IEC 61158-5-12

Edition 3.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

colour inside

Industrial communication networks – Fieldbus specifications – Part 5-12: Application layer service definition – Type 12 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 5-12: Définition des services de la couche application – Éléments de type 12

IEC 61158-5-12:2014-08(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 It also gives information on projects, replaced and withdrawn publications.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 un 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 61158-5-12

Edition 3.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

colour inside

Industrial communication networks – Fieldbus specifications – Part 5-12: Application layer service definition – Type 12 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 5-12: Définition des services de la couche application – Éléments de type 12

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

PRICE CODE XF
CODE PRIX

ICS 25.040.40; 35.100.70; 35.110

ISBN 978-2-8322-1737-5

CONTENTS

FOREWORD 5
INTRODUCTION 7
1 Scope 8
1.1 General 8
1.2 Specifications 9
1.3 Conformance 9
2 Normative references 9
3 Terms, definitions, symbols, abbreviations and conventions 10
3.1 Reference model terms and definitions 10
3.2 Service convention terms and definitions 11
3.3 Application layer and data-link service terms and definitions 11
3.4 Common symbols and abbreviations 15
3.5 Conventions 16
4 Concepts 17
4.1 Common concepts 17
4.2 Type specific concepts 17
5 Data type ASE 26
5.1 General 26
5.2 Formal definition of data type objects 26
5.3 FAL defined data types 26
5.4 Data type ASE service specification 35
6 Communication model specification 35
6.1 ASEs 35
6.2 AR 116
Bibliography 129

Figure 1 – Producer consumer model 19
Figure 2 – Client server model 19
Figure 3 – Server triggered invocation 19
Figure 4 – Slave reference model 21
Figure 5 – Simple slave device 22
Figure 6 – Complex slave device 23
Figure 7 – Master functional overview 24
Figure 8 – Process output data sequence 36
Figure 9 – Process input data sequence 37
Figure 10 – CoE server model 55
Figure 11 – Successful single SDO-Download sequence 60
Figure 12 – Unsuccessful single SDO-Download sequence 61
Figure 13 – Successful segmented SDO-Download sequence 62
Figure 14 – Successful single SDO-Upload sequence 63
Figure 15 – Unsuccessful single SDO-Upload sequence 64
Figure 16 – Successful segmented SDO-Upload sequence 65

Figure 17 – SDO information sequence 66
Figure 18 – Emergency service 67
Figure 19 – Command sequence 68
Figure 20 – PDO mapping 70
Figure 21 – Sync manager PDO assigment 71
Figure 22 – RxPDO service 73
Figure 23 – TxPDO service 74
Figure 24 – RxPDO remote transmission sequence 75
Figure 25 – TxPDO remote transmission sequence 76
Figure 26 – EoE sequence 96
Figure 27 – FoE read sequence with success 104
Figure 28 – FoE read sequence with error 105
Figure 29 – FoE write sequence with success 106
Figure 30 – FoE write sequence with error 107
Figure 31 – FoE write sequence with busy 108
Figure 32 – Successful AL control sequence 118
Figure 33 – Unsuccessful AL control sequence 119
Figure 34 – AL state changed sequence 120

Table 1 – Process output data 39
Table 2 – Process input data 40
Table 3 – Update process input data 41
Table 4 – SII read 49
Table 5 – SII write 50
Table 6 – SII reload 51
Table 7 – Allocation of SDO areas 55
Table 8 – SDO download expedited 80
Table 9 – SDO download normal 81
Table 10 – Download SDO segment 82
Table 11 – SDO upload expedited 83
Table 12 – SDO upload normal 84
Table 13 – Upload SDO segment 85
Table 14 – Abort SDO transfer 85
Table 15 – Get OD list 86
Table 16 – OD list segment 87
Table 17 – Get object description 88
Table 18 – Get entry description 89
Table 19 – Object entry segment 91
Table 20 – Emergency 92
Table 21 – RxPDO 93
Table 22 – TxPDO 93
Table 23 – RxPDO remote transmission 94
Table 24 – TxPDO remote transmission 94

Table 25 – Initiate EoE 99
Table 26 – EoE fragment 100
Table 27 – Set IP parameter 101
Table 28 – Set address filter 102
Table 29 – FoE read 109
Table 30 – FoE write 110
Table 31 – FoE data 110
Table 32 – FoE ack 111
Table 33 – FoE busy 111
Table 34 – FoE error 112
Table 35 – MBX read 113
Table 36 – MBX write 114
Table 37 – MBX read upd 115
Table 38 – AL management and ESM service primitives 117
Table 39 – AL control 127
Table 40 – AL state change 128

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 5-12: Application layer service definition – Type 12 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 itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
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.

Attention is drawn to the fact that the use of the associated protocol type is restricted by its 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 layer protocol type to be used with other layer protocols of the same type, or in other type combinations explicitly authorized by its intellectual-property-right holders.

NOTE Combinations of protocol types are specified in IEC 61784-1 and IEC 61784-2.

International Standard IEC 61158-5-12 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This third edition cancels and replaces the second edition published in 2010. This edition constitutes a technical revision. The main changes with respect to the previous edition are listed below:

- bug fixes;
- editorial improvements;
- support of Explicit Device Identification added in ESM (see 6.2.2)

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

FDIS	Report on voting
65C/763/FDIS	65C/773/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.

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

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

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

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

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 61158-1.

The application service is provided by the application protocol making use of the services available from the data-link or other immediately lower layer. This standard defines the application service characteristics that fieldbus applications and/or system management may exploit.

Throughout the set of fieldbus standards, the term “service” refers to the abstract capability provided by one layer of the OSI Basic Reference Model to the layer immediately above. Thus, the application layer service defined in this standard is a conceptual architectural service, independent of administrative and implementation divisions.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 5-12: Application layer service definition – Type 12 elements

1 Scope

1.1 General

The fieldbus Application Layer (FAL) provides user programs with a means to access the fieldbus communication environment. In this respect, the FAL can be viewed as a "window between corresponding application programs."

This standard provides common elements for basic time-critical and non-time-critical messaging communications between application programs in an automation environment and material specific to Type 12 fieldbus. The term "time-critical" is used to represent the presence of a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This standard defines in an abstract way the externally visible service provided by the different Types of the fieldbus Application Layer in terms of

a) an abstract model for defining application resources (objects) capable of being manipulated by users via the use of the FAL service,
b) the primitive actions and events of the service;
c) the parameters associated with each primitive action and event, and the form which they take; and
d) the interrelationship between these actions and events, and their valid sequences.

The purpose of this standard is to define the services provided to

a) the FAL user at the boundary between the user and the Application Layer of the Fieldbus Reference Model, and
b) Systems Management at the boundary between the Application Layer and Systems Management of the Fieldbus Reference Model.

This standard specifies the structure and services of the IEC fieldbus Application Layer, in conformance with the OSI Basic Reference Model (ISO/IEC 7498) and the OSI Application Layer Structure (ISO/IEC 9545).

FAL services and protocols are provided by FAL application-entities (AE) contained within the application processes. The FAL AE is composed of a set of object-oriented Application Service Elements (ASEs) and a Layer Management Entity (LME) that manages the AE. The ASEs provide communication services that operate on a set of related application process object (APO) classes. One of the FAL ASEs is a management ASE that provides a common set of services for the management of the instances of FAL classes.

Although these services specify, from the perspective of applications, how request and responses are issued and delivered, they do not include a specification of what the requesting and responding applications are to do with them. That is, the behavioral aspects of the applications are not specified; only a definition of what requests and responses they can

send/receive is specified. This permits greater flexibility to the FAL users in standardizing such object behavior. In addition to these services, some supporting services are also defined in this standard to provide access to the FAL to control certain aspects of its operation.

1.2 Specifications

The principal objective of this standard is to specify the characteristics of conceptual application layer services suitable for time-critical communications, and thus supplement the OSI Basic Reference Model in guiding the development of application layer protocols for time-critical communications.

A secondary objective is to provide migration paths from previously-existing industrial communications protocols. It is this latter objective which gives rise to the diversity of services standardized as the various Types of IEC 61158, and the corresponding protocols standardized in subparts of IEC 61158-6.

This specification may be used as the basis for formal Application Programming-Interfaces. Nevertheless, it is not a formal programming interface, and any such interface will need to address implementation issues not covered by this specification, including

a) the sizes and octet ordering of various multi-octet service parameters, and

b) the correlation of paired request and confirm, or indication and response, primitives.

1.3 Conformance

This standard does not specify individual implementations or products, nor does it constrain the implementations of application layer entities within industrial automation systems.

There is no conformance of equipment to this application layer service definition standard. Instead, conformance is achieved through implementation of conforming application layer protocols that fulfill any given Type of application layer services as defined in this standard.

