d'absence d'activité. Cette indication est émise par la couche physique (PhL) Type 2 (la primitive PhL-DATA indication (SILENCE)).

T0 est désactivée par un fonctionnement correct de la liaison locale, lors de la réception de l'indication de présence de signal. Cette indication est émise par la couche physique (PhL) Type 2 (la primitive PhL-DATA indication (BUSY)).

Le temps maximal de silence autorisé sur la liaison locale, qui est, T0 est lié au temps de changement de stations dans le système.

5.6.2 Temps de changement

La réception ou l'émission du dernier symbole MAC est signalée par une indication SILENCE de la PhL. De même, la réception ou l'émission du premier symbole MAC est signalée par une indication BUSY de la PhL.

Le temps écoulé entre ces deux signaux dans une seule station définit le temps de changement de la station (Station Turnaround Time (STT)), que la fonction de la station soit celle d'une entité productrice/consommatrice ou d'un administrateur de bus.

Le temps de changement prend en compte des paramètres qui sont locaux à la station et des paramètres qui sont associés au système auquel la station appartient. Ces paramètres sont:

— temps de réaction physique (Physical Reaction Time (PRT)): le temps nécessaire à la PhL pour détecter la fin ou le début d'activité. Ces temps dépendent des paramètres de liaisons locales physiques:

  station: le temps nécessaire pour détecter l'absence ou la présence d'un signal

  support: le temps nécessaire pour traverser le support et les étoiles en cuivre ou optiques actives

— temps de latence (TI) résultant du temps de traitement dans la station.

Le temps de latence (TI) prend différentes valeurs en fonction de la situation de protocole dans laquelle la station se trouve:

a) la station a émis une DLPDU et émettra la DLPDU suivante,

b) la station a reçu une DLPDU et émettra la DLPDU suivante,

c) la station a émis une DLPDU et recevra la DLPDU suivante,

d) la station a reçu une DLPDU et recevra la DLPDU suivante.

Tous les paramètres se combinent pour former un temps de changement minimal de station (dans les conditions les plus favorables) et un temps de changement maximal de station (dans les conditions les moins favorables).

Pour que le système fonctionne correctement, les critères suivants *doivent être* respectés:

Règle 1) une station qui reçoit une DLPDU doit toujours avoir un temps de changement plus court que celui de la station qui émet la DLPDU, indépendamment des deux stations impliquées (station destinataire, station émettrice).

Cette condition est exprimée par:

Max(i) STT de la station destinataire i ≤ Min(j) STT de la station émettrice j

Règle 2) le temps de changement minimal correspond à la valeur maximale du temps de réaction physique PRT.

Règle 3) Étant donné l'impact du temps de changement d'une station sur le rendement de l'émission (utilisation de la bande passante), il est important que le STT maximal prenne les critères de rendement en considération.

Les règles 1 et 2 sont des critères d'interfonctionnement. La règle 3 est un critère de performance qui peut être vérifié au niveau de la gestion de système à travers l'accès à la valeur maximale du temps de changement et le temporisateur T0 résultant.

NOTE Le choix des valeurs STT minimale et maximale peut engendrer différentes classes d'interopérabilité et de performance.

5.6.3 Fonctions de temporisateurs

Cinq temporisateurs sont définis en utilisant T0 comme une base.

Les temporisateurs T1, T4 et T6 correspondent à l'utilisation de T0 dans le contexte spécifique du statut d'une DLE (Station ou administrateur de bus). Les conditions d'activation et de désactivation sont ainsi identiques (SILENCE ou BUSY), et l'action après l'expiration des temporisateurs diffère selon le statut de la DLE.

Les temporisateurs T3 et T5 se réfèrent aussi à T0: leurs longueurs (qui sont différentes de T0) sont calculées en utilisant la longueur T0 comme une base. De plus, ils sont activés et désactivés dans les mêmes conditions (SILENCE ou BUSY).

Un temporisateur T2 complémentaire est aussi utilisé. T2 n'est pas basé sur T0. Ce temporisateur détermine la longueur d'un cycle de base synchronisé.

Tous ces temporisateurs sont énumérés dans le Tableau 5.

Tableau 5 – Temporisateurs de DL

Nom du temporisateur	Emplacement	Fonction
T1	B.A.	surveille l'absence de RPxx après IDxx
T2	B.A.	surveille le remplissage de la fenêtre de synchronisation par des identificateurs à partir de la séquence de remplissage de base
T3	B.A.	surveille l'absence de IDxx après commutation RPxx des administrateurs de bus
T4	Station consommatrice	surveille l'absence de RP_DATxx après ID_DAT
T5	B.A.	surveille l'absence de RP_END après ID_MSG
T6	Station source de message	surveille l'absence de RP_ACK après RP_MSG_ACK

Les définitions de temporisateur se réfèrent aux noms de statuts mentionnés dans les descriptions de protocole présentées à l'Article 7.

5.6.3.1 Temporisateur T1

— Se situe dans l'administrateur de bus actif.

— Son but est de surveiller l'émission d'une DLPDU de réponse de variable ou d'une DLPDU de réponse à une demande dans un temps donné.

— Est actif lorsque l'administrateur de bus est en attente d'une DLPDU de réponse de variable ou d'une DLPDU de réponse à une demande (statut BA_WAITING_DAT ou BA_WAITING_RQ), après son émission d'une DLPDU d'identificateur de variable ou d'identificateur de demande.

— La longueur du temporisateur T1 est égale à la longueur du temporisateur T0.

— L'expiration du temporisateur donne à l'administrateur de bus un nouveau statut: celui d'émettre la DLPDU identificatrice suivante (statut NEXT).

5.6.3.2 Temporisateur T2

— Se situe dans l'administrateur de bus actif.

— Son but est de surveiller la longueur de la fenêtre de synchronisation du cycle de base. L'administrateur de bus actif émet les DLPDU identificatrices de la séquence de remplissage de base tant que le temporisateur T2 n'a pas expiré.

— Est activé au début de chaque cycle de base synchronisé, lorsque l'administrateur de bus prend le statut des DLPDU identificatrices émettrices (statut NEXT après le statut BA_WAITING_SYNC).

— N'est pas désactivé par n'importe quel événement.

— La longueur de la fenêtre de synchronisation est une fonction des échanges apériodiques du cycle de base ou des services de message.

— L'expiration du temporisateur donne à l'administrateur de bus un nouveau statut: celui d'émettre la DLPDU identificatrice suivante (statut NEXT).

5.6.3.3 Temporisateur T3

— Se situe dans l'administrateur de bus potentiel.

— Son but est de surveiller l'émission par l'administrateur de bus actif des DLPDU identificatrices dans un temps donné.

— Est actif tant que l'administrateur de bus a potentiellement un statut actif.

— Il convient que la longueur du temporisateur T3 soit supérieure à la longueur du temporisateur T0. Il convient qu'elle ait une valeur différente pour chacun des administrateurs de bus potentiels. La note ci-dessous propose une valeur pour cette longueur.

— L'expiration du temporisateur déclenche l'activation d'un administrateur de bus.

NOTE Les dimensions du temporisateur T3 sont définies comme suit:

T3 = $k \times n \times$ T0 avec
n>2 une fonction de l'adresse géographique de l'administrateur de bus auquel T3 est attachée et
k un coefficient qui permet au temps de changement de l'administrateur de bus d'être couvert.

5.6.3.4 Temporisateur T4

— Se situe dans les stations.

— Son but est de surveiller l'émission d'une DLPDU de réponse de variable sur le bus dans un temps donné, par exemple afin d'associer correctement une DLPDU d'identificateur de DLCEP reçue avec la DLPDU de réponse correspondante.

— Est actif lorsque la station est en attente d'une DLPDU de réponse de variable (statut WAITING_DAT).

— La longueur de T4 est égale à la longueur de T0.

— L'expiration du temporisateur (timer) fait que la station est placée dans le statut d'attente pour la DLPDU identificatrice suivante (statut WAITING_ID).

5.6.3.5 Temporisateur T5

— Se situe dans l'administrateur de bus actif.

— Son but est de surveiller l'émission par une station source de messages d'une DLPDU de réponse indiquant la fin d'une transaction de message dans un temps donné.

— Est actif lorsque l'administrateur de bus est en attente d'une DLPDU de réponse de fin de message (statut BA_WAITING_END), après son émission d'une DLPDU d'identificateur de message.

— Il convient que la longueur du temporisateur T5 soit plus grande que la longueur du temporisateur T0. La note ci-dessous propose une valeur pour T5.

— L'expiration du temporisateur donne à l'administrateur de bus un nouveau statut: celui d'émettre la DLPDU identificatrice suivante (statut NEXT).

NOTE La longueur du temporisateur T5 est égale au double de la longueur du temporisateur T0.

Avec un tel choix de longueurs, une collision entre deux opérateurs est évitée. Par exemple, lorsqu'une DLPDU d'acquittement RP_ACK est perdue, l'administrateur de bus ne peut pas émettre un deuxième ID_MSG au même moment où la station source répond avec l'émission de RP_MSG_ACK (voir T6).

5.6.3.6 Temporisateur T6

— Se situe dans les stations.

— Son but est de surveiller l'émission d'une DLPDU d'acquittement par la station destination dans un temps donné.

— Est actif lorsque la station source est en attente d'une DLPDU d'acquittement (statut WAITING_ACK) après la fin de son émission d'une DLPDU de réponse de message avec acquittement.

— La longueur de T6 est égale à la longueur de T0.

— L'expiration du temporisateur (timer) fait que la station est placée dans le statut de réémission du message si possible (statut RESTART_SOURCE) ou sinon dans le statut d'attente pour la DLPDU identificatrice suivante (statut WAITING_ID) après l'émission de la DLPDU de fin de transaction de message.

La définition de ces temporisateurs est nécessaire pour garantir la capacité d'interopérabilité des différents dispositifs connectés à une liaison DLL locale.

Il convient que tous les temporisateurs dans une seule liaison locale aient la même valeur. Une valeur par défaut est définie. Ces valeurs peuvent être modifiées par la gestion de système d'une station tant que l'équivalence des valeurs est respectée. Les valeurs seront convenues ultérieurement conformément aux besoins de mise en œuvre.

6 Structure spécifique de DLPDU, codage et élément de procédure

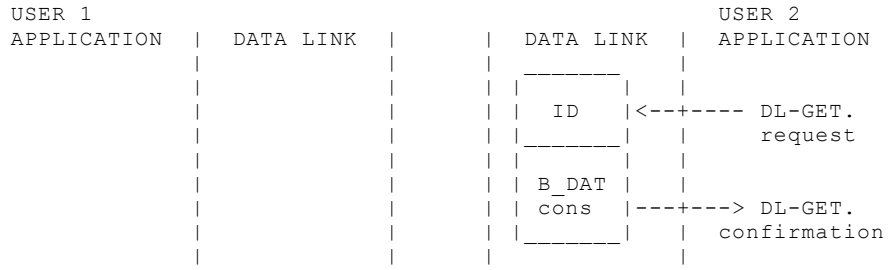
6.1 Généralités

Le présent article décrit la structure, le contenu de PDU et spécifie les éléments de procédure liés au service invoqué.

Le comportement des cas particuliers est décrit en détail dans le paragraphe correspondant en utilisant des diagrammes d'états dédiés.

6.2 Lecture de la mémoire tampon

Le service de lecture de la mémoire tampon est local à une entité, car les interactions liées à ce service ont lieu seulement à travers l'interface utilisateur de DLS/DLL, comme montré à la Figure 24. La correspondance entre les unités de données de services DLSDU et les interactions entre la DLL et le support de communication sont fournies par le service de transfert de mémoire tampon.



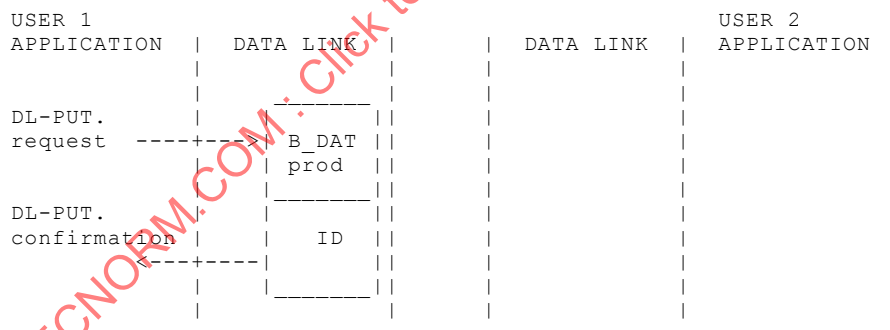
Légende

Anglais	Français
USER 1 APPLICATION	APPLICATION D'UTILISATEUR 1
USER 2 APPLICATION	APPLICATION D'UTILISATEUR 2
DATA LINK	LIAISON DE DONNÉES

Figure 24 – Interactions de service de lecture de mémoire tampon

6.3 Écriture dans la mémoire tampon

Le service d'écriture dans la mémoire tampon est local à une entité étant donné que les interactions liées à ce service ont lieu seulement à travers l'interface utilisateur de DLS/DLL, comme montré à la Figure 25. La correspondance entre les unités de données de services DLSDU et les interactions entre la DLL et le support de communication sont fournies par le service de transfert de mémoire tampon.