2 Normative references

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

NOTE All parts of the IEC 61158 series, as well as IEC 61784-1 and IEC 61784-2 are maintained simultaneously. Cross-references to these documents within the text therefore refer to the editions as dated in this list of normative references.

IEC 61131-3, *Programmable controllers – Part 3: Programming languages*

IEC 61158-1:2014, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-3-12, *Industrial communication networks – Fieldbus specifications – Part 3-12: Data-link layer service definition – Type 12 elements*

ISO/IEC 646:1991, *Information technology – ISO 7-bit coded character set for information interchange*

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 8802-3, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*

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

ISO/IEC 10646, *Information technology – Universal Coded Character Set (UCS)*

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

ISO/IEC/IEEE 60559, *Information technology – Microprocessor Systems – Floating-Point arithmetic*

IEEE 802.1D, *IEEE standard for local and metropolitan area networks – Media access control (MAC) Bridges;* available at <http://www.ieee.org>

IETF RFC 791, *Internet Protocol darpa internet program protocol specification;* available at <http://www.ietf.org>

3 Terms, definitions, symbols, abbreviations and conventions

For the purposes of this document, the following terms, definitions, symbols, abbreviations and conventions 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 [ISO/IEC 7498-1]
correspondent AL-entities (N=7)

3.1.2 (N)-entity [ISO/IEC 7498-1]
AL-entity (N=7)

3.1.3 (N)-layer [ISO/IEC 7498-1]
AL-layer (N=7)

3.1.4 layer-management [ISO/IEC 7498-1]

3.1.5 peer-entities [ISO/IEC 7498-1]

3.1.6 primitive name [ISO/IEC 7498-3]

3.1.7 AL-protocol [ISO/IEC 7498-1]

3.1.8 AL-protocol-data-unit [ISO/IEC 7498-1]

3.1.9 reset [ISO/IEC 7498-1]

3.1.10 routing [ISO/IEC 7498-1]

3.1.11 segmenting [ISO/IEC 7498-1]

3.1.12 (N)-service
AL-service (N=7) [ISO/IEC 7498-1]

3.1.13 AL-service-data-unit [ISO/IEC 7498-1]

3.1.14 AL-simplex-transmission [ISO/IEC 7498-1]

3.1.15 AL-subsystem [ISO/IEC 7498-1]

3.1.16 systems-management [ISO/IEC 7498-1]

3.1.17 AL-user-data [ISO/IEC 7498-1]

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 acceptor

3.2.2 asymmetrical service

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

3.2.4 deliver (primitive)

3.2.5 AL-service-primitive;
primitive

3.2.6 AL-service-provider

3.2.7 AL-service-user

3.2.8 indication (primitive);
acceptor.deliver (primitive)

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)

3.2.13 symmetrical service

3.3 Application layer and data-link service terms and definitions

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

3.3.1
application
function or data structure for which data is consumed or produced

3.3.2
application objects
multiple object classes that manage and provide a run time exchange of messages across the network and within the network device]

3.3.3
basic slave
slave device that supports only physical addressing of data

3.3.4
bit
unit of information consisting of a 1 or a 0

Note 1 to entry: This is the smallest data unit that can be transmitted.

3.3.5
client
1) object which uses the services of another (server) object to perform a task

2) initiator of a message to which a server reacts

3.3.6
communication object
component that manage and provide a run time exchange of messages across the network

3.3.7
connection
logical binding between two application objects within the same or different devices

3.3.8
cyclic
events which repeat in a regular and repetitive manner

3.3.9
data
generic term used to refer to any information carried over a fieldbus

3.3.10
data consistency
means for coherent transmission and access of the input- or output-data object between and within client and server

3.3.11
data type
relation between values and encoding for data of that type

Note 1 to entry: The data type definitions of IEC 61131-3 apply.

3.3.12
data type object
entry in the object dictionary indicating a data type

3.3.13
default gateway
device with at least two interfaces in two different IP subnets acting as router for a subnet.

3.3.14
device
physical entity connected to the fieldbus composed of at least one communication element (the network element) and which may have a control element and/or a final element (transducer, actuator, etc.)

3.3.15
device profile
collection of device dependent information and functionality providing consistency between similar devices of the same device

3.3.16
diagnosis information
all data available at the server for maintenance purposes

3.3.17
distributed clocks
method to synchronize slaves and maintain a global time base

3.3.18
error
discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition

3.3.19
error class
general grouping for related error definitions and corresponding error codes

3.3.20
error code
identification of a specific type of error within an error class

3.3.21
event
instance of a change of conditions

3.3.22
fieldbus memory management unit
function that establishes one or several correspondences between logical addresses and physical memory

3.3.23
fieldbus memory management unit entity
single element of the fieldbus memory management unit: one correspondence between a coherent logical address space and a coherent physical memory location

3.3.24
frame
denigrated synonym for DLPDU

3.3.25
full slave
slave device that supports both physical and logical addressing of data

3.3.26
index
address of an object within an application process

3.3.27
interface
shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics as appropriate

3.3.28
little endian
method for data representation of numbers greater 8 bit where the least significant octet is transmitted first

3.3.29
master
device that controls the data transfer on the network and initiates the media access of the slaves by sending messages and that constitutes the interface to the control system

3.3.30
mapping
correspondence between two objects in that way that one object is part of the other object

3.3.31
mapping parameters
set of values defining the correspondence between application objects and process data objects

3.3.32
medium
cable, optical fibre, or other means by which communication signals are transmitted between two or more points

Note 1 to entry: "media" is used as the plural of medium.

3.3.33
message
ordered series of octets intended to convey information

Note 1 to entry: Normally used to convey information between peers at the application layer.

3.3.34
network
set of nodes connected by some type of communication medium, including any intervening repeaters, bridges, routers and lower-layer gateways

3.3.35
node
a) single DL-entity as it appears on one local link

b) end-point of a link in a network or a point at which two or more links meet

[Derived from IEC 61158-2]

3.3.36
object
abstract representation of a particular component within a device

Note 1 to entry: An object can be

a) an abstract representation of the capabilities of a device. Objects can be composed of any or all of the following components:

   1) data (information which changes with time);

   2) configuration (parameters for behavior);

   3) methods (things that can be done using data and configuration).

b) a collection of related data (in the form of variables) and methods (procedures) for operating on that data that have clearly defined interface and behavior.

3.3.37
object dictionary
data structure addressed by Index and Sub-index that contains descriptions of data type objects, communication objects and application objects

3.3.38
process data
collection of application objects designated to be transferred cyclically or acyclically for the purpose of measurement and control

3.3.39
process data object
structure described by mapping parameters containing one or several process data entities

3.3.40
segment
collection of one real master with one or more slaves

3.3.41
server
object which provides services to another (client) object

3.3.42
service
operation or function that an object and/or object class performs upon request from another object and/or object class

3.3.43
slave
DL-entity accessing the medium only after being initiated by the preceding slave or the master

3.3.44
sub-index
subaddress of an object within the object dictionary

3.3.45
Sync Manager
collection of control elements to coordinate access to concurrently used objects.

3.3.46
Sync Manager channel
single control elements to coordinate access to concurrently used objects.

3.3.47
switch
MAC bridge as defined in IEEE 802.1D

3.4 Common symbols and abbreviations

AL-	Application layer (as a prefix)
ALE	AL-entity (the local active instance of the application layer)
AL	AL-layer
APDU	AL-protocol-data-unit
ALM	AL-management
ALME	AL-management Entity (the local active instance of AL-management)
ALMS	AL-management service
ALS	AL-service
AR	Application relationship
ASE	Application service element
CAN	Controller Area Network

CiA	CAN in Automation
CoE	CAN appliclication protocol over Type 12 services
CSMA/CD	Carrier sense multiple access with collision detection
DC	Distributed clocks
DL	Data-link-layer
DNS	Domain name system (server for name resolution in IP networks)
E²PROM	Electrically erasable programmable read only memory
EoE	Ethernet tunneled over Type 12 services
ESC	Type 12 slave controller
FCS	Frame check sequence
FIFO	First-in first-out (queuing method)
FMMU	Fieldbus memory management unit
FoE	File access with Type 12 services
HDR	Header
ID	Identifier
IETF	Internet engineering task force
IP	Internet protocol
LAN	Local area network
MAC	Medium access control
OD	Object dictionary
OSI	Open systems interconnection
PDI	Physical device internal interface (a set of elements that allows access to DL-services from the AL)
PDO	Process data object
PhL	Ph-layer
QoS	Quality of service
RAM	Random access memory
Rx	Receive
SDO	Service data object
SII	slave information interface
SM	Synchronization manager
SyncM	Synchronization manager
TCP	Transmission control protocol
Tx	Transmit
UDP	User datagram protocol
WKC	Working counter

3.5 Conventions

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

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

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

This standard uses a tabular format to describe the component parameters of the service primitives. The parameters that apply to each group of service primitives are set out in tables

throughout the remainder of this standard. Each table consists of up to five columns, containing the name of the service parameter, and a column each for those primitives and parameter-transfer directions used by the service:

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

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

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

M parameter is mandatory for the primitive.