Légende

Anglais	Français
USER 1 APPLICATION	APPLICATION D'UTILISATEUR 1
USER 2 APPLICATION	APPLICATION D'UTILISATEUR 2
DATA LINK	LIAISON DE DONNÉES

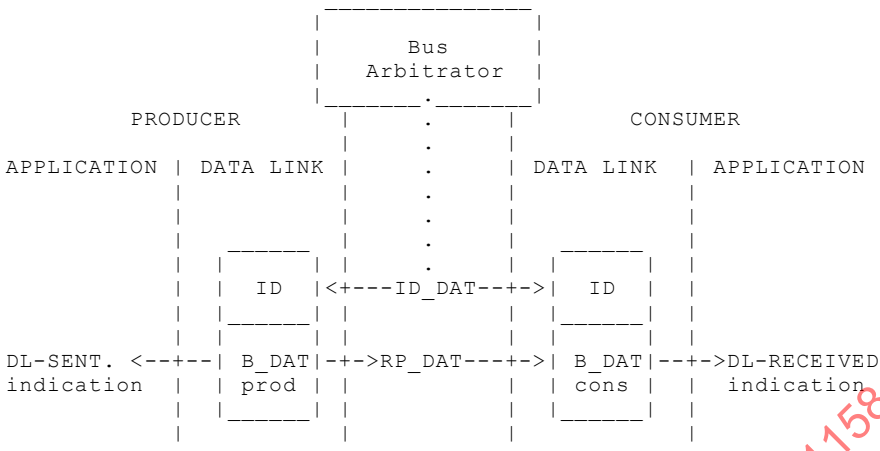
Figure 25 – Interactions de service d'écriture dans la mémoire tampon

6.4 Transfert de mémoire tampon

Il y a deux phases pour le transfert de la mémoire tampon pour toutes les variables, comme montré à la Figure 26:

— l'administrateur de bus émet la DLPDU identificatrice associée à la variable,

— l'entité productrice émet la DLPDU de réponse correspondante et génère une primitive DL-SENT indication pour l'utilisateur de DLS. La valeur de la variable est captée par toutes les entités consommatrices et une primitive DL-RECEIVED indication est générée et transférée à chacun des utilisateurs de DLS concernés.



Légende

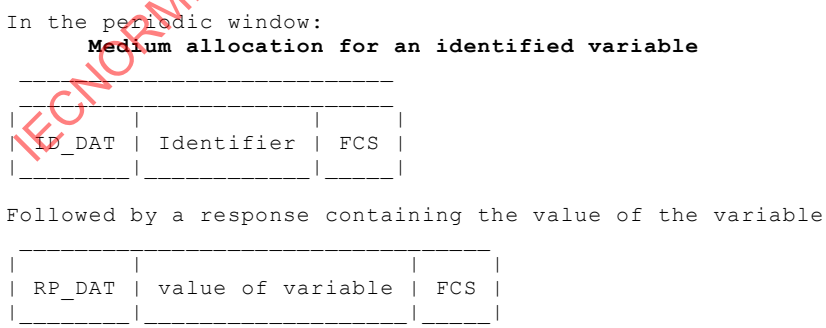
Anglais	Français
Bus Arbitrator	Administrateur de Bus
PRODUCER	Producteur
CONSUMER	Consommateur
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 26 – Interactions de service de transfert de mémoire tampon

6.4.1 DLPDU associées

Un transfert de mémoire tampon implique la circulation de deux DLPDU.

L'administrateur de bus diffuse une DLPDU d'identificateur de DLCEP (ID_DAT) qui alloue le support à l'entité qui produit la variable associée. L'entité productrice diffuse alors la variable dans une DLPDU de réponse (RP_DAT) qui est reçue par l'/les entité(s) consommatrice(s). Les phases de cette séquence sont montrées à la Figure 27.



Légende

Anglais	Français
In the periodic window:	Dans la fenêtre périodique
Medium allocation for an identified variable	Allocation du support pour une variable identifiée
Identifier	Identificateur
Followed by a response containing the value of the variable	Suivie par une réponse contenant la valeur de la variable

Anglais	Français
value of the variable	Valeur de la variable

Figure 27 – Séquence de DLPDU de transfert de mémoire tampon

NOTE Une description de toutes les DLPDU et l'utilisation de divers champs, peuvent être trouvées en 4.5.

6.5 Demande explicite spécifiée

Une demande explicite spécifiée pour un échange de variables peut être émise par n'importe quelle entité. La demande déclenche les échanges dont la forme dépend de la valeur de l'indicateur RQ_INHIBIT associé à SPECIFIED_IDENTIFIER. Cette valeur est réglée à la suite de la configuration.

Si RQ_INHIBIT = FALSE, il y a trois phases pour l'échange:

phase 1:

Suite à une primitive «request» de DL-SPEC-UPDATE, l'entité émettrice utilise le champ de contrôle de la DLPDU de réponse de variable (correspondant à l'identificateur spécifié lorsque le service a été demandé) pour demander l'administrateur de bus de faire circuler un identificateur de demande urgente.

phase 2:

L'administrateur de bus émet l'identificateur de demande correspondant dans la fenêtre allouée aux échanges déclenchés de variables (fenêtre apériodique).

phase 3:

L'entité émettrice émet une DLPDU de réponse à une demande dont le champ de "données" contient la liste d'identificateurs que l'entité souhaite diffuser. L'entité émettrice émet aussi à l'utilisateur de DLS une primitive «confirm» de DL-SPEC-UPDATE. Un ou plusieurs transferts de mémoire tampon ont lieu alors dans le cycle de trafic apériodique.

— l'administrateur de bus émet les identificateurs des variables demandées,

— le producteur de chaque variable demandée émet une DLPDU de réponse de variable.

Si RQ_INHIBIT = TRUE, il y a deux phases pour l'échange:

phase 1:

Indépendamment des demandes, pour tous les identificateurs mis de côté pour un service spécifié dont l'indicateur RQ_INHIBIT est "true", l'administrateur de bus émet un identificateur de demande dans la fenêtre périodique.

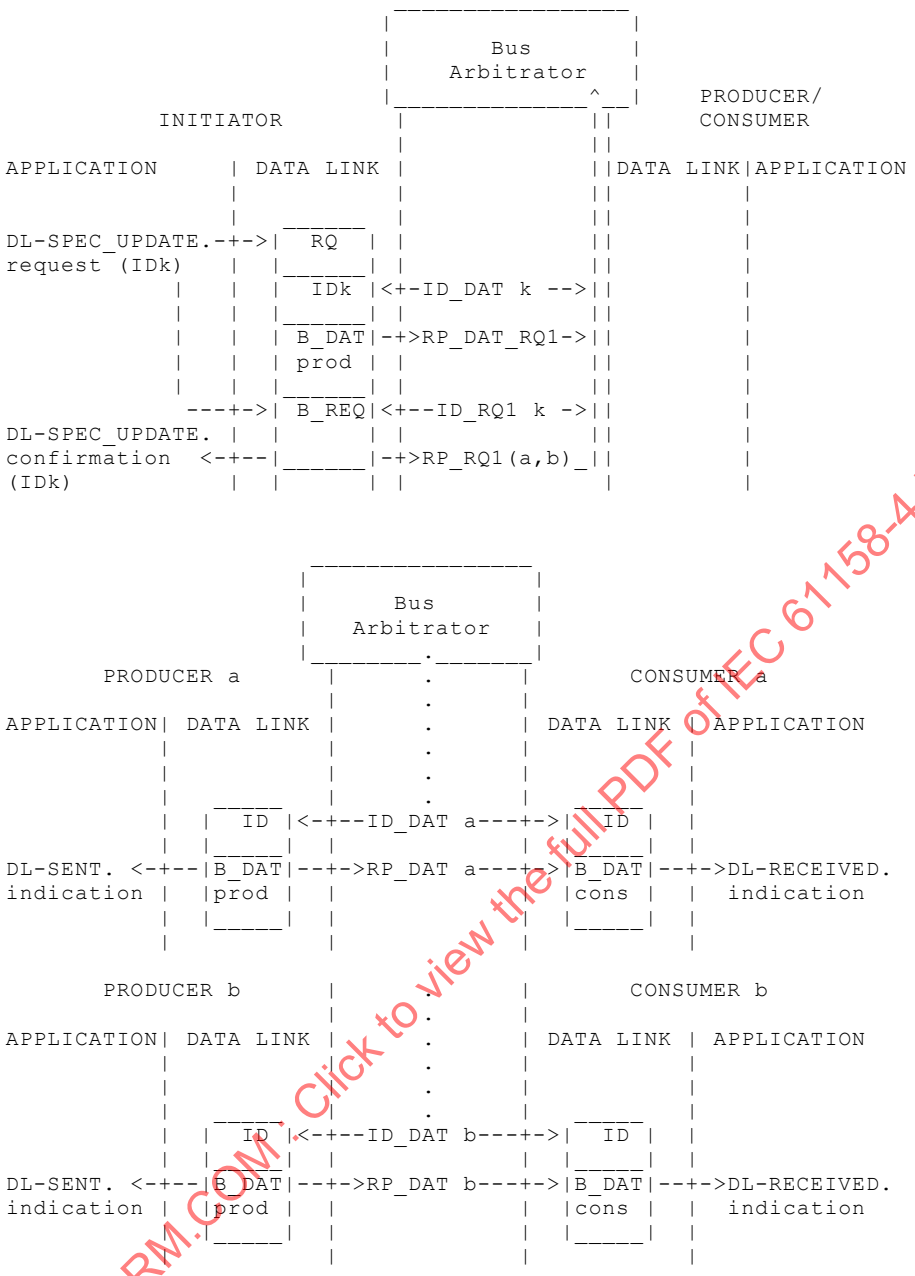
phase 2:

L'entité émettrice émet une DLPDU de réponse à une demande dont le champ de "données" contient la liste d'identificateurs de variables que l'entité souhaite diffuser. L'entité émettrice émet aussi à l'utilisateur de DLS une primitive de confirmation de DL-SPEC-UPDATE. Un ou plusieurs transferts de mémoire tampon ont lieu alors dans le cycle de trafic périodique immédiatement à la suite de la demande.

— l'administrateur de bus émet les identificateurs des variables demandées,

— le producteur de chaque variable demandée émet une DLPDU de réponse de variable.

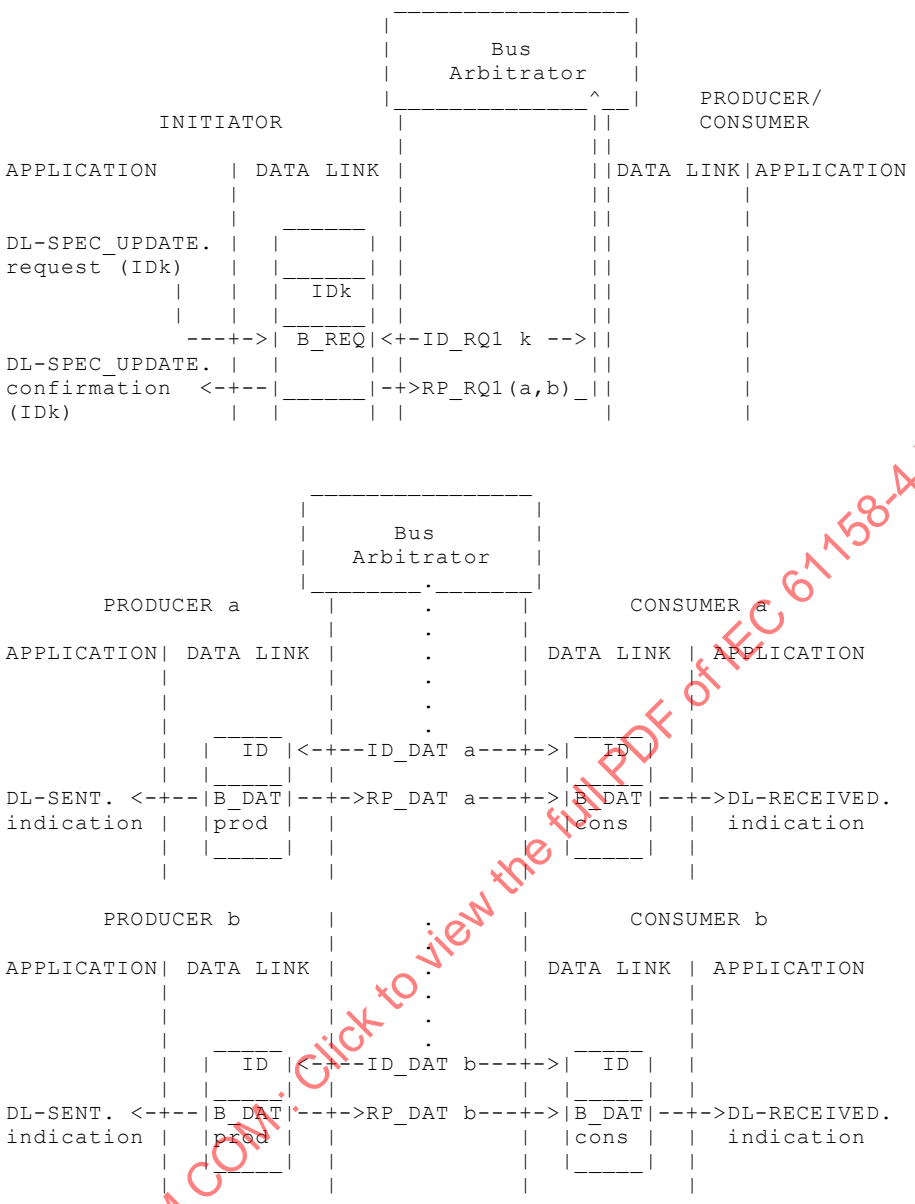
Les séquences d'interaction pour les deux classes d'opération non erronée sont montrées dans la Figure 28 et la Figure 29.



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
PRODUCER/ CONSUMER	PRODUCTEUR/CONSOMMATEUR
INITIATOR	INITIATEUR
CONSUMER a	CONSOMMATEUR a
PRODUCER a	PRODUCTEUR a
CONSUMER b	CONSOMMATEUR b
PRODUCER b	PRODUCTEUR b
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 28 – Interactions dans une demande explicite spécifiée pour un service de transfert de mémoire tampon dans la fenêtre apériodique



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
PRODUCER/ CONSUMER	PRODUCTEUR/CONSOMMATEUR
INITIATOR	INITIATEUR
CONSUMER a	CONSOMMATEUR a
PRODUCER a	PRODUCTEUR a
CONSUMER b	CONSOMMATEUR b
PRODUCER b	PRODUCTEUR b
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 29 – Interactions dans une demande explicite spécifiée pour un service de transfert de mémoire tampon dans la fenêtre périodique

6.5.1 DLPDU associées

Une demande explicite pour un transfert de mémoire tampon correspond à la circulation de trois DLPDU. La première de ces DLPDU fait partie d'un balayage périodique normal.