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

C parameter is conditional upon other parameters or upon the environment of the service-user.

(blank) parameter is never present.

Some entries are further qualified by items in brackets. These may be a parameter-specific constraint:

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

In any particular interface, not all parameters need be explicitly stated. Some may be implicitly associated with the primitive.

In the diagrams which illustrate these interfaces, dashed lines indicate cause-and-effect or time-sequence relationships, and wavy lines indicate that events are roughly contemporaneous.

4 Concepts

4.1 Common concepts

All of IEC 61158-1, Clause 9 is incorporated by reference, except as specifically overwritten in 4.2.

4.2 Type specific concepts

4.2.1 Operating principle

This standard and its companion Type 12 standards describe a real-time Ethernet technology that aims to maximize the utilization of the full duplex Ethernet bandwidth. Medium access control employs the master/slave principle, where the master node (typically the control system) sends the Ethernet frames to the slave nodes, which extract data from and insert data into these frames.

From an Ethernet point of view, a Type 12 segment is a single Ethernet device, which receives and sends standard ISO/IEC 8802-3 Ethernet frames. However, this Ethernet device

is not limited to a single Ethernet controller with downstream microprocessor, but may consist of a large number of Type 12 slave devices. These process the incoming frames directly and extract the relevant user data, or insert data and transfer the frame to the next slave device. The last slave device within the segment sends the fully processed frame back, so that it is returned by the first slave device to the master as response frame.

This procedure utilizes the full duplex capability of Ethernet: both communication directions are operated independently. Communication without switch between a master device and a Type 12 segment consisting of one or several slave devices may be established.

Industrial communication systems have to meet different requirements in terms of the data transmission characteristics. Parameter data is transferred acyclically and in large quantities, whereby the timing requirements are relatively non-critical, and the transmission is usually triggered by the control system. Diagnostic data is also transferred acyclically and event-driven, but the timing requirements are more demanding, and the transmission is usually triggered by a peripheral device.

Process data, on the other hand, is typically transferred cyclically with different cycle times. The timing requirements are most stringent for process data communication. Type 12 AL supports a variety of services and protocols to meet these differing requirements.

4.2.2 Communication model overview

The Type 12 application layer distinguishes between master and slave. The communication relationship is always initiated by the master.

A Type 12 segment consists of at least one master device and one or many slave devices. All slave devices support the Type 12 State Machine (ESM) and support the transmission of Type 12 process data.

The application relationship can be modeled independent of communication relationship. The master-slave relationship is the standard application relationship.

4.2.3 Application layer element description

4.2.3.1 Management

The mandatory management consists of a set of object to control the state of a slave. An interface to DL provides read access to all DL registers.

4.2.3.2 Information interface

The mandatory slave information interface (SII) consists of all objects that can be stored persistently.

4.2.3.3 Synchronization support

The optional support of isochronous operation consists of several attributes for synchronization and timestamping of binary signals.

4.2.3.4 Access to slave

The real time entity consists of an interface for network triggered exchange of data and an interface for user triggered access to slave objects. Objects mainly used for network triggered access are called PDO. SDO are the objects that are used for user triggered access.

The access methods for PDO are read and write to a buffer. The communication structure is a producer-consumer relationship as shown in Figure 1 with no direct acknowledgement of the delivery of data. The delivery of data will be monitored by additional elements in the slave and

in the master (i.e. watchdog and working counter). The producer can be a master as well as a slave.

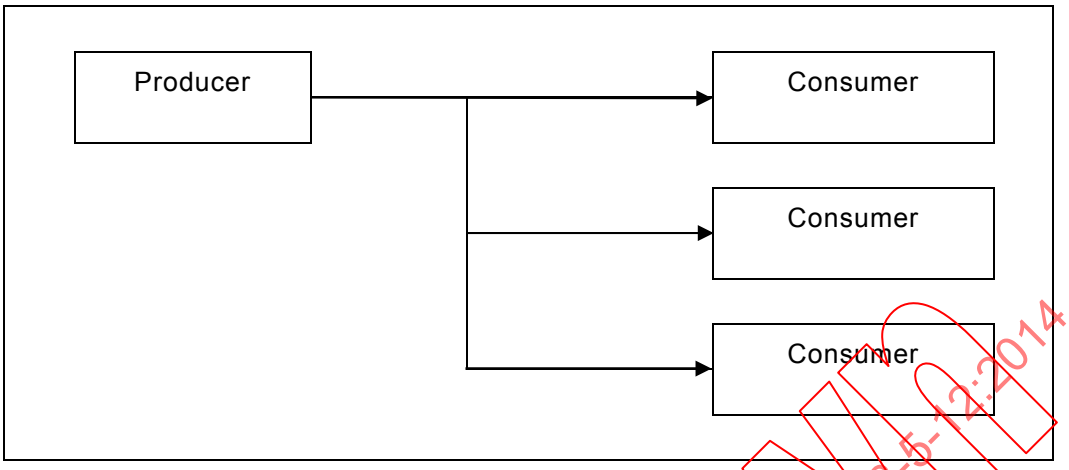


Figure 1 – Producer consumer model

The access to SDO follows the client server principle. The client issues a service invocation to the server. The slave starts the service execution and replies the result afterwards. There is always a response needed to conclude this type of service.

Figure 2 shows the workflow of this communication interaction.

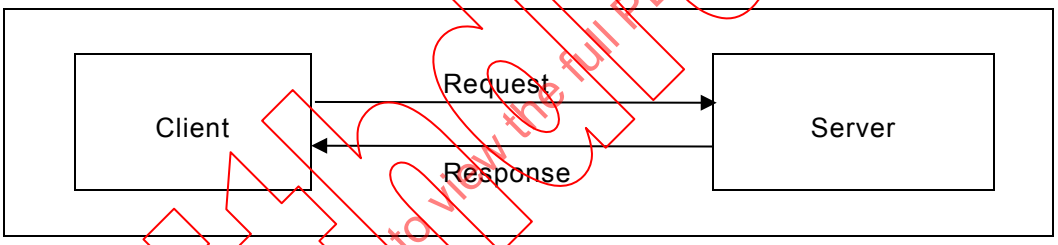


Figure 2 – Client server model

There may be an unsolicited type of server to client interaction as well. This model is used to convey data server triggered. This type of service called notification is shown in Figure 3.

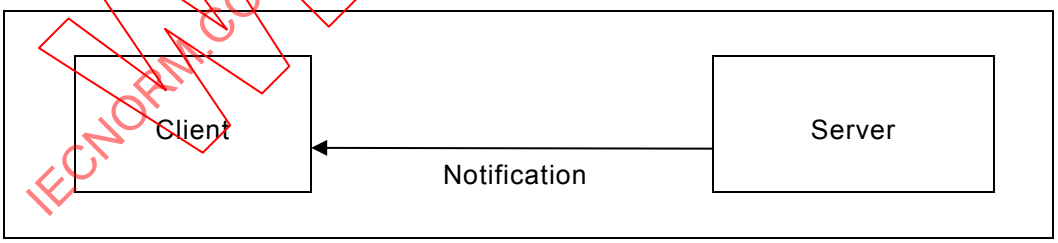


Figure 3 – Server triggered invocation

4.2.3.5 TCP/UDP/IP suite

The optional protocol is dedicated for slaves who will use the standard internet protocol suite. The protocols itself are defined in IETF. EoE describes the mapping of the IP-Protocol (and similar kind of communication) to Type 12 data-link layer. IP is a connectionless communication type with bidirectional data flow.

4.2.3.6 File access

The primary use of the file transfer is the download and upload of program files and configuration data. The file access is done with client server protocol architecture.

4.2.4 Slave reference model

4.2.4.1 Mapping onto OSI basic reference model

Type 12 is described using the principles, methodology and model of ISO/IEC 7498 Information processing systems — Open Systems Interconnection — Basic Reference Model (OSI). The OSI model provides a layered approach to communications standards, whereby the layers can be developed and modified independently. The Type 12 specification defines functionality from top to bottom of a full OSI stack, and some functions for the users of the stack. Functions of the intermediate OSI layers, layers 3 – 6, are consolidated into either the Type 12 data-link layer or the Type 12 application layer. Likewise, features common to users of the Fieldbus application layer may be provided by the Type 12 application layer to simplify user operation, as noted in

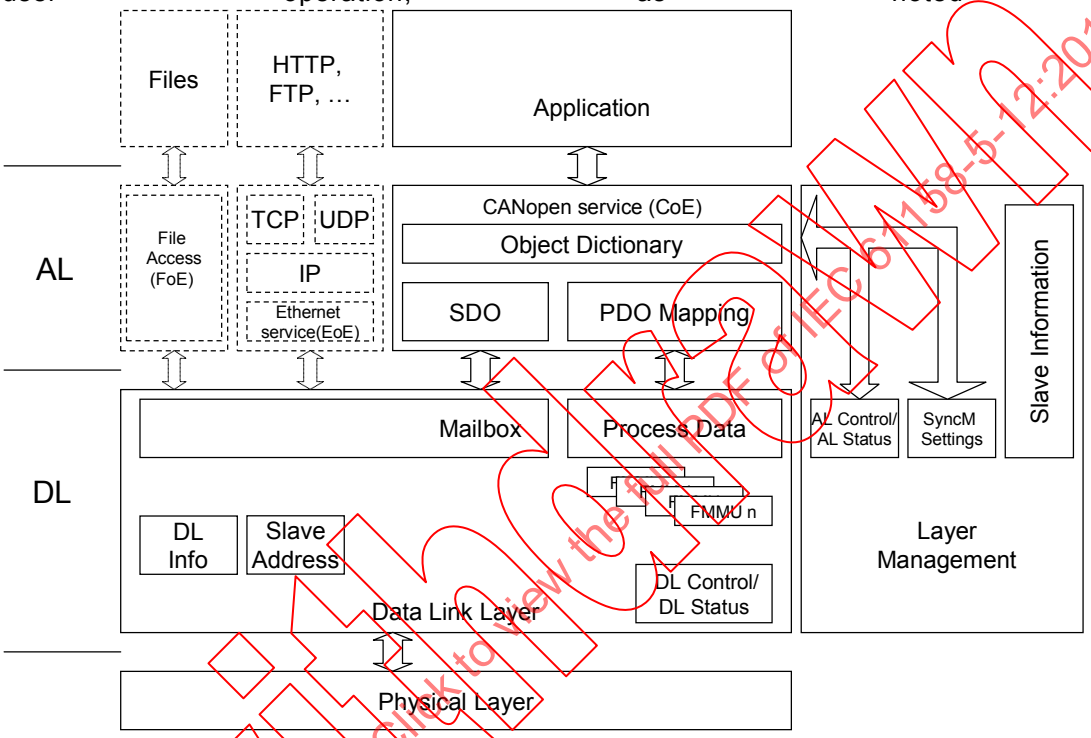


Figure 4.

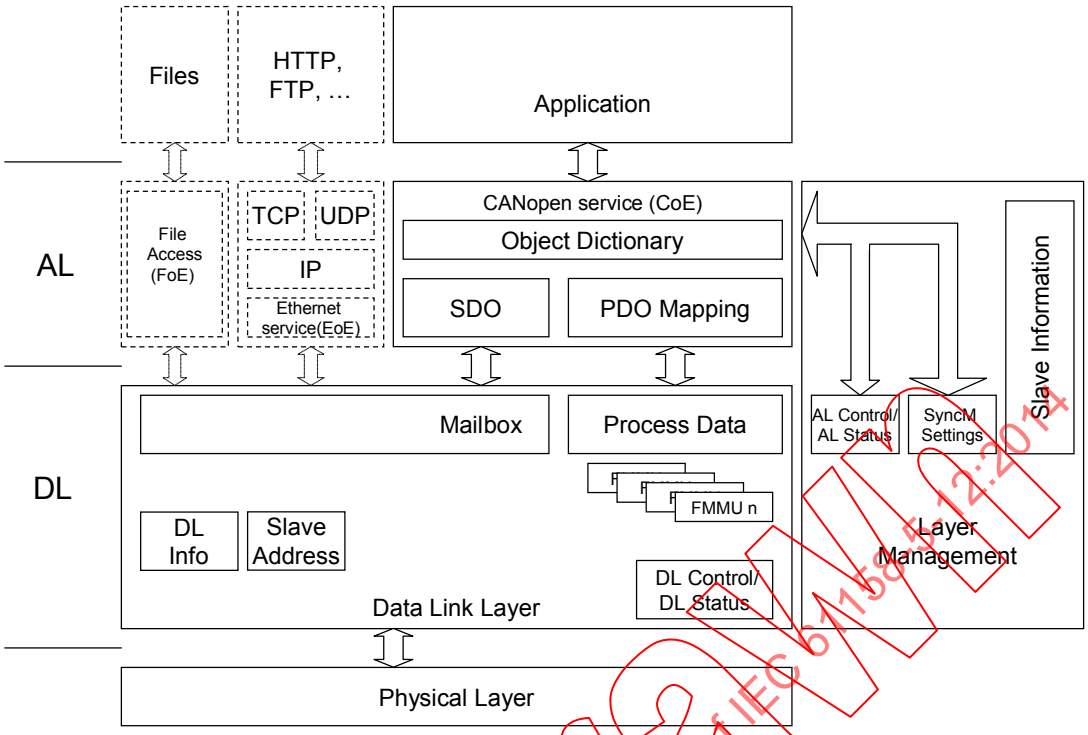


Figure 4 – Slave reference model

4.2.4.2 Data-link layer features

The data-link layer provides basic time critical support for data communications among devices connected via a Type 12 segment. The term "time-critical" is used to describe applications having a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

The data-link layer has the task to compute, compare and generate the frame check sequence and provide communications by extracting data from and/or including data into the Ethernet frame. This is done depending on the data-link layer parameters which are stored at pre-defined memory locations. The application data is made available to the application layer in physical memory, either in a mailbox configuration or within the process data section.

Additionally some data structures in the data-link layer will be used to allow a coordination of the interaction between Master and slave such as AL Control, Status and Event and Sync Manager settings.

4.2.4.3 Slave AL classification

4.2.4.3.1 Simple slave device

From the application layer point of view, slave devices are classified in simple devices without an application controller and complex devices with an application controller.

NOTE The DL slave classification in basic slaves and full slaves is independent of the AL view, since DL addressing mechanisms are invisible at the AL interface.

Simple devices have a fixed process data layout, which is described in the device description file. Simple devices may confirm the AL Management services without a reaction within the local application, as noted in Figure 5. There is no special reaction needed for safe state operation (e.g. the value 0 will be processed in the same way as no valid value will be send).

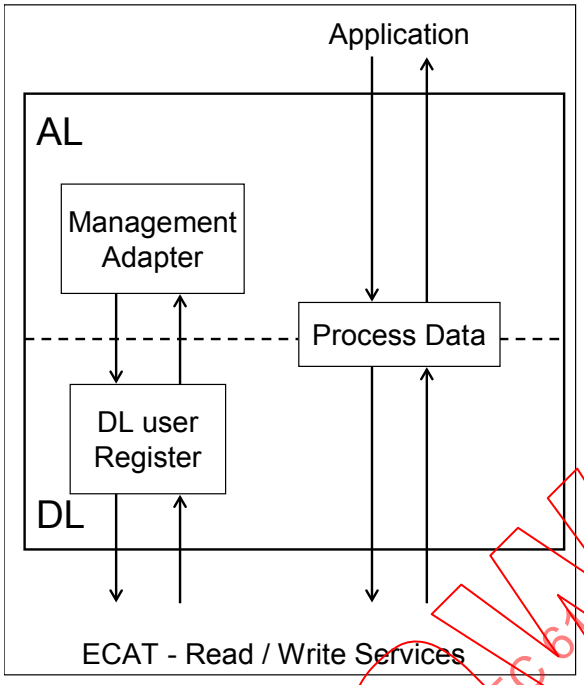


Figure 5 – Simple slave device

4.2.4.3.2 Complex slave device

As noted in Figure 6, complex devices support

- the ESM,
- the Mailbox (optional),
- the CoE object dictionary (recommended if mailbox is supported),
- the SDO services to read and/or write the object dictionary data entries (recommended if mailbox is supported), and
- the SDO information service to read the defined objects in the object dictionary and each entry description in compact format (recommended if mailbox is supported).

For the process data transmission the PDO mapping objects and the Sync Manager PDO assign objects, which describe the process data layout, shall be supported for reading. If an complex device supports configurable process data, the configuration is done by writing the PDO mapping and/or the Sync Manager PDO assign objects.