L'entité émettrice diffuse une DLPDU de réponse d'échanges de variables dont le champ de contrôle inclut la demande à l'administrateur de bus de faire circuler un identificateur de demande. L'entité source énonce le niveau de priorité (RP_DAT_RQi, ou RP_DAT_RQi_MSG si un service de message apériodique est aussi demandé avec cet identificateur).

L'administrateur de bus diffuse alors une DLPDU d'identificateur de demande (ID_RQi) pendant une fenêtre apériodique. L'administrateur de bus utilise l'identificateur qui a porté la demande.

La liste d'identificateurs variables à diffuser est envoyée à l'administrateur de bus (RP_RQi).

L'administrateur de bus répond par un ou plusieurs transferts de mémoire tampon: transaction (ID_DAT, RP_DAT).

Si la demande explicite spécifiée fait partie du cycle de balayage périodique de l'administrateur de bus, la circulation de la première DLPDU est nulle et les autres DLPDU sont échangées pendant le balayage apériodique.

La Figure 30 montre la séquence de DLPDU pour une demande explicite pour un transfert:

Légende

Anglais	Français
In the periodic window:	Dans la fenêtre périodique
Medium allocation for an identified variable	Allocation du support pour une variable identifiée
Identifier	Identificateur
list of identifiers	Liste d'identificateurs
followed by a response with an explicit request for a transfer, and the transfer's priority:	suivie d'une réponse avec une demande explicite pour un transfert, et la priorité du transfert:
value of the variable	Valeur de la variable
or	Ou
and then a number of identifiers later, in the aperiodic window	et puis un nombre d'identificateurs plus tard, dans la fenêtre apériodique
Medium allocation for an identified variable and a response list of variable identifiers	Allocation du support pour une variable identifiée et une liste de réponse d'identificateurs de variables
followed by a list of the identifiers required	suivie d'une liste d'identificateurs requis
followed by the transactions at a later time during the aperiodic window	suivie des transactions ultérieurement durant la fenêtre apériodique
medium allocation for identified variables (ID_DAT)	Allocation du support pour des variables identifiées (ID_DAT)
associated responses (RP_DAT).	Réponses associées (RP_DAT).

Figure 30 – Séquence de DLPDU pour une demande explicite pour un transfert

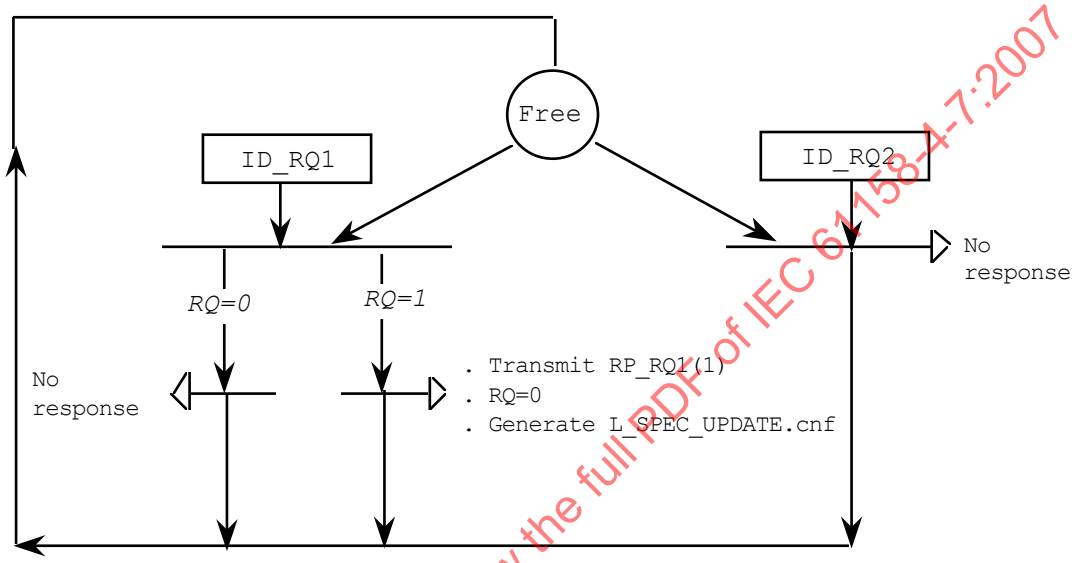
6.5.2 Comportement avec des paramètres discordants

6.5.2.1 Demande explicite spécifiée de transfert de mémoire tampon (RQ_INHIBITED = False)

Le réseau d'évaluation de la Figure 31 permet de spécifier le comportement d'une entité productrice/consommatrice dans les cas suivants:

— réception d'une DLPDU ID_RQ2 contenant un identificateur de DLCEP portant une demande explicite spécifiée d'échange de transfert de mémoire tampon (RQ=1),

— réception d'une DLPDU ID_RQ1 contenant un identificateur de DLCEP ne portant pas une demande explicite spécifiée d'échange de transfert de mémoire tampon (RQ=0).



NOTE 1 Lorsque RQ=1 et B_REQ contiennent une liste vide, RP_RQ1 est émis avec une liste vide.

Légende

Anglais	Français
Free	Libre
No response	Pas de réponse
Transmit RP_RQ1(1)	Émettre RP_RQ1(1)
Generate L_SPEC_UPDATE.cnf	Générer L_SPEC_UPDATE.cnf

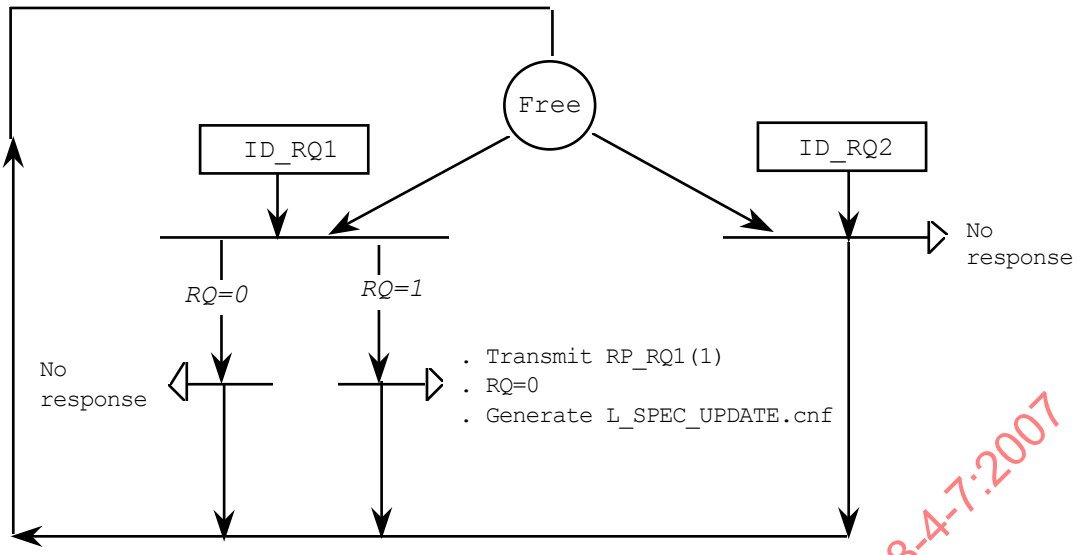
Figure 31 – Réseau d'évaluation pour une demande explicite spécifiée de transfert de mémoire tampon avec (RQ_INHIBITED = False)

6.5.2.2 Demande explicite spécifiée de transfert de mémoire tampon (RQ_INHIBITED = True)

Le réseau d'évaluation de la Figure 32 permet de spécifier le comportement d'une entité productrice/consommatrice dans les cas suivants:

— réception d'une DLPDU ID_RQ2 contenant un identificateur de DLCEP configuré pour le service de demande explicite spécifiée de transfert de mémoire tampon comme RQ_INHIBITED=TRUE,

— réception d'une DLPDU ID_RQ1 contenant un identificateur de DLCEP configuré pour le service de demande explicite spécifiée de transfert de mémoire tampon comme RQ_INHIBITED=TRUE et dont le contenu a déjà été émis sur le bus.



NOTE 1 Lorsque RQ=1 et B_REQ contiennent une liste vide, RP_RQ1 est émis avec une liste vide.

Légende

Anglais	Français
Free	Libre
No response	Pas de réponse
Transmit RP_RQ1(1)	Émettre RP_RQ1(1)
Generate L_SPEC_UPDATE.cnf	Générer L_SPEC_UPDATE.cnf

Figure 32 – Réseau d'évaluation pour une demande explicite spécifiée de transfert£ de mémoire tampon avec (RQ_INHIBITED = True)

6.6 Demande explicite libre

Une demande explicite libre pour l'échange d'une variable identifiée peut être émise par n'importe quelle entité et a lieu en trois phases.

phase 1:

Suite à une primitive «request» DL-FREE-UPDATE, l'entité émettrice utilise le champ de contrôle de la DLPDU de réponse de variable pour demander l'administrateur de bus de faire circuler un identificateur de demande. L'entité indique la priorité.

phase 2:

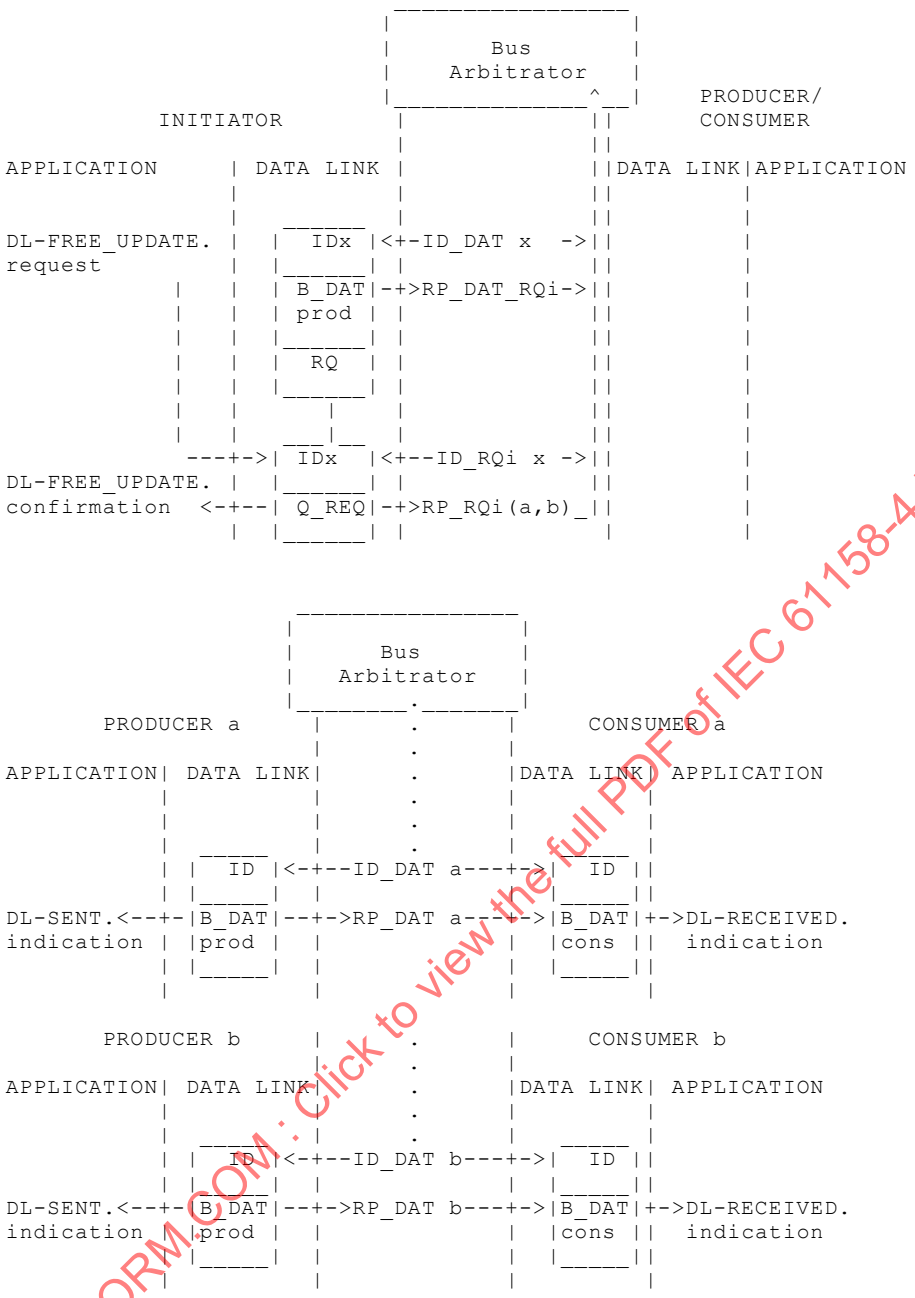
L'administrateur de bus émet l'identificateur de demande correspondant dans la fenêtre allouée aux échanges déclenchés de variables (fenêtre apériodique).

phase 3:

L'entité émettrice émet une DLPDU de réponse à une demande dont le champ de "données" contient la liste d'identificateurs de variables que l'entité souhaite diffuser. L'entité émettrice émet aussi à l'utilisateur de DLS une primitive de confirmation DL-FREE-UPDATE.

Un ou plusieurs transferts de mémoire tampon ont lieu alors dans le cycle de trafic apériodique.

Les séquences d'interaction pour une opération non erronée sont montrées à la Figure 33.



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
PRODUCER/ CONSUMER	PRODUCTEUR/CONSOMMATEUR
INITIATOR	INITIATEUR
CONSUMER a	CONSOMMATEUR a
PRODUCER a	PRODUCTEUR a
CONSUMER b	CONSOMMATEUR b
PRODUCER b	PRODUCTEUR b
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 33 – Diagramme d'interactions dans une demande explicite libre pour un service de transfert de mémoire tampon

6.6.1 DLPDUs associées

Une demande explicite libre pour un transfert de mémoire tampon correspond à la circulation de trois DLPDU, comme montré en 6.5.1. La première de ces DLPDU fait partie d'un balayage périodique normal.