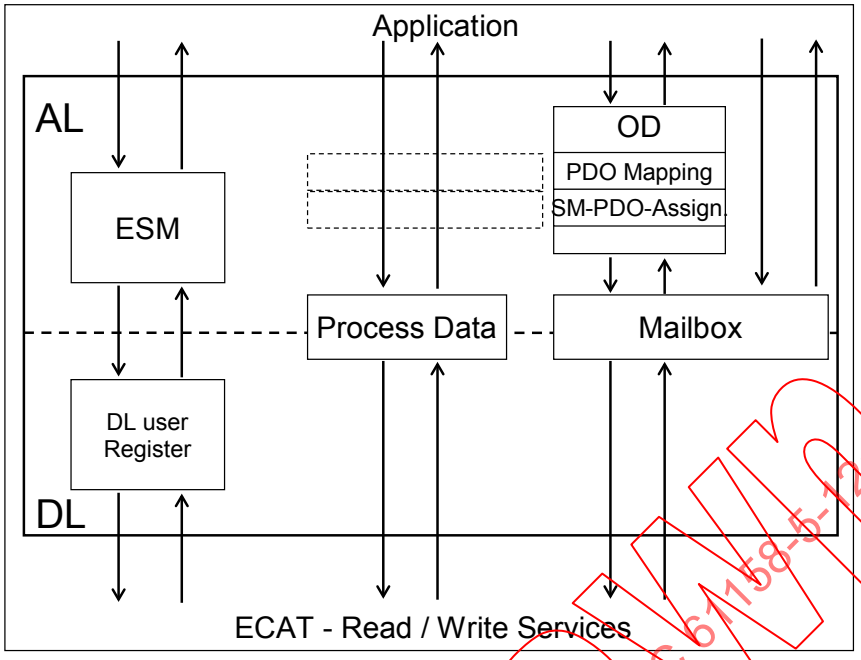


Figure 6 – Complex slave device

There are different interaction types defined in this standard:

- CAN application protocol over Type 12 services (CoE)
- Ethernet over Type 12 services (EoE)
- File Access over Type 12 services (FoE)

The different types are used to address different classes of objects. These types can be mixed at a single application relationship.

4.2.5 Master reference model

4.2.5.1 Overview

The master communicates with the slaves by using the services described in the slave chapter. Additionally there is a slave Handler for each slave defined in the master to control the ESM of the slave and a Router which enables slave-to-slave communication

via the mailbox, as noted in

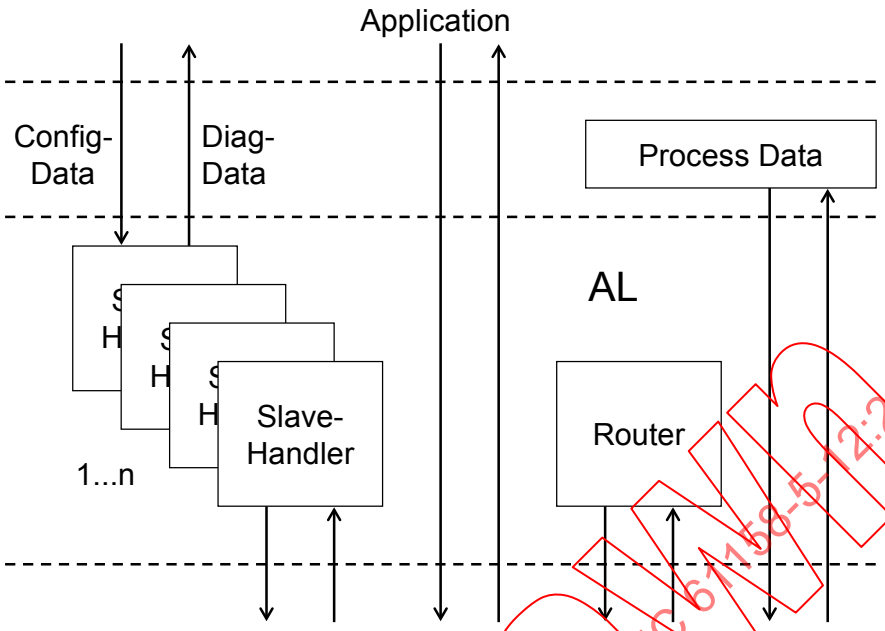


Figure 7.

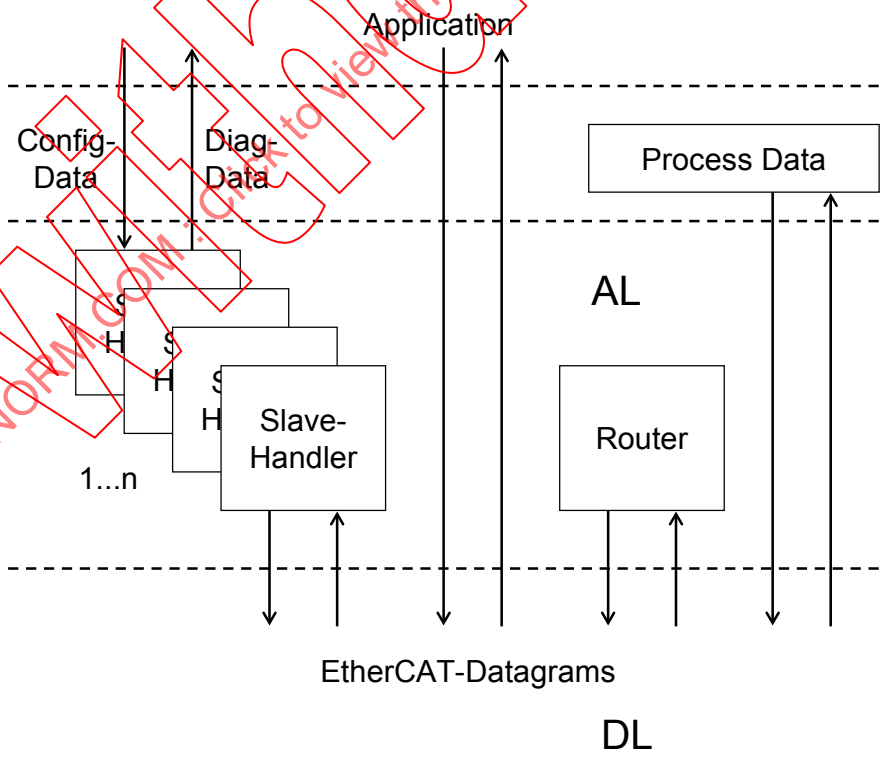


Figure 7 – Master functional overview

4.2.5.2 Slave handler

The master should support a slave handler for each slave using the state services to control the ESM of the slave. The slave Handler is the image of the slave's ESM in the master.

Additionally the slave Handler may send SDO services before changing the state of the slave's ESM

Parameter

Position

This parameter specifies the position in the logical ring which is used to address the slave when reading the Identification and writing the Station Address. It is mandatory for all slaves.

Expected Identification

This parameter specifies the expected identification of the slave, which should be read and compared by the master in the Init state. This parameter is mandatory for all slaves.

Station Address

This parameter specifies the station address which is assigned in the Init state to the slave. All further services will use this station address to address the slave. This parameter is mandatory for all slaves.

Mailbox Configuration

This parameter specifies the configuration of the Sync Manager channels 0 and 1 for the mailbox which is written in the Init state to the slave. This parameter is mandatory for complex slaves.

FMMU Configuration

This parameter specifies the configuration of the FMMU channels which is written in the Pre-Operational state to the slave.

Process Data Configuration

This parameter specifies the configuration of the Sync Manager channels which is used for process data and is written in the Pre-Operational state to the slave.

PDO mapping

This parameter specifies the PDO mapping objects which may be written in the Pre-Operational state to the slave.

Sync Manager PDO assign

This parameter specifies the Sync Manager PDO assign objects which may be written in the Pre-Operational state to the slave.

Start Up Objects

This parameter specifies the objects from the object dictionary of the slave which may be written to the slave from the slave Handler during start up.

4.2.5.3 Router

The Router can be used for several applications:

- routing mailbox services from the client slave to the server slave
- routing mailbox service responses from the server slave to the client slave
- forwarding mailbox services from third party devices
- forwarding mailbox service responses to third party devices

The task of the router is to overwrite the address field of the mailbox service with the station address of the client or with a virtual address before routing the mailbox service to the server addressed by the original address field. The address field of the mailbox service response is overwritten by the router with the station address of the server before routing the mailbox service response to the client slave addressed by the original address field or to the corresponding IP Address/ MAC Address in case of a virtual address.

5 Data type ASE

5.1 General

All of IEC 61158-1, 5.1 is incorporated by reference.

5.2 Formal definition of data type objects

All of IEC 61158-1, 5.2 is incorporated by reference.

5.3 FAL defined data types

5.3.1 Fixed length types

5.3.1.1 Boolean types

CLASS: **Data type**
ATTRIBUTES:

1	Data type Numeric Identifier	=	1
2	Data type Name	=	Boolean
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

This data type expresses a Boolean data type with the values TRUE and FALSE.

5.3.1.2 Bitstring types

5.3.1.2.1 BIT2

CLASS: **Data type**
ATTRIBUTES:

1	Data type Numeric Identifier	=	31
2	Data type Name	=	BIT2
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT2 is an ordered sequence of Boolean data types, numbered from 1 to 2.

5.3.1.2.2 BIT3

CLASS: **Data type**
ATTRIBUTES:

1	Data type Numeric Identifier	=	32
2	Data type Name	=	BIT3
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT3 is an ordered sequence of Boolean data types, numbered from 1 to 3.

5.3.1.2.3 BIT4

CLASS: **Data type**
ATTRIBUTES:

1	Data type Numeric Identifier	=	33
2	Data type Name	=	BIT3
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT4 is an ordered sequence of Boolean data types, numbered from 1 to 4.

5.3.1.2.4 BIT5

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	34
2	Data type Name	=	BIT5
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT5 is an ordered sequence of Boolean data types, numbered from 1 to 5.

5.3.1.2.5 BIT6

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	35
2	Data type Name	=	BIT6
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT6 is an ordered sequence of Boolean data types, numbered from 1 to 6.

5.3.1.2.6 BIT7

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	36
2	Data type Name	=	BIT7
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT7 is an ordered sequence of Boolean data types, numbered from 1 to 7.

5.3.1.2.7 BIT8

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	37
2	Data type Name	=	BIT8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BIT8 is an ordered sequence of Boolean data types, numbered from 1 to 8.

5.3.1.2.8 BITARR8

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	45
2	Data type Name	=	BITARR8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

A BITARR8 is an ordered sequence of bits, numbered from 1 to 8.

5.3.1.2.9 BITARR16

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	46

2	Data type Name	=	BITARR16
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	2

A BITARR16 is an ordered sequence of bits, numbered from 1 to 15.

5.3.1.2.10 BITARR32

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	47
2	Data type Name	=	BITARR32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

A BITARR32 is an ordered sequence of bits, numbered from 1 to 31.

5.3.1.3 Currency types

There are no Currency types defined for Type 12.

5.3.1.4 Date/Time types

5.3.1.4.1 TimeOfDay

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	12
2	Data type Name	=	TimeOfDay
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	6

This data type is composed of two elements of unsigned values and expresses the time of day and the date. The first element is an Unsigned32 data type and gives the time after the midnight in milliseconds. The second element is an Unsigned16 data type and gives the date counting the days from January 1, 1984.

5.3.1.4.2 TimeDifference

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	13
2	Data type Name	=	TimeDifference
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4 or 6

This data type is composed of two elements of unsigned values that express the difference in time. The first element is an Unsigned32 data type that provides the fractional portion of one day in milliseconds. The optional second element is an Unsigned16 data type that provides the difference in days.

5.3.1.5 Enumerated types

There are no Enumerated types defined for Type 12.

5.3.1.6 Handle types

There are no Handle types defined for Type 12.

5.3.1.7 Numeric types

5.3.1.7.1.1 float

This data type is the same as Float32.

5.3.1.7.1.2 Float32

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	8
2	Data type Name	=	Float32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

This type has a length of four octets. The format for float32 is that defined by ISO/IEC/IEEE 60559 as single precision.

5.3.1.7.1.3 double

This data type is the same as Float64.

5.3.1.7.1.4 Float64

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	17
2	Data type Name	=	Float64
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	8

This type has a length of eight octets. The format for float64 is that defined by ISO/IEC/IEEE 60559 as double precision.

5.3.1.7.2 Integer types

5.3.1.7.2.1 Integer8

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	2
2	Data type Name	=	Integer8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

This integer type is a two's complement binary number with a length of one octet.

5.3.1.7.2.2 SINT

This IEC 61131-3 type is the same as Integer8.

5.3.1.7.2.3 char

This data type is the same as Integer8.

5.3.1.7.2.4 Integer16

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	3

2	Data type Name	=	Integer16
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	2

This integer type is a two's complement binary number with a length of two octets.

5.3.1.7.2.5 INT

This IEC 61131-3 type is the same as Integer16.

5.3.1.7.2.6 short

This data type is the same as Integer16.

5.3.1.7.2.7 Integer24

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	16
2	Data type Name	=	Integer24
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	3

This integer type is a two's complement binary number with a length of three octets.

5.3.1.7.2.8 Integer32

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	4
2	Data type Name	=	Integer32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

This integer type is a two's complement binary number with a length of four octets.

5.3.1.7.2.9 DINT

This IEC 61131-3 type is the same as Integer32.

5.3.1.7.2.10 long

This data type is the same as Integer32.

5.3.1.7.2.11 Integer40

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	18
2	Data type Name	=	Integer40
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	5

This integer type is a two's complement binary number with a length of five octets.

5.3.1.7.2.12 Integer48

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	19
2	Data type Name	=	Integer48
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	6

This integer type is a two's complement binary number with a length of six octets.

5.3.1.7.2.13 Integer56

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	20
2	Data type Name	=	Integer56
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	7

This integer type is a two's complement binary number with a length of seven octets.

5.3.1.7.2.14 Integer64

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	21
2	Data type Name	=	Integer64
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	8

This integer type is a two's complement binary number with a length of eight octets.

5.3.1.7.2.15 LINT

This IEC 61131-3 type is the same as Integer64.

5.3.1.7.3 Unsigned types

5.3.1.7.3.1 Unsigned8

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	5
2	Data type Name	=	Unsigned8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This type has a length of one octet.

5.3.1.7.3.2 USINT

This IEC 61131-3 type is the same as Unsigned8.

5.3.1.7.3.3 unsigned char

This data type is the same as Unsigned8.

5.3.1.7.3.4 BYTE

This data type is the same as USINT.

5.3.1.7.3.5 Unsigned16

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	6
2	Data type Name	=	Unsigned16
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	2

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of two octets.

5.3.1.7.3.6 UINT

This IEC 61131-3 type is the same as Unsigned16.

5.3.1.7.3.7 WORD

This type is used in the same way as UINT.

5.3.1.7.3.8 Unsigned24

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	22
2	Data type Name	=	Unsigned24
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	3

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of three octets.

5.3.1.7.3.9 Unsigned32

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	7
2	Data type Name	=	Unsigned32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of four octets.

5.3.1.7.3.10 UDINT

This IEC 61131-3 type is the same as Unsigned32.

5.3.1.7.3.11 Unsigned40

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	24
2	Data type Name	=	Unsigned40
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	5

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of five octets.

5.3.1.7.3.12 Unsigned48

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	25
2	Data type Name	=	Unsigned48
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	6

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of six octets.

5.3.1.7.3.13 Unsigned56

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	26
2	Data type Name	=	Unsigned56
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	7

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of seven octets.

5.3.1.7.3.14 Unsigned64

CLASS: **Data type**

ATTRIBUTES:

1	Data type Numeric Identifier	=	27
2	Data type Name	=	Unsigned64
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	8

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of eight octets.

5.3.1.7.3.15 ULINT

This IEC 61131-3 type is the same as Unsigned64.

5.3.1.8 Pointer types

There are no Pointer types defined for Type 12.

5.3.1.9 OctetString types

There are no OctetString types of fixed length defined for Type 12.

5.3.1.10 VisibleString character types

There are no VisibleString types of fixed length defined for Type 12.

5.3.2 String types

5.3.2.1 OctetString

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	10
2	Data type Name	=	OctetString
3	Format	=	STRING
4.1	Octet Length	=	1 to n

An OctetString is an ordered sequence of octets, numbered from 1 to n. For the purposes of discussion, octet 1 of the sequence is referred to as the first octet.

NOTE IEC 61158-6-12 defines the order of transmission.

5.3.2.2 VisibleString

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	9
2	Data type Name	=	VisibleString
3	Format	=	STRING
4.1	Octet Length	=	1 to n

This type is defined as the ISO/IEC 646 string type.

5.3.2.3 UnicodeString

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	11
2	Data type Name	=	UnicodeString
3	Format	=	STRING
4.1	Octet Length	=	1 to n

This type is defined as the ISO/IEC 10646 string type.

5.3.3 GUID Types

5.3.3.1 GUID

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	29
2	Data type Name	=	GUID
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	16

A GUID is a globally unique identifier with a length of 128 Bit.

5.4 Data type ASE service specification

All of IEC 61158-1, 5.4 is incorporated by reference.

6 Communication model specification

6.1 ASEs

6.1.1 Process data ASE

6.1.1.1 Overview

In the Type 12 application layer environment, each application process of slave can contain several objects for each application process instance to convey process data. It is structured by means of PDOs. The contents of the process data can be described by the PDO Mapping and the Sync Manager PDO assign objects of the CoE ASE. For simple slave devices the process data is fixed and is defined in the device description file.