L'entité émettrice diffuse une DLPDU de réponse d'échanges de variables dont le champ de contrôle inclut la demande à l'administrateur de bus de faire circuler un identificateur de demande. L'entité source énonce le niveau de priorité (RP_DAT_RQi, ou RP_DAT_RQi_MSG si un service de message apériodique est aussi demandé avec cet identificateur).

L'administrateur de bus diffuse alors une DLPDU d'identificateur de demande (ID_RQi) pendant une fenêtre apériodique. L'administrateur de bus utilise l'identificateur qui a porté la demande.

La liste d'identificateurs variables à diffuser est envoyée à l'administrateur de bus (RP_RQi).

L'administrateur de bus répond par un ou plusieurs transferts de mémoire tampon: transaction (ID_DAT, RP_DAT).

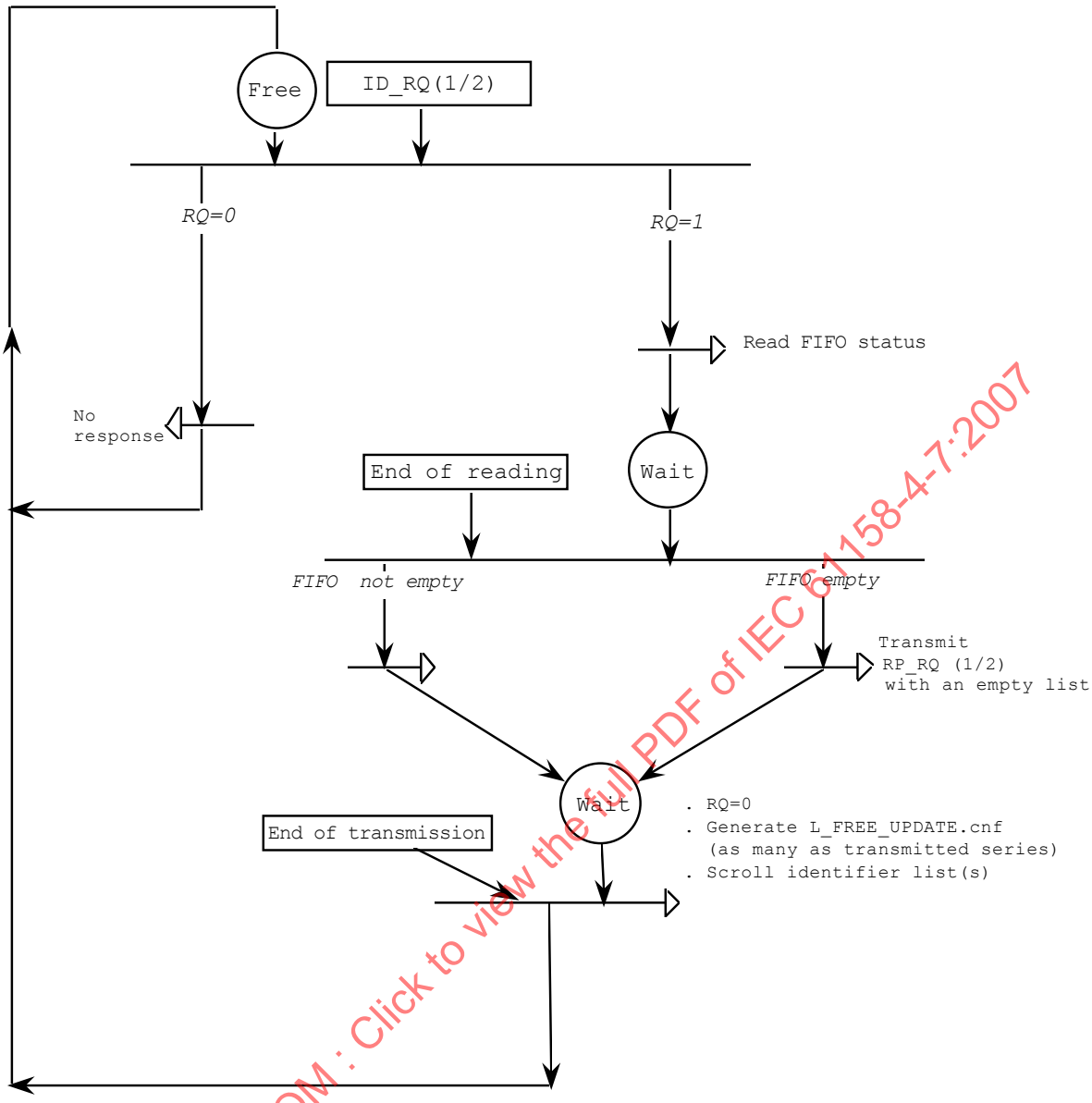
Si la demande explicite spécifiée fait partie du cycle de balayage périodique de l'administrateur de bus, la circulation de la première DLPDU est nulle et les autres DLPDU sont échangées pendant le balayage périodique.

6.6.2 Comportement avec des paramètres discordants

Le réseau d'évaluation de la Figure 34 permet de spécifier la gestion d'une entité productrice/consommatrice dans les cas suivants:

— Réception d'une DLPDU ID_RQ1 ou ID_RQ2 contenant un identificateur de DLCEP configuré pour le service de demande explicite libre de transfert de mémoire tampon et qui n'est pas associé ni à Q_REQ1 ni à Q_REQ2 (Le RQ de l'identificateur a une valeur nulle (0)).

— Réception d'une DLPDU ID_RQ1 ou ID_RQ2 contenant un identificateur de DLCEP associé à une file d'attente Q_REQ1 ou Q_REQ2 vide.



Légende

Anglais	Français
Free	Libre
Read FIFO status	Lire le statut de la FIFO
End of reading	Fin de lecture
Wait	Attendre
FIFO not empty	FIFO non vide
FIFO empty	FIFO vide
No response	Pas de réponse
Transmit RP_RQ1(1/2) with an empty list	Émettre RP_RQ1(1) avec une liste vide
End of transmission	Fin d'émission
Generate L_FREE_UPDATE.cnf (as many as transmitted series)	Générer L_FREE_UPDATE.cnf (autant de fois que de séries émises)
Scroll identifier list(s)	Parcourir la/les liste(s) d'identificateurs

Figure 34 – Réseau d'évaluation pour une demande explicite libre

Après l'émission du RP_RQ, la demande est détachée du séparateur concerné (RQ mis à 0); Si la file d'attente FIFO n'est pas vide, la demande sera jointe durant la prochaine réception d'un ID_DAT concernant un identificateur de DLCEP configuré pour ce service. Le bit RQ de cet identificateur sera mis à 1.

6.7 Messagerie

6.7.1 Introduction générale aux services de messages

Les services de messages permettent d'échanger les messages avec et sans acquittement.

L'émetteur de la demande d'échange de messages est l'utilisateur de DLS qui est la source du message. La demande est remplie, soit pendant le balayage apériodique soit pendant le cycle de balayage périodique de l'administrateur de bus, en fonction de la configuration.

Si la demande est remplie pendant le balayage apériodique, le message est considéré comme apériodique. Une seule ressource est mise de côté pour ce type de message: une file d'attente qui est liée de façon dynamique à un ou plusieurs identificateurs configurés pour ce mode de transfert de messages. Les DLPDU de réponse de variables associées à ces identificateurs portent les demandes de transfert de messages. Les demandes sont remplies dans la fenêtre apériodique.

Si la demande est remplie pendant le balayage périodique, elle est considérée comme étant cyclique, et la demande de transfert de messages a lieu dans la fenêtre périodique par configuration. Les ressources sont allouées lors de la configuration et elles ne changent pas. La ressource mise de côté est une file d'attente jointe à un identificateur de DLCEP configuré pour ce mode de transfert. La DLPDU d'identificateur de message associée à cet identificateur est émise dans la fenêtre périodique.

Un identificateur unique ne peut pas être configuré pour les deux modes de transfert de messages.

Les primitives de service pour les demandes de transfert de messages sont identiques pour les modes apériodiques et cycliques. Seulement un paramètre de la primitive "request" indique le mode demandé du transfert de message et permet de faire appel au mécanisme approprié au sein de la DLE source.

Pour mettre les entités communicantes en contact, deux adresses sont données pendant la transaction de message:

— une adresse destination (point d'accès pour un service de message) constituée de 24 bits qui encodent le numéro de la liaison locale de l'entité et les sous adresses de l'entité dans ce segment de DL;

— une adresse source (point d'accès pour un service de message) constituée de 24 bits qui encodent le numéro de la liaison locale de l'entité et les sous-adresses de l'entité dans ce segment de DL.

Un point d'accès pour un service de message attribue des adresses à une entité de service de message d'application pour l'émission (source) et pour la réception (destination). Chaque adresse est unique dans le système.

Le format de l'adresse destination et de l'adresse source est décrit en détail dans l'annexe.

Chaque entité détecte la DLPDU de réponse de message (RP_MSG_NOACK ou RP_MSG_ACK) et vérifie son adresse destination pour déterminer si elle la destination du message ou pas.

Un mécanisme de rétablissement intervient lors de la détection de la perte d'une DLPDU d'acquittement liée à un transfert de message avec acquittement.

Ce mécanisme de rétablissement en cas de perte d'acquittement, ainsi que les compteurs de redémarrage se trouvent dans l'entité source. Le nombre de redémarrages autorisés est le même pour toutes les entités sources dans le système, et ce nombre peut être défini lors de la configuration. Le mécanisme de rétablissement peut engendrer la duplication de messages. Un algorithme pour compter les messages entre l'entité source et l'entité de destination est utilisé pour détecter une duplication de message provoquée par le mécanisme de rétablissement (voir 4.4.3).

6.7.2 Diagramme d'interactions

Le fait de demander le service de demande de transfert de message sans acquittement entraîne l'ajout d'un message à la file d'attente pour les messages qui seront émis. Cette file d'attente est désignée par le paramètre SPECIFIED_IDENTIFIER. Comment l'échange est effectué dépend du type de la file d'attente.

6.7.2.1 Transferts apériodiques

La file d'attente est mise de côté pour le transfert apériodique. Le transfert inclut quatre phases:

phase 1:

À la réception d'un identificateur de DLCEP qui peut prendre en charge une demande apériodique, et qui n'est pas encore utilisée, l'entité source indique dans le champ de contrôle de la DLPDU de réponse de variable sa demande que l'administrateur de bus fait circuler une DLPDU de transfert de message, à condition que le nombre de places occupées dans la file d'attente soit inférieur au nombre d'identificateurs associés à cette ressource.

De plus, pour tous les identificateurs déjà utilisés, l'entité source maintient l'indication d'une demande de transfert de message dans la DLPDU de réponse de variable.

phase 2:

L'administrateur de bus donne le contrôle de la liaison locale à l'entité source en émettant une adresse de DLSAP de message dans la fenêtre allouée aux messages.

phase 3:

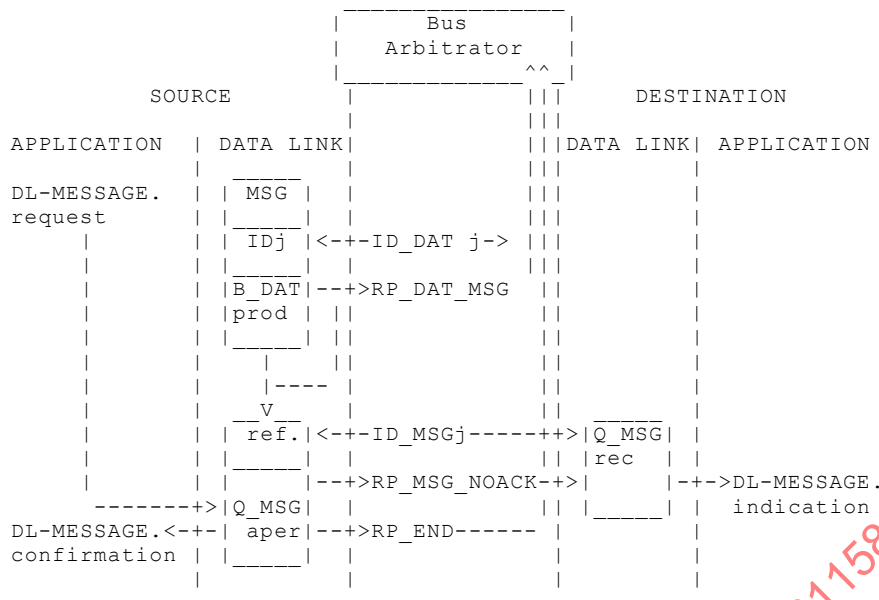
L'entité source émet le premier message stocké dans la file d'attente des messages qui seront émis. L'entité émet le message dans une DLPDU de réponse de message dont la DLPDU de contrôle étendue contient l'adresse destination et l'adresse source du message.

phase 4:

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signifie la fin de transaction de message. Une primitive «confirm» de DL-MESSAGE est envoyée à l'utilisateur de DLS (Le paramètre SPECIFIED_IDENTIFIER de cette primitive a la valeur NIL).

NOTE Si l'entité source n'a aucun message associé à la DLPDU d'identificateur de message circulée (file d'attente vide), elle retourne la commande de la liaison locale immédiatement à l'administrateur de bus et la phase 3 est éliminée.

Ces phases sont représentées schématiquement dans la Figure 35.



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
Source	Source
Destination	Destination
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 35 – Diagramme d'interactions dans un service de demande de transfert de message sans acquittement pour un transfert apériodique

6.7.2.2 Transferts cycliques

La ressource disponible pour la réception est une file d'attente nommée Q_MSGrec. La file d'attente est mise de côté pour les transferts cycliques.

Il y a trois phases pour les transferts:

phase 1:

L'administrateur de bus donne à l'entité source le contrôle de la liaison locale en émettant une DLPDU d'identificateur de message dans la fenêtre périodique.

phase 2:

L'entité source émet la DLPDU de réponse de message dont le champ de contrôle étendu contient les adresses destination et source du message.