For the process data communication usually the application memory of the buffered type is used that master and slave always have access to the process data.

Additional services are provided to read acyclically the values of the process data objects and to indicate new values for the input data object and the output data object.

The process data objects are implicitly addressed through the related services. The granularity of input or output data in a server/provider is according to the correspondent configuration attributes.

The process data ASE uses the producer/consumer access model. That means that the update of the process data with the values of the inputs and the update of the outputs with the process data are decoupled from the conveyance of the data. The receipt of a new value is indicated by the Process Output Data indication service primitive.

The primitives of the Process Output Data services are mapped to the buffered type application memory primitives described in the DL. It is recommended but not required to use FMMU entities. Configured with FMMU a single Process Output Data request can result in multiple Process Output Data indications. The process data confirmation will show the master the success or failure of the update procedure.

Figure 8 shows the primitives between master and the slaves for a Process Output Data sequence.

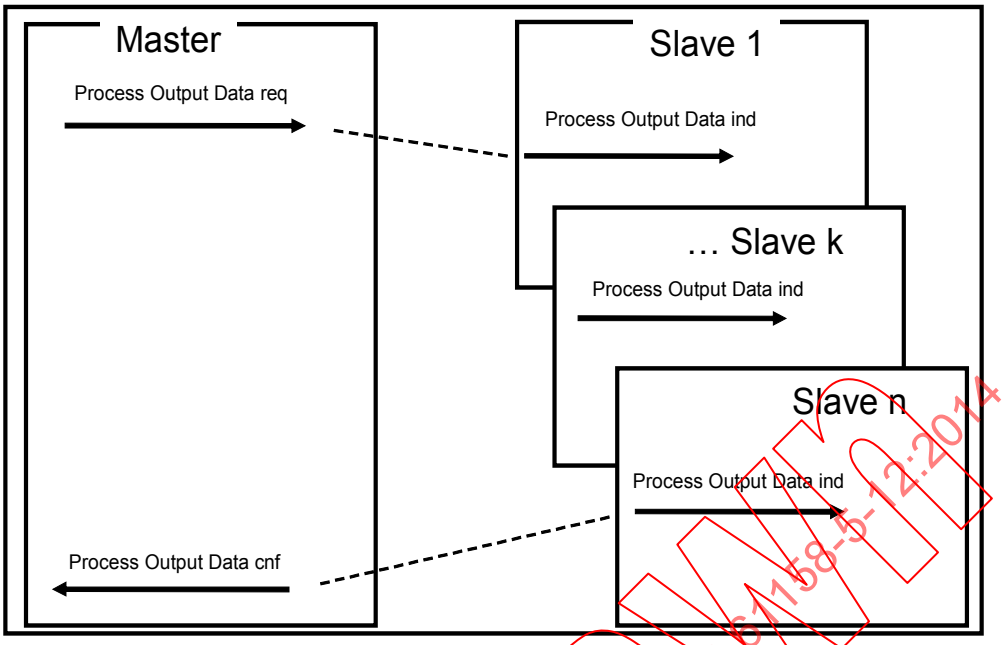


Figure 8 – Process output data sequence

The master usually sends process output data to several slaves issuing a DL write or RW service. Each slave gets the AL Event of the corresponding Sync Manager. The slave's AL controller may read the process output data at any time from the related application memory.

The primitives of the Process Input Data services are mapped to the buffered type application memory primitives described in the DL.

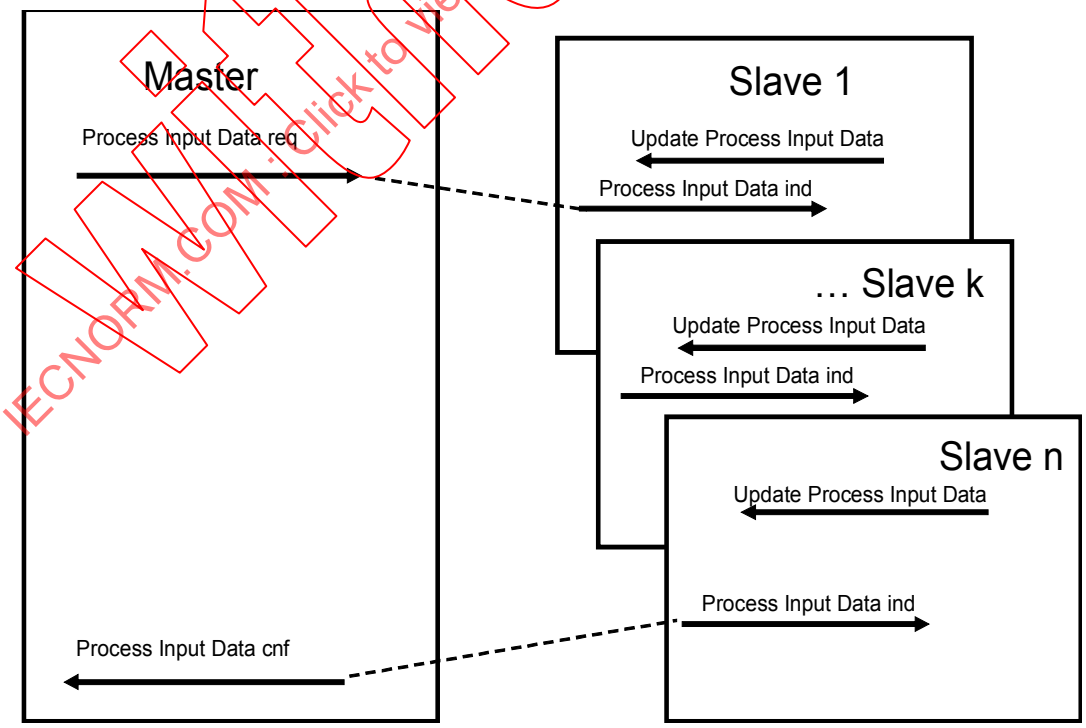


Figure 9 shows the primitives between master and the slaves for a Process Input Data sequence.

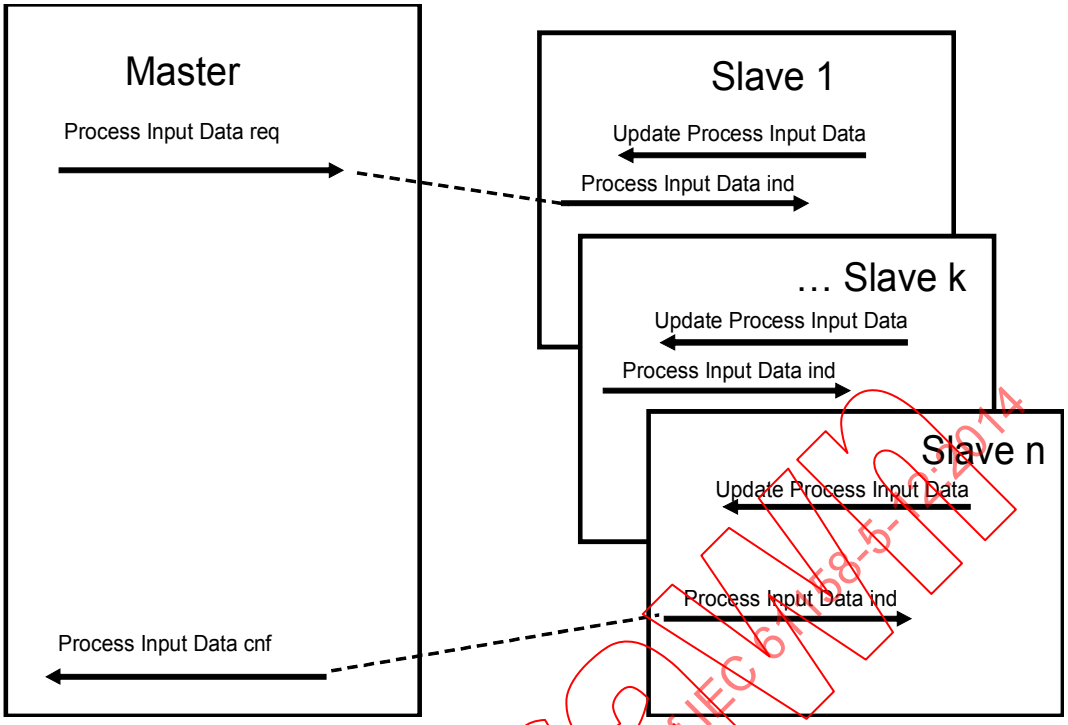


Figure 9 – Process input data sequence

The master usually reads process input data from several slaves with a logical read or RW service. The master gets the data from the previously written buffer. Each slave gets the AL Event of the corresponding Sync Manager if the input data are read out. The slave's AL controller may write the process input data at any time in the related application memory.