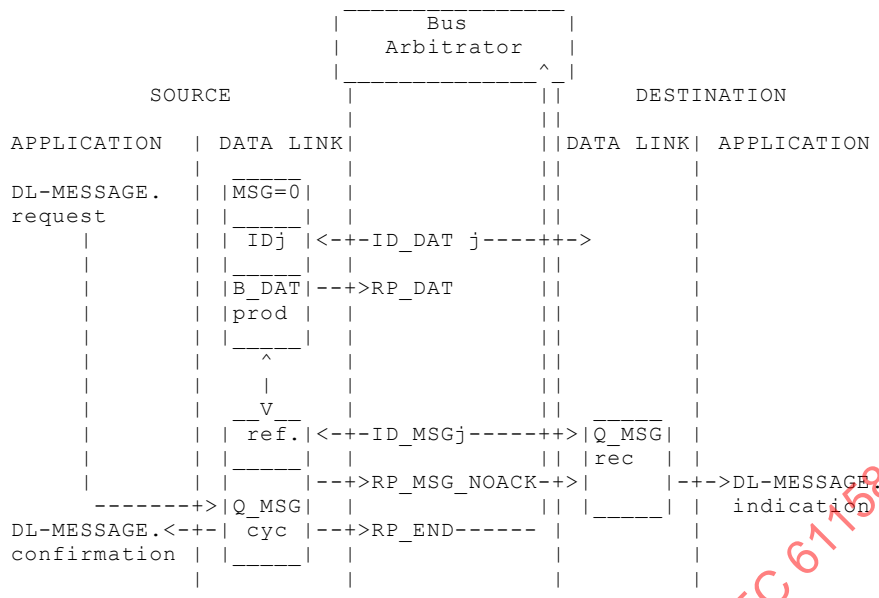
phase 3

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signifie la fin de transaction de message. Une primitive «confirm» DL-MESSAGE est envoyée à l'utilisateur de DLS (Le paramètre SPECIFIED_IDENTIFIER de cette primitive correspond à l'identificateur dans la DLPDU d'identificateur de message).

NOTE Si l'entité source n'a aucun message associé à la DLPDU d'identificateur de message circulée (file d'attente vide), elle retourne la commande de la liaison locale immédiatement à l'administrateur de bus et la phase 2 est éliminée.

La ressource disponible pour la réception est une file d'attente nommée Q_MSGrec.

Ces phases sont représentées schématiquement dans la Figure 36.



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
Source	Source
Destination	Destination
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 36 – Diagramme d'interactions dans un service de demande de transfert de message sans acquittement pour un transfert cyclique

6.7.3 DLPDU associées

Une demande de transfert de message sans acquittement implique la circulation de quatre DLPDU en cas d'un transfert apériodique de message. Dans le cas d'un transfert de message cyclique, la première DLPDU n'est pas diffusée.

6.7.3.1 Transfert apériodique de messages

Dans le cas d'un transfert apériodique de messages, l'entité source diffuse une DLPDU de réponse de variable dont la DLPDU de contrôle contient la demande que l'administrateur de bus fait circuler une DLPDU d'identificateur de message (RP_DAT_MSG ou RP_DAT_RQi_MSG). La première de ces DLPDU fait partie d'un trafic cyclique normal.

L'identificateur se réfère à la file d'attente pour les messages en attente d'être émis, qui est mise de côté pour le transfert apériodique de messages.

L'administrateur de bus donne le contrôle de la liaison locale à l'entité source durant une fenêtre apériodique de messages en diffusant une DLPDU d'identificateur de message (ID_MSG) avec un identificateur de DLCEP qui a transporté une demande.

Le message est émis par l'entité source (RP_MSG_NOACK) qui énonce dans le champ de contrôle étendu les adresses destination et source du message.

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signale la fin de transaction de message (RP_END).

À la réception d'une PDU contenant une adresse destination qui est homogène avec une adresse de groupe, la couche 2 consulte le répertoire local (par des fonctions spécifiques) afin d'avoir la correspondance entre cette adresse de groupe et une ou plusieurs adresses locales individuelles auxquelles est envoyée la primitive «indication» de "DL-MESSAGE indication(Ad,As,m)" avec l'adresse de groupe contenue dans la PDU en tant qu'adresse destination.

La Figure 37 montre la séquence de DLPDU telle qu'elle se produit dans la fenêtre périodique et la fenêtre apériodique:

Légende

Anglais	Français
In the periodic window:	Dans la fenêtre périodique
medium allocation for identified variables	Allocation du support pour des variables identifiées
Identifier	Identificateur
followed by a response plus message request	suivie par une réponse plus une demande de message
value of the variable	Valeur de la variable
or	ou
and then a number of identifiers later, in the aperiodic window	et puis un nombre d'identificateurs plus tard, dans la fenêtre apériodique
Medium allocation for a message	Allocation du support pour un message
followed by an unacknowledged message + destination address + source address	suivie par un message sans acquittement + adresse destination + adresse source

Anglais	Français
after which the source entity returns control to the bus-arbitrator	après lequel l'entité source retourne le contrôle à l'administrateur de bus

Figure 37 – Séquence de DLPDU pour un transfert apériodique de messages

6.7.3.2 Transfert cyclique de messages

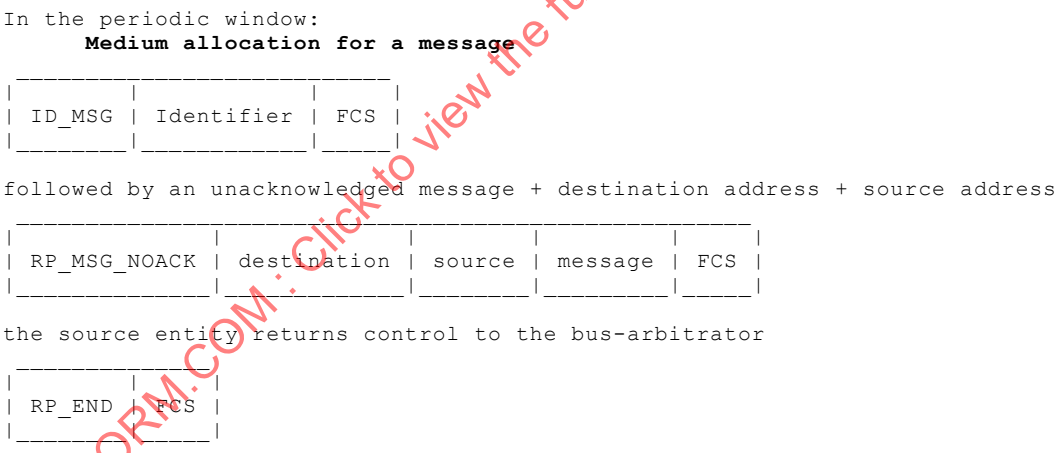
Dans le cas d'un transfert cyclique de messages, l'administrateur de bus donne le contrôle à l'entité source en diffusant une DLPDU d'identificateur de message (ID_MSG) dans la fenêtre périodique.

Le message est émis par l'entité source (RP_MSG_NOACK) qui note dans le champ de contrôle étendu les adresses destination et source.

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signale la fin de transaction de message (RP_END).

À la réception d'une PDU contenant une adresse destination qui est homogène avec une adresse de groupe, la couche 2 consulte le répertoire local (par des fonctions spécifiques) afin d'avoir la correspondance entre cette adresse de groupe et une ou plusieurs adresses locales individuelles auxquelles est envoyée la primitive «indication» de "DL-MESSAGE indication(Ad,As,m)" avec l'adresse de groupe contenue dans la PDU en tant qu'adresse destination.

L'allocation du support pour une variable identifiée est montrée à la Figure 38.



Légende

Anglais	Français
In the periodic window:	Dans la fenêtre périodique:
Medium allocation for a message	Allocation du support pour un message
Identifier	Identificateur
followed by an unacknowledged message + destination address + source address	suivie par un message sans acquittement + adresse destination + adresse source
the source entity returns control to the bus-arbitrator	l'entité source retourne le contrôle à l'administrateur de bus

Figure 38 – Séquence de DLPDU pour un transfert cyclique de messages

6.8 Messagerie avec acquittement

6.8.1 Diagramme d'interactions

Le fait de demander le service de demande de transfert de message avec acquittement (demande DL-MESSAGE-ACK) entraîne l'ajout d'un message à la file d'attente des messages qui seront émis. Cette file d'attente est désignée par le paramètre SPECIFIED_IDENTIFIER. La manière dans laquelle les échanges sont effectués dépend du type de la file d'attente.

6.8.1.1 File d'attente dédiée à un transfert apériodique

Si la file d'attente est mise de côté pour un transfert apériodique, l'échange inclut cinq phases:

phase 1:

À la réception d'un identificateur de DLCEP qui peut prendre en charge une demande apériodique, et qui n'est pas encore utilisée, l'entité source indique dans le champ de contrôle de la DLPDU de réponse de variable sa demande que l'administrateur de bus fait circuler une DLPDU de transfert de message, à condition que le nombre de places occupées dans la file d'attente soit inférieur au nombre d'identificateurs associés à cette ressource.

De plus, pour tous les identificateurs déjà utilisés, l'entité source maintient l'indication d'une demande de transfert de message dans la DLPDU de réponse de variable.

phase 2:

L'administrateur de bus donne le contrôle à l'entité source en émettant une adresse de DLSAP de message dans la fenêtre allouée aux messages.

phase 3:

L'entité source émet le premier message stocké dans la file d'attente des messages qui seront émis. L'entité émet le message dans une DLPDU de réponse de message dont la DLPDU de contrôle étendue contient l'adresse destination et l'adresse source du message.

phase 4:

L'entité destination émet la DLPDU d'acquittement immédiatement, sauf si l'adresse destination est une adresse de groupe. Dans ce cas, l'entité ignore la DLPDU.

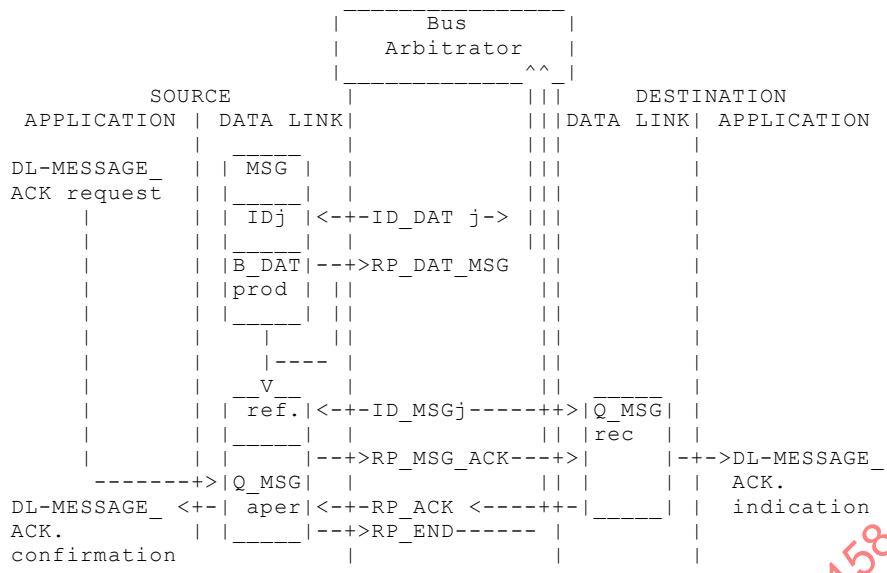
phase 5:

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signifie la fin de transaction de message. Une primitive «confirm» de DL-MESSAGE-ACK est envoyée à l'utilisateur de DLS (Le paramètre SPECIFIED_IDENTIFIER de cette primitive a la valeur NIL).

NOTE Si l'entité source n'a aucun message associé à la DLPDU d'identificateur de message circulée (file d'attente vide), elle retourne la commande de la liaison locale immédiatement à l'administrateur de bus et les phases 3 et 4 sont éliminées.

La ressource disponible pour la réception est une file d'attente nommée Q_MSGrec.

Ces phases sont représentées schématiquement dans la Figure 39.



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
SOURCE	SOURCE
DESTINATION	DESTINATION
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 39 – Diagramme d'interactions dans un service de demande de transfert de message avec acquittement pour un transfert apériodique

6.8.1.2 File d'attente dédiée aux transferts cycliques

La file d'attente est mise de côté pour les transferts cycliques. Il y a quatre phases pour les transferts:

phase 1:

L'administrateur de bus donne à l'entité source le contrôle de la liaison locale en émettant une DLPDU d'identificateur de message dans la fenêtre périodique.

phase 2:

L'entité source émet la DLPDU de réponse de message dont le champ de contrôle étendu contient les adresses destination et source du message.

phase 3:

L'entité destination émet la DLPDU d'acquittement immédiatement, sauf si l'adresse destination est une adresse de groupe. Dans ce cas, l'entité ignore la DLPDU.

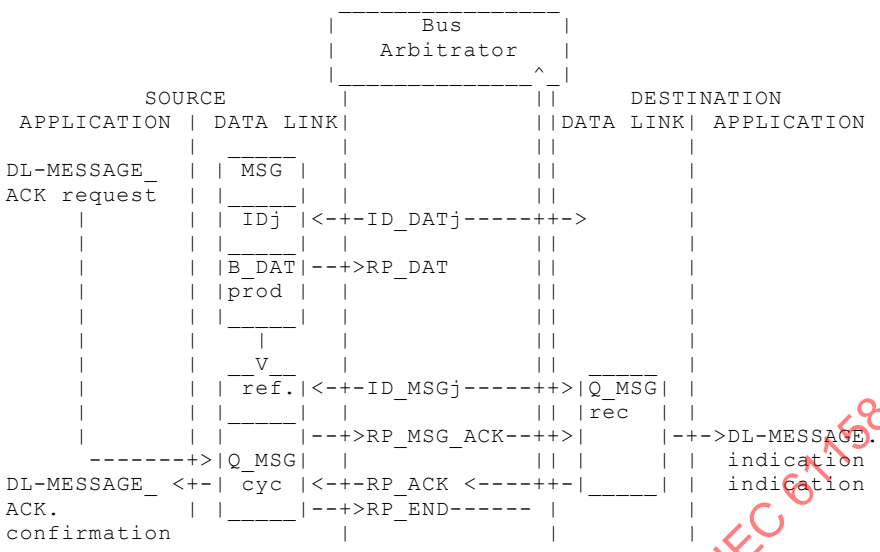
phase 4:

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signifie la fin de transaction de message. Une primitive «confirm» de DL-MESSAGE-ACK est envoyée à l'utilisateur de DLS. (Le paramètre SPECIFIED_IDENTIFIER de cette primitive correspond à l'identificateur dans la DLPDU d'identificateur de message).

NOTE Si l'entité source n'a aucun message associé à la DLPDU d'identificateur de message circulée (file d'attente vide), elle retourne la commande de la liaison locale immédiatement à l'administrateur de bus et les phases 2 et 3 sont éliminées.

La ressource disponible pour la réception est une file d'attente nommée Q_MSGrec.

Ces phases sont représentées schématiquement dans la Figure 40.



Légende

Anglais	Français
Bus Arbitrator	Administrateur de Bus
SOURCE	SOURCE
DESTINATION	DESTINATION
APPLICATION	APPLICATION
DATA LINK	LIAISON DE DONNÉES

Figure 40 – Diagramme d'interactions dans un service de demande de transfert de message avec acquittement pour un transfert cyclique

6.8.2 DLPDUs associées

Une demande de transfert de message avec acquittement implique la circulation de cinq DLPDU en cas d'un transfert apériodique de message. Dans le cas d'un transfert de message cyclique, la première DLPDU n'est pas diffusée.

6.8.2.1 Transfert d'un message apériodique

Dans le cas d'un transfert de message apériodique, l'entité source diffuse une DLPDU de réponse de variable dont la DLPDU de contrôle contient la demande que l'administrateur de bus fait circuler une DLPDU d'identificateur de message (RP_DAT_MSG ou RP_DAT_RQi_MSG). La première de ces DLPDU fait partie d'un trafic cyclique normal.

L'identificateur se réfère à la file d'attente pour les messages en attente d'être émis, qui est mise de côté pour le transfert apériodique de messages.

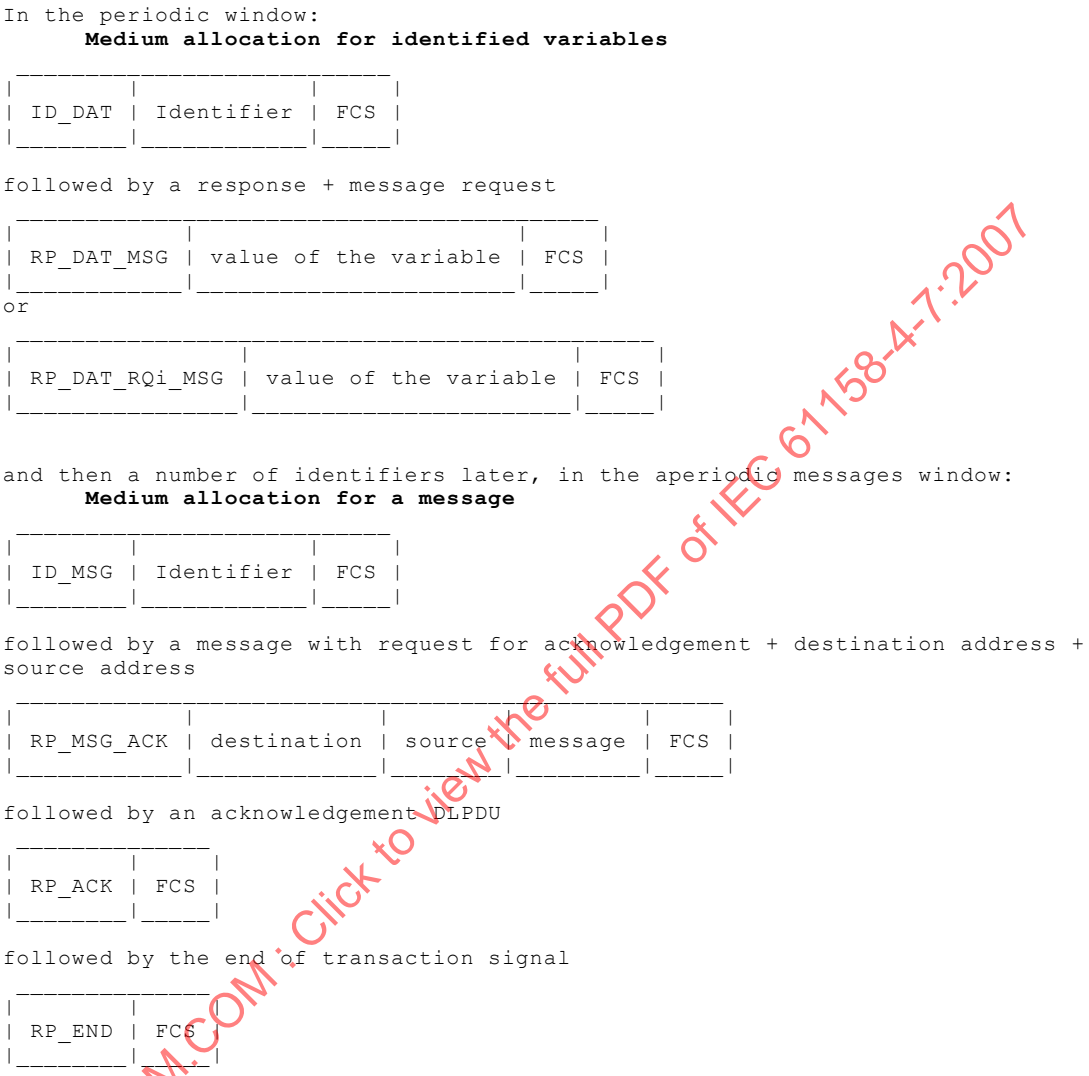
L'administrateur de bus donne le contrôle de la liaison locale à l'entité source durant une fenêtre apériodique de messages en diffusant une DLPDU d'identificateur de message (ID_MSG) avec l'identificateur qui a transporté la demande.

Le message est émis par l'entité source (RP_MSG_ACK) qui énonce dans le champ de contrôle étendu les adresses destination et source du message.

L'entité destination émet immédiatement la DLPDU d'acquittement.

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signale la fin de transaction de message (RP_END).

La Figure 41 montre la séquence de DLPDU telle qu'elle se produit dans la fenêtre périodique et la fenêtre apériodique:



Légende

Anglais	Français
In the periodic window:	Dans la fenêtre périodique:
medium allocation for identified variables	Allocation du support pour des variables identifiées
Identifier	Identificateur
followed by a response plus message request	suivie par une réponse plus une demande de message
value of the variable	Valeur de la variable
or	ou
and then a number of identifiers later, in the aperiodic window	et puis un nombre d'identificateurs plus tard, dans la fenêtre apériodique
Medium allocation for a message	Allocation du support pour un message
followed by a message with request for acknowledgement + destination address + source address	suivie par un message avec demande pour acquittement + adresse destination + adresse source
followed by an acknowledgement DLPDU	suivie par une DLPDU d'acquittement
followed by the end of transaction signal	suivie par la fin du signal de transaction

Figure 41 – Séquence de DLPDU pour un transfert apériodique de messages

6.8.2.2 Transfert cyclique de messages

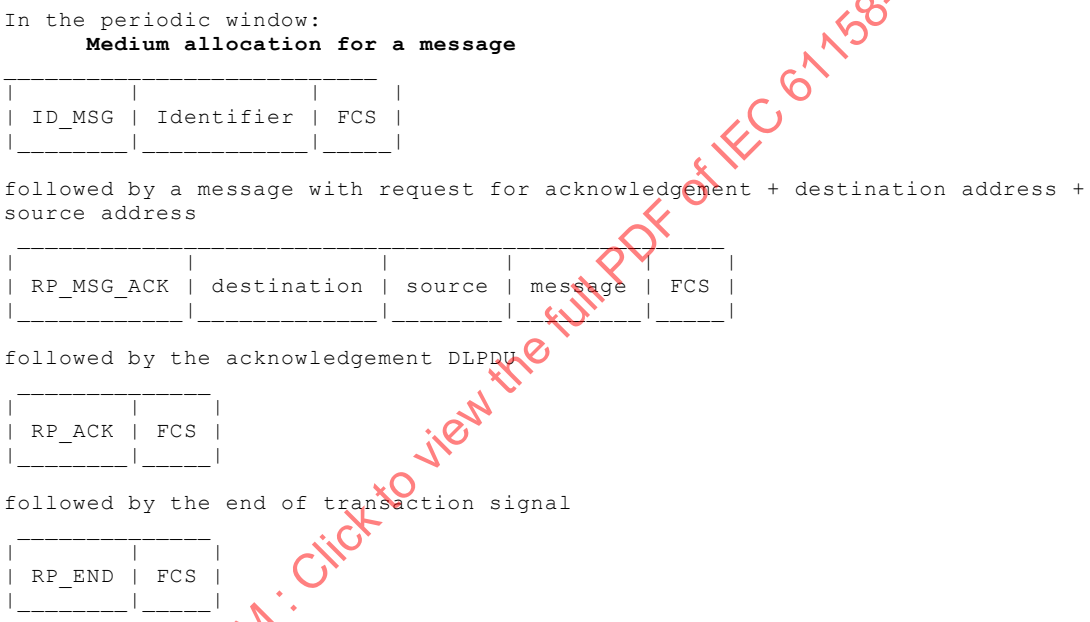
Dans le cas d'un transfert cyclique de messages, l'administrateur de bus donne le contrôle à l'entité source en diffusant une DLPDU d'identificateur de message (ID_MSG) dans la fenêtre périodique.

Le message est émis par l'entité source (RP_MSG_ACK) qui note dans le champ de contrôle étendu les adresses destination et source.

L'entité destination émet la DLPDU d'acquittement (RP_ACK) immédiatement.

L'entité source passe le contrôle à l'administrateur de bus en émettant une DLPDU qui signale la fin de transaction de message (RP_END).

La Figure 42 montre la séquence de DLPDU telle qu'elle se produit dans la fenêtre périodique:



Légende

Anglais	Français
In the periodic window:	Dans la fenêtre périodique:
Medium allocation for a message	Allocation du support pour un message
Identifier	Identificateur
followed by a message with request for acknowledgement + destination address + source address	suivie par un message avec demande pour acquittement + adresse destination + adresse source
followed by an acknowledgement DLPDU	suivie par une DLPDU d'acquittement
followed by the end of transaction signal	suivie par la fin du signal de transaction

Figure 42 – Séquence de DLPDU pour un transfert cyclique de messages

Les champs de contrôle RP_MSG_ACK et RP_ACK contenant le bit de numérotage Bp donné par l'algorithme défini en 6.9.

NOTE Ce protocole de DL fournit des services SDA et SDN tels que spécifiés dans la CEI 60955 (PROWAY C). Les primitives DLL correspondent aussi directement aux primitives spécifiées dans les documents ISO/CEI 8802-2 et ISO TC 97 SC6 WG1 N169. La primitive DL-MESSAGE correspond à la primitive DLSDU. La primitive DL-MESSAGE-ACK correspond à la primitive DLSDU_ACK.

6.9 Numérotage des messages avec acquittement

L'algorithme décrivant le protocole de numérotage de messages utilise les données suivantes:

— chaque entité destination enregistre l'adresse source (ADRsource_stoc) de la dernière DLPDU de réponse de message reçue, et le bit pair/impair (Bc) associé,

— à la réception d'une DLPDU autre que RP_MSG_ACK, les entités donnent à l'adresse source stockée la valeur "empty" (c'est-à-dire "vide"),

— le rétablissement est immédiat (pas d'autre transaction intermédiaire). Le rétablissement est fourni par l'entité source, qui gère une période d'attente pour la DLPDU d'acquittement et un compteur de redémarrage,

— l'entité source envoie le message avec l'indication Bp, et change la valeur de ce bit lors de l'émission de RP_END.

Lorsque le système est initialisé, chaque entité dispose des informations suivantes:

— ADRsource_stoc = “empty” (vide),

— Bp n'a pas de signification,

— Bc n'a pas de signification.

L'algorithme se compose de trois parties. Une partie est mise en œuvre par l'entité source de messages, une autre par l'entité destination, et la troisième partie par l'administrateur de bus.

6.9.1 Algorithme de la destination:

```
À la réception d'un RP_MSG_ACK,
comparer ADRsource reçue avec ADR source_stoc
si égale, comparer Bp reçue avec Bc
  si égale
    s'il y a une cellule pour stocker
       alors envoyer RP_ACK+Bp
            stocker le message
            Bc=1-Bp
    sinon envoyer RP_ACK-Bp
  sinon envoyer RP_ACK+Bp
      ignorer le message, car il est dupliqué
sinon
  s'il y a une cellule pour stocker
    envoyer RP_ACK+Bp
    Bc=1-Bp
    stocker le message
    ADRsource_stoc=ADR source reçue
  sinon envoyer RP_ACK-Bp

si réception d'une DLPDU autre que RP_MSG_ACK
    ADRsource_stoc=empty (vide)
```

6.9.2 Algorithme de la source:

À la réception d'un ID_MSG(IDk) et l'IDk est reconnu par l'entité source
envoyer le message avec une indication Bp
attendre RP_ACK

si réception de RP_ACK
définir MSG de IDk à 0
envoyer une confirmation avec statut (ack ±)
réamorcer le compteur de redémarrage à zéro
envoyer RP_END et Bp=1-Bp

si expiration du temps d'attente pour RP_ACK,
soumettre à l'essai le compteur de redémarrage

si compteur > nombre maximal de redémarrages autorisés
définir MSG de IDk à 0
réamorcer le compteur de redémarrage à zéro
envoyer une confirmation avec statut négatif
envoyer RP_END et Bp=1-Bp

si compteur ≤ nombre maximal de redémarrages autorisés
réémettre le même RP_MSG_ACK
incrémenter le compteur de redémarrage
attendre RP_ACK

6.9.3 Algorithme de l'administrateur de bus:

Après l'envoi de ID_MSG
L'administrateur de bus est en attente de RP_END:
si le temporisateur d'attente du RP_END a expiré
continuer vers l'identificateur suivant

si réception de RP_END
continuer vers l'identificateur suivant

si réception de DLPDU autre que RP_END, ou d'une DLPDU non valide
continuer à attendre RP_END

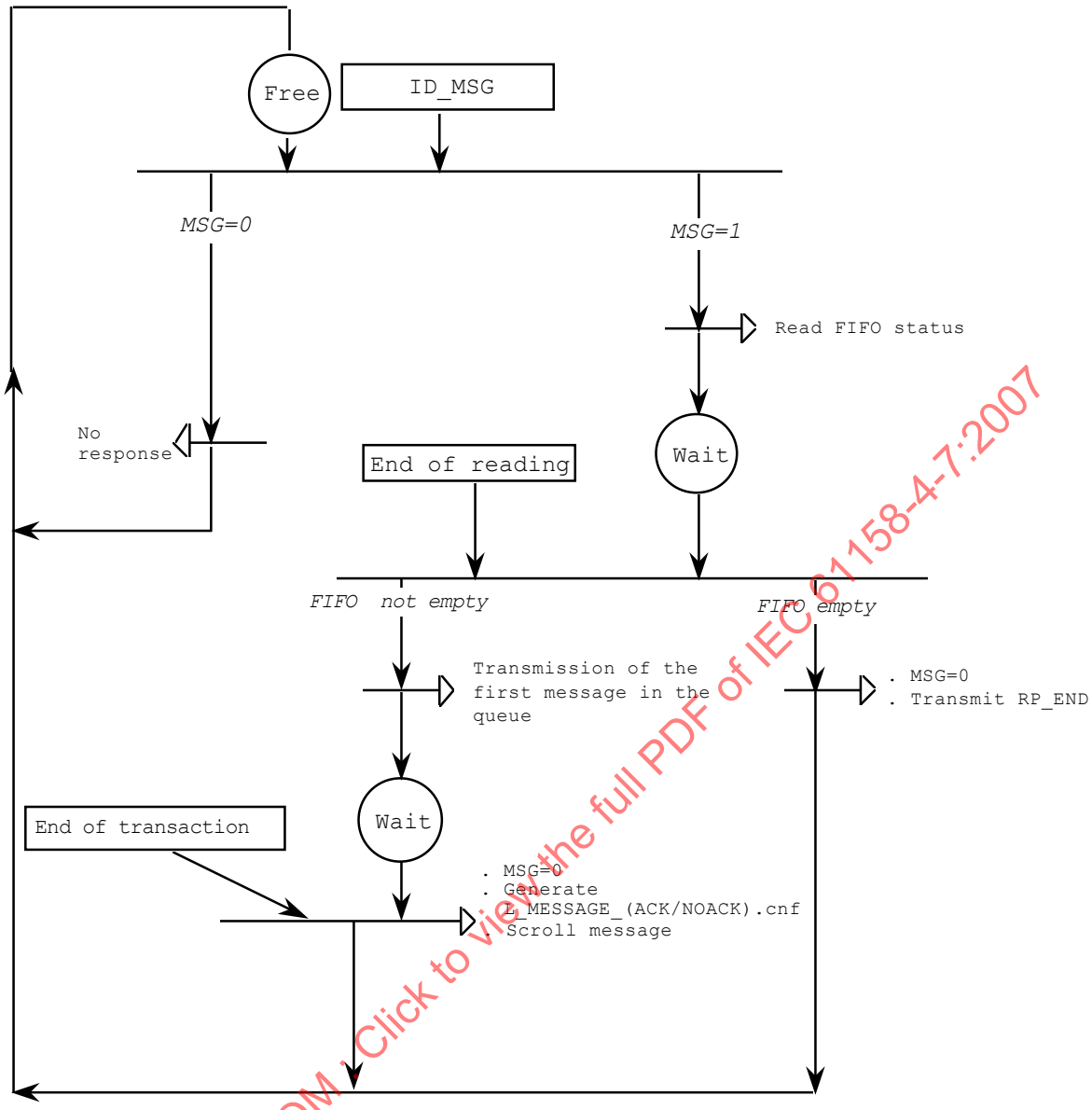
6.10 Comportement avec des paramètres discordants

6.10.1 Transfert apériodique de messages (avec ou sans acquittement)

Le réseau d'évaluation de la Figure 43 permet de spécifier le comportement d'une entité productrice/consommatrice dans les cas suivants.

— Réception d'une DLPDU ID_MSG contenant un identificateur de DLCEP configuré pour un transfert apériodique de messages et non associé à la file d'attente Q_MSG.aper (l'identificateur MSG a une valeur nulle (0)).

— Réception d'une DLPDU ID_MSG contenant un identificateur de DLCEP configuré pour un transfert apériodique de messages et associé à la file d'attente Q_MSG.aper (l'identificateur MSG a une valeur 1), la dernière étant vide.



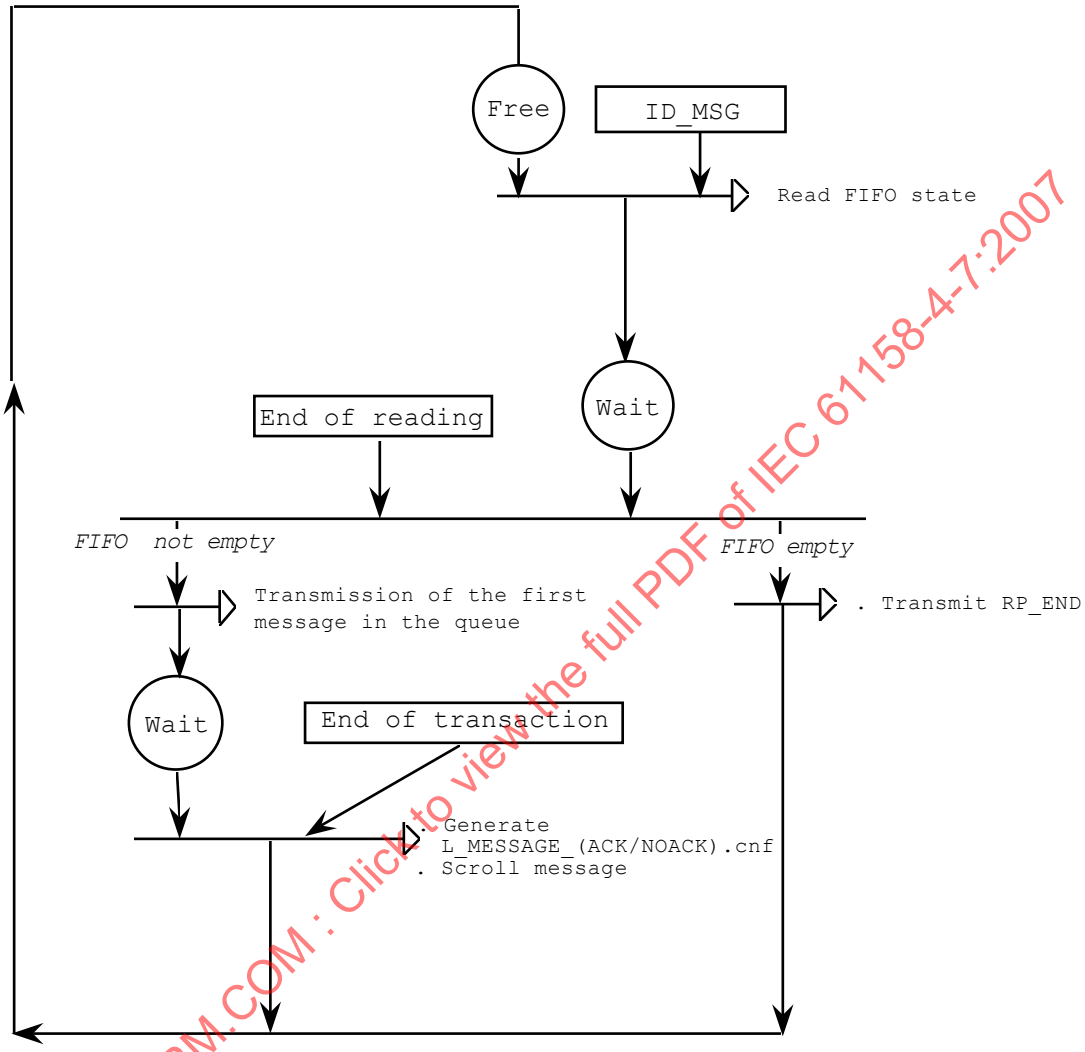
Légende

Anglais	Français
Free	Libre
Read FIFO status	Lire le statut de la FIFO
End of reading	Fin de lecture
FIFO not empty	FIFO non vide
FIFO empty	FIFO vide
No response	Pas de réponse
Wait	Attendre
Transmission of the first message in the queue	Émission du premier message dans la file d'attente
Transmit RP_END	Émettre RP_END
End of transmission	Fin d'émission
Generate	Générer
Scroll message	Parcourir le message

Figure 43 – Réseau d'évaluation pour un transfert apériodique de messages

6.10.2 Transfert cyclique de messages (avec ou sans acquittement)

Le réseau d'évaluation de la Figure 44 permet de spécifier le comportement d'une entité productrice/consommatrice, à la réception d'une DLPDU ID_MSG contenant un identificateur de DLCEP configuré pour un transfert cyclique de messages et dont la file d'attente Q_MSG.cyc est vide.



Légende

Anglais	Français
Free	Libre
Read FIFO state	Lire l'état de la FIFO
End of reading	Fin de lecture
FIFO not empty	FIFO non vide
FIFO empty	FIFO vide
Wait	Attendre
Transmission of the first message in the queue	Émission du premier message dans la file d'attente
Transmit RP_END	Émettre RP_END
End of transmission	Fin d'émission
Generate	Générer
Scroll message	Parcourir le message

Figure 44 – Réseau d'évaluation pour un transfert cyclique de messages

7 Éléments de procédure, interfaces et conformité de services de DL

7.1 Généralités

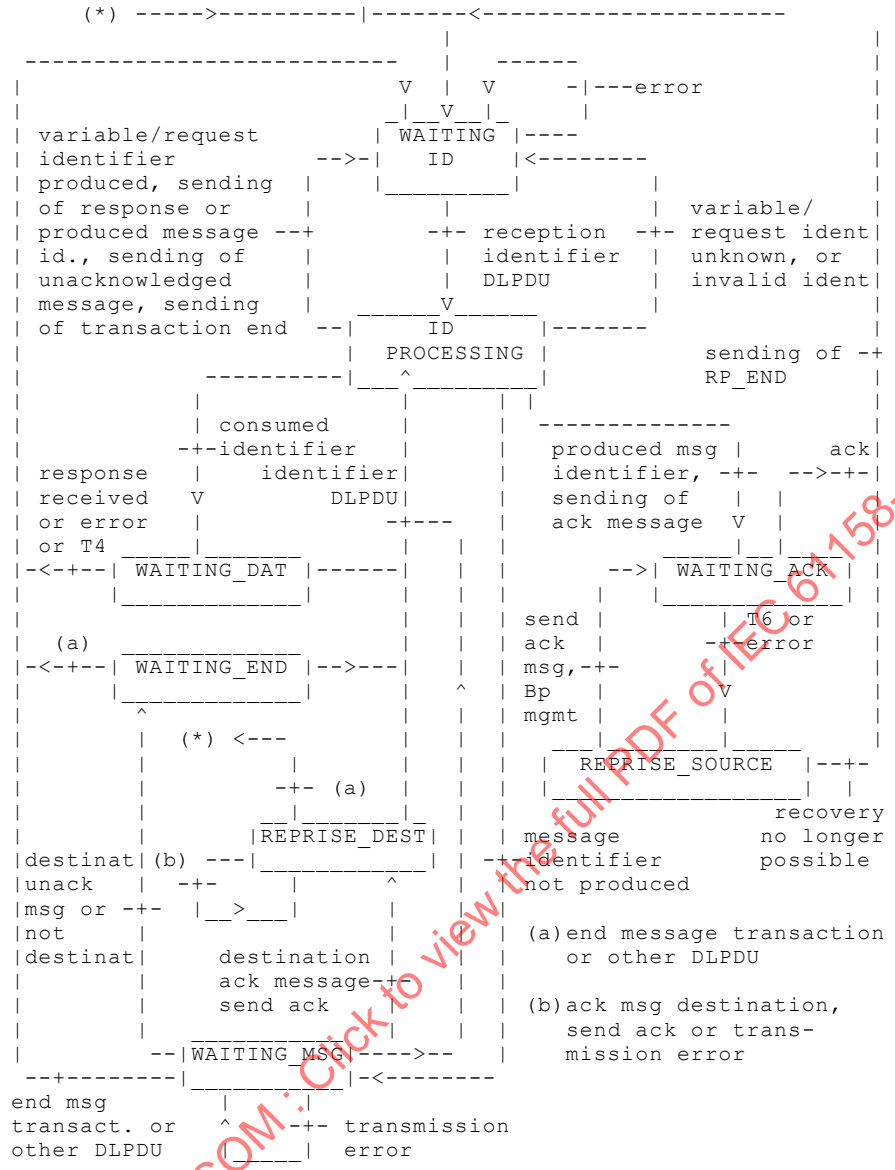
Ces diagrammes d'états finis décrivent le fonctionnement général d'une DLE en tant qu'une fonction des DLPDU circulant sur le support, et ils utilisent les chargeurs de temps définis ci-dessus.

Les tables de transition complètent la description du fonctionnement d'une DLE en énonçant des actions exécutées lors de la réception d'une DLPDU.

Les protocoles associés à chaque service fourni par la DLL sont décrits pour une entité productrice/consommatrice et pour l'administrateur de bus.

Les interfaces vers l'utilisateur de DLS, la gestion de DL et la PhL sont résumées et les classes de conformité de Type 7 sont montrées dans la Figure 45.

7.2 Entité productrice/consommatrice



Légende

Anglais	Français
error	erreur
variable/request ident unknown, or invalid ident	Identificateur de variable/demande inconnu, ou Identificateur non valide
reception identifier DLPDU	Réception de DLPDU identificatrice
variable/request identifier produced sending of response or produced message id., sending of unacknowledged message, sending of transaction end	Identificateur de variable/demande produit envoi d'id de message produit ou de réponse, transaction message sans acquittement, envoi de fin de transaction
sending of RP_END	envoi de RP_END
produced msg identifier,sending of ack message	Identificateur de message produit, envoi de message avec acquittement
consumed identifier	Identificateur consommé
identifier DLPDU	DLPDU identificatrice
response received or error or T4	Réponse reçue ou erreur ou T4
T6 or error	T6 ou erreur
send ack msg,Bp mgmt	Envoi de ack msg,Bp mgmt

Anglais	Français
recovery no longer possible	Rétablissement n'est plus possible
message identifier not produced	Identificateur de message non produit
(a) end message transaction or other DLPDU	(a) Fin de transaction de message ou autre DLPDU
(b)ack msg destination, send ack or transmission error	(b) destination de ack msg, envoyer ack ou erreur d'émission
Destinat unack msg or not destinat	Destination unack msg ou pas de destination
destination ack message send ack	Destination ack message envoyer ack
transmission error	erreur d'émission
end msg transact. or other DLPDU	Fin de transaction de message ou autre DLPDU

Figure 45 – Diagramme d'états simplifié pour une entité productrice/consommatrice

Le diagramme d'états indique le comportement d'une station pendant la réception et l'émission. Ce diagramme prend en compte dans ses différents états:

— la réception de différents types de DLPDU de liaison,

— les délais spécifiques à la station (ils ne sont pas ceux de l'administrateur de bus).

Il est important de noter que ces diverses DLPDU de liaison prises en compte dans les différents états de ce diagramme peuvent être prises en compte seulement dans la mesure où ce dernier observe au moins les caractéristiques temporelles du protocole.

a) Caractéristiques temporelles du protocole sans interruption.

En cas d'ordonnancement normal du protocole, les caractéristiques temporelles de ce dernier sont fixées par la valeur du temps de changement lors de la production des différentes stations. Chaque DLPDU reçue par un dispositif qui doit fournir une réponse est séparée de celle-ci sur le bus par un intervalle de temps égal à son propre temps de changement pendant la production.

NOTE Sachant que le dispositif qui a le temps de changement maximal pendant la production conditionne le choix de l'expiration du délai T0.

b) Caractéristiques temporelles du protocole en cas d'interruption

Si l'ordonnancement de DLPDU est dégradé pour une raison ou une autre (perturbation sur le bus, absence de production), les caractéristiques temporelles du protocole sont fixées par l'administrateur de bus.

Par exemple, un ID_XX DLPDU doit être pris en compte dans les différents états de réception seulement si l'administrateur de bus observe les contraintes temporelles suivantes.

— Dans l'état WAIT_DAT, l'administrateur de bus doit attendre au moins T1=T0 avant de procéder à l'émission d'une ID_XX. Sachant que le délai T1 au niveau de l'administrateur de bus est défini par le silence de la ligne suite à l'émission d'une DLPDU ID_DAT et qu'il ne peut expirer que si le silence est maintenu tout au long de sa vie.

— Dans l'état MSG_WAIT ou END_WAIT ou DEST_RESTART, l'administrateur de bus doit atteindre au moins T5=2T0, en cas de non-réception de RP_END avant de procéder à l'émission d'une DLPDU ID_XX. Sachant que le délai T5 au niveau de l'administrateur de bus est défini par le silence de chaque ligne suite à l'émission d'une DLPDU ID_MSG et qu'il ne peut expirer que si le silence est maintenu tout au long de sa vie.

NOTE Dans le cas d'un administrateur qui a identifié la cause de cette interruption (par exemple, producteur absent) et qui voudrait se retourner immédiatement pour passer à l'émission de l'identificateur suivant, le fonctionnement du protocole n'est pas garanti.

7.2.1 Table de transitions pour une entité productrice/consommatrice

```
WAITING_ID (statut initial)
   ID_xx:                                                --> PROCESSING_ID

   autre DLPDU:                                             --> WAITING_ID
   erreur d'émission:                                       --> WAITING_ID


PROCESSING_ID
   ID_DAT,ID produced:
      si l'identificateur est non valide, pas de réponse
                                                            --> WAITING_ID

      si un indicateur d'une demande explicite urgente émettre RP_DAT_RQ1
      si un indicateur d'une demande explicite normale émettre RP_DAT_RQ2
      si un indicateur d'une demande de message émettre RP_DAT_MSG
      si 2 indicateurs émettre RP_DAT_RQi_MSG
      sinon émettre RP_DAT

      générer une indication DL-SENT (ID produced)          --> WAITING_ID

   ID_DAT,ID consumed: activer T4                           --> WAITING_DAT
   ID_DAT, nouvel ID:                                       --> WAITING_ID

   ID_MSG,ID produced:
      si msg existe et il est unack, émettre RP_MSG_NOACK
                                émettre RP_END              --> WAITING_ID

      si msg existe et il est ack, émettre RP_MSG_ACK
                                activer T6                  --> WAITING_ACK

      si msg n'existe pas, émettre RP_END                   --> WAITING_ID

   ID_MSG, nouvel ID:                                       --> WAITING_MSG

   ID_RQ1,ID produced:
      émettre RP_RQ1
      générer DL-SPEC_UPDATE confirm (ID produced, OK)
      ou DL-FREE_UPDATE confirm (OK)                        --> WAITING_ID

   ID_RQ1, nouvel ID:                                       --> WAITING_ID

   ID_RQ2, ID produced:
      émettre RP_RQ2
      générer DL-FREE_UPDATE confirm (OK)                   --> WAITING_ID

   ID_RQ2, nouvel ID:                                       --> WAITING_ID


WAITING_DAT
   RP_DAT ou RP_DAT_MSG ou RP_DAT_RQi ou RP_DAT_RQi_MSG:
      stocker la valeur de la variable
      générer une indication DL-RECEIVED  (ID consumed)
                                                            --> WAITING_ID

   ID_xx:                                                --> PROCESSING_ID

   autre DLPDU:                                             --> WAITING_ID
   erreur d'émission:                                       --> WAITING_ID
   temporisateur T4: indiquer transaction interrompue       --> WAITING_ID
```

```
WAITING_MSG
   RP_MSG_NOACK, mon adresse est la destination:
      recevoir le message
      générer une indication DL-MESSAGE (ADRdestination, ADRsource,
      message)                                                --> WAITING_END

   RP_MSG_NOACK, autre adresse destination:                   --> WAITING_END

   RP_MSG_ACK, mon adresse est la destination:
      générer le bit de numérotage de message
      recevoir msg, envoyer RP_ACK
      générer une indication DL-MESSAGE_ACK (ADRdestination, ADRsource,
      message)                                              --> RECOVERY_DEST
   RP_MSG_ACK, autre adresse destination                      --> WAITING_END

   RP_END: fin de la transaction de message                    --> WAITING_ID

   ID_xx:                                                  --> PROCESSING_ID

   autre DLPDU:                                                --> WAITING_ID
   erreur d'émission:                                          --> WAITING_MSG


WAITING_ACK
   RP_ACK:
      générer le bit de numérotage de message (voir 6.9)
      générer une DL-MESSAGE_ACK confirm
      (ADRdestination, ADRsource, status)
   émettre RP_END                                              --> WAITING_ID

   temporisateur T6:                                   --> RECOVERY_SOURCE

   autre DLPDU:                                        --> RECOVERY_SOURCE
   erreur d'émission:                                  --> RECOVERY_SOURCE


WAITING_END
   RP_END:                                                     --> WAITING_ID

   ID_xx                                                   --> PROCESSING_ID

   autre DLPDU:                                                --> WAITING_ID
   erreur d'émission:                                          --> WAITING_ID


RECOVERY_DESTINATION
   RP_END:                                                     --> WAITING_ID

   RP_MSG_ACK: gestion de Bp (voir 6.9)
               envoyer ack                                 --> RECOVERY_DEST

   ID_xx                                                   --> PROCESSING_ID

   autre DLPDU:                                                --> WAITING_ID
   erreur d'émission:                                      --> RECOVERY_DEST


RECOVERY_SOURCE
   si le rétablissement est possible
      émettre le même RP_MSG_ACK
      activer T6                                           --> WAITING_ACK
   sinon envoyer RP_END
      indiquer transaction interrompue                         --> WAITING_ID
```


NOTE Les types d'erreurs sont définis en 4.6.22 (description de la classe DLL-counter) de l'EN 50170-7.

7.3 Éléments de protocole par service

NOTE Les protocoles associés aux services fournis par une entité productrice/consommatrice sont spécifiés en décrivant le comportement de l'entité pour chaque primitive échangée avec l'utilisateur de DLS.

7.3.1 Réception d'une DL-PUT request

À la réception d'une primitive DL-PUT request (IDk, DLSDU), un producteur déclenche les actions suivantes:

* essai sémantique (identificateur inconnu ou longueur incorrecte)

— sémantique incorrecte

émission immédiate de DL-PUT confirm (IDk, identificateur inconnu ou longueur de DLSDU incorrecte)

— sémantique correcte

* soumettre à l'essai la validité de l'identificateur

— identificateur non valide

émission immédiate de DL-PUT confirm (IDk, identificateur non valide)

— identificateur valide

* soumettre à l'essai la disponibilité de la mémoire tampon

— mémoire tampon non disponible,

émission immédiate de DL-PUT confirm (IDk, mémoire tampon non disponible)

— mémoire tampon disponible,

écrire la nouvelle valeur

émission immédiate de DL-PUT confirm (IDk, OK)

Les différents essais réalisés se basent sur les données concernant:

— reconnaissance du paramètre identificateur,
— l'identificateur est connu s'il a été déclaré comme un identificateur produit par l'entité productrice/consommatrice,
— la compatibilité de la longueur de DLSDU avec la mémoire tampon B_DATprod (longueur inférieure à 128 octets),
— validité de l'identificateur,
— un identificateur de DLCEP est valide si la configuration liaison de données de la variable associée est valide,
— disponibilité de la mémoire tampon,
— mémoire tampon non disponible si la même mémoire tampon B_DATprod a déjà été accédée par la liaison locale. Afin de gérer ce conflit, la norme recommande de donner à la liaison locale la plus haute priorité dans tous les cas.

7.3.2 Réception d'une DL-GET request

À la réception d'une primitive DL-GET request (IDk), un producteur déclenche les actions suivantes:

* soumettre à l'essai l'identificateur IDk

— identificateur inconnu

émission immédiate de DL-GET confirm (IDk, identificateur inconnu)

— identificateur connu

* soumettre à l'essai la disponibilité de la mémoire tampon

— mémoire tampon non disponible,

émission immédiate de DL-GET confirm (IDk, mémoire tampon non disponible)

— mémoire tampon disponible

* soumettre à l'essai la validité de l'identificateur

— identificateur non valide

émission immédiate de DL-GET confirm (IDk, identificateur non valide)

— identificateur valide

émission immédiate de DL-GET confirm (IDk, DLSDU, lecture OK).

Les différents essais réalisés se basent sur les données concernant:

— reconnaissance du paramètre identificateur

l'identificateur est connu s'il a été déclaré comme un identificateur consommé par l'entité productrice/consommatrice,

— disponibilité de la mémoire tampon

mémoire tampon non disponible si la même mémoire tampon B_DATcons a déjà été accédée par la liaison locale. Afin de gérer ce conflit, la norme recommande de donner à la liaison locale la plus haute priorité dans tous les cas,

— validité de l'identificateur,

— un identificateur de DLCEP est valide si la configuration de liaison de données de la variable associée est valide.

7.3.3 Réception d'une DL-SPEC-UPDATE request

À la réception d'une primitive DL-SPEC-UPDATE (IDk, LIST), un producteur déclenche les actions suivantes:

* soumettre à l'essai IDk

— IDk inconnu ou non configuré pour ce service

émission immédiate de DL-SPEC-UPDATE confirm (IDk, identificateur inconnu)

— IDk connu et configuré pour ce service

* soumettre à l'essai la disponibilité de la mémoire tampon B_REQ (IDk)

— mémoire tampon non disponible

émission immédiate de DL-SPEC-UPDATE confirm (IDk, mémoire tampon non disponible)

— mémoire tampon disponible

* soumettre à l'essai l'émission de la demande précédente

— demande non émise (RQ de IDk=1)

. émission immédiate de DL-SPEC-UPDATE confirm (IDk, substitution) de la demande précédente

. stocker la liste d'identificateurs (cette liste peut être vide)

— demande émise (RQ de IDk=0)

. définir l'indicateur de demande (RQ de IDk=1)

. stocker la liste d'identificateurs

* réception de ID_DAT (IDk) produit

* soumettre à l'essai RQ de IDk

— RQ de IDK inhibé par la configuration

émettre seulement RP_DAT

— RQ de IDK non inhibé par la configuration

émettre RP_DAT_RQ

* réception de ID_RQ (IDk)

. émettre RP_RQ avec la liste d'identificateurs demandés

. générer DL-SPEC-UPDATE confirm (IDk, OK)

. définir le RQ de IDk à 0

Les différents essais réalisés se basent sur les données concernant:

— disponibilité de la mémoire tampon

mémoire tampon non disponible si B_REQ a déjà été accédée par la liaison locale,

— attente de la demande qui sera prise en compte par l'administrateur de bus

RQ de IDk est mis à 1 par DL-SPEC-UPDATE request

RQ de IDk est mis à 0 lors de l'émission de RP_RQ.

Une demande de transfert spécifique qui contient une liste vide de variables permet d'effacer une mémoire tampon de demandes de transfert spécifiques.

7.3.4 Réception d'une DL-FREE-UPDATE

À la réception d'une primitive «request» de DL-FREE-UPDATE (LIST, PRIORITY), un producteur déclenche les actions suivantes:

* soumettre à l'essai la file d'attente

— espace suffisant non disponible,

émission immédiate de DL-FREE-UPDATE confirm (priorité, saturation)

— espace disponible et file d'attente non vide

stocker la liste d'identificateurs

— file d'attente vide

. stocker la liste d'identificateurs

. attendre le prochain ID_DAT produit

* réception de ID_DAT produit (IDp)

* soumettre à l'essai IDp

— IDp non autorisé

attendre le prochain ID_DAT produit

— IDp autorisé

. associer IDp avec la file d'attente concernée

. définir le RQ de IDp à 1

. envoyer RP_DAT_RQ