The formal model of the process data ASE is described next, followed by a description of its services. Furthermore, the process data ASE represents the real input and output structure of a device.

For all service primitives, parameter memory areas that are written by concurrently active service primitives should not overlap.

6.1.1.2 Process data class specification

6.1.1.2.1 Formal model

The process data object is described by the following template:

ASE:			**Process Data ASE**
CLASS:			**Process Data**
CLASS ID:			not used
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Implicit
2.	(m)	Attribute:	List of Buffer
2.1	(m)	Attribute:	BufferType
2.2	(m)	Attribute:	List of PDO
2.2.1	(m)	Attribute:	PDO Index
2.2.2	(m)	Attribute:	List of Entry
2.2.2.1	(m)	Attribute:	Index
2.2.2.2	(m)	Attribute:	Subindex
2.2.2.3	(m)	Attribute:	Bitlen
SERVICES:			
1.	(o)	OpsService:	Process Output Data
2.	(o)	OpsService:	Process Input Data
3.	(o)	OpsService:	Update Process Input Data

Objects for structuring and additional access can be found in CoE ASE.

6.1.1.2.2 Attributes

Implicit

The attribute Implicit indicates that the process data object is implicitly addressed by the services.

List of buffer
One Buffer is composed of the following list elements:

Buffertype
This attribute specifies the type of the buffer.

Allowed values: Input or Output.

List of PDO
One Buffer Element is composed of the following list elements:

PDO index
This attribute specifies to the index of the the process data object. Its permissible range is 0x1600 to 0x17FF and 0x1A00 to 0x1BFF.

List of entry
This attribute consists of the following attributes.

Index
This attribute specifies to the reference to the object. Its permissible range is 1 to 0x7FFF.

Subindex
This attribute specifies to the reference to the object entry. Its permissible range is 0 to 0xFF.

Bitlen
This attribute specifies to the length in bits of the object entry value. Its permissible range is 1 to 255.

6.1.1.2.3 Services

Process output data
This optional service is used by masters to convey output data to the slave.

Process input data
This optional service is used by slaves to publish input data.

Update process Input data
This optional service is used by slaves to provide input data to publish.

6.1.1.3 Process data ASE service specification

6.1.1.3.1 Supported services

The process data ASE defines the services

Process Output Data

Process Input Data

Update Process Input Data

6.1.1.3.2 Process output data service

6.1.1.3.2.1 Service overview

The Process Output Data service is used to convey data from the master to the slave.

6.1.1.3.2.2 Service primitives

Table 1 shows the parameters of the service.

Table 1 – Process output data

Parameter name	Req	Ind	Cnf
Argument			
List of Slave Address	M	M (=)	
List of RxPDO	M	M (=)	
Output Data	M	M (=)	
Result(+)			S
Result(–)			S
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.			

Argument
The argument conveys the service specific parameters of the service request.

List of slave address
This parameter specifies the identifier for the slaves.

List of RxPDO
This parameter specifies a predefined fixed set of RxPDOs.

Output data
This parameter specifies the value of the Output Data object elements.

The allocation of the entries is done in that way, that the entries will be placed in sequential order. If the bit length is less than 8 and the number of bits fits in the unused bit of the actual octet, the object will be mapped onto the actual octet with the first bit positioned in the first unused bit.

NOTE The structuring of the output data is not fixed and can be changed with the use CoE services and FMMUs.

Result(+)
This selection type parameter indicates that the service request succeeded.
Result(–)
This selection type parameter indicates that the service request failed.

6.1.1.3.2.3 Service procedure

According to the cyclic buffer to buffer transportation characteristics the following behavior is possible:

- The Process Output Data requests are issued faster than the contents of the buffers are transported over the network. In this case the values provided with the latest Process Output Data service are conveyed over the network.
- The Process Output Data requests are issued slower than the contents of the buffers are transported over the network. In this case the values provided with each Process Output Data service are conveyed more than once over the network.
- If the Process Output Data requests are issued synchronous with the transport of the contents of the buffers the values provided with each Process Output Data service are conveyed once over the network.

6.1.1.3.3 Process input data service

6.1.1.3.3.1 Service overview

The Process Input Data service is used to convey data from the slave to the master.

6.1.1.3.3.2 Service primitives

Table 2 shows the parameters of the service.

Table 2 – Process input data

Parameter name	Req	Ind	Cnf
Argument			
List of Slave Address	M	M (=)	
List of TxPDO	M	M (=)	
Result(+)			S
Input Data			M
Result(–)			S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

List of slave address
This parameter specifies the identifier for the slaves.

List of TxPDO
This parameter specifies a predefined fixed set of TxPDOs.

Result(+)
This selection type parameter indicates that the service request succeeded.

Input data
This parameter specifies the value of the Input Data object elements.

The allocation of the entries is done in that way, that the entries will be placed in sequential order. If the bit length is less than 8 and the number of bits fits in the unused bit of the actual octet, the object will be mapped onto the actual octet with the first bit positioned in the first unused bit.

NOTE The structuring of the input data is not fixed and can be changed with the use CoE services and FMMUs.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.1.3.3.3 Service procedure

According to the cyclic buffer to buffer transportation characteristics the following behavior is possible.

- The Process Input Data requests issued faster than the contents of the buffers are transported over the network. In this case only the values provided with the latest Update Process Input Data service are conveyed over the network.
- The Process Input Data requests issued slower than the contents of the buffers are transported over the network. In this case the values provided with each Update Process Input Data service are conveyed more than once over the network.
- If the Process Input Data requests are issued synchronous with the transport of the contents of the buffers the values provided with each Update Process Input Data service are conveyed once over the network.

6.1.1.3.4 Update process input data service

6.1.1.3.4.1 Service overview

The Update Process Input Data service is used to update input data buffer in the slave.

6.1.1.3.4.2 Service primitives

Table 3 shows the parameters of the service.

Table 3 – Update process input data

Parameter name	Req	Cnf
Argument		
List of TxPDO	M	
Input Data	M	
Result(+)		S
Result(–)		S
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.		

Argument
The argument conveys the service specific parameters of the service request.

List of TxPDO
This parameter specifies a predefined fixed set of TxPDOs.

Input data
This parameter specifies the value of the Input Data object elements.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.1.3.4.3 Service procedure

– The service Procedure is described with the Process Input Data service.

6.1.2 SII ASE

6.1.2.1 Overview

In the Type 12 application layer environment, each application process of a slave has a SII which contains all information needed for identification of a device. The slave information interface is stored persistently and specifies the boot configuration data and the application information data. The boot configuration data contain the settings to initialize the interface of the slave controller during power on. The application information data contain the product code that can be used by the master to find the corresponding configuration file for the device. The slave information interface registers define the access to the slave information interface which is supported by the slave controller (ESC).

The formal model of the SII ASE is described next, followed by a description of its services. Furthermore, the SII ASE represents the real input and output structure of a device.

6.1.2.2 SII class specification

6.1.2.2.1 Formal model

The SII object is described by the following template:

ASE:			**SII ASE**
CLASS:			**SII**
CLASS ID:			not used
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Implicit
2.	(m)	Attribute:	PDI Control
2.1	(m)	Attribute:	PDI Type
2.2	(m)	Attribute:	AL State Control
2.3	(o)	Attribute:	Specific Settings
2.4	(o)	Attribute:	Sync Impulse Length
2.4	(o)	Attribute:	Configured station address
3.	(m)	Attribute:	Device Identification
3.1	(m)	Attribute:	Vendor ID
3.2	(m)	Attribute:	Product Code
3.3	(m)	Attribute:	Revision Number
3.4	(o)	Attribute:	Serial Number
4.	(o)	Attribute:	Mailbox Configuration
4.1	(o)	Attribute:	Bootstrap
4.1.1	(o)	Attribute:	Receive Memory Offset
4.1.2	(o)	Attribute:	Receive Size
4.1.3	(o)	Attribute:	Send Memory Offset
4.1.4	(o)	Attribute:	Send Size

4.2	(o)	Attribute:	Standard
4.2.1	(o)	Attribute:	Receive Memory Offset
4.2.2	(o)	Attribute:	Receive Size
4.2.3	(o)	Attribute:	Send Memory Offset
4.2.4	(o)	Attribute:	Send Size
4.2.5	(o)	Attribute:	Mailbox Protocols
5.	(m)	Attribute:	SII information
5.1	(m)	Attribute:	Size
5.2	(o)	Attribute:	Version
6.	(o)	Attribute:	Additional Parameter
6.1	(o)	Attribute:	Delay Correcion 1
6.1.1	(o)	Attribute:	Port0 Delay
6.1.1	(o)	Attribute:	Port1 Delay
6.2	(o)	Attribute:	Delay Correcion 2
6.2.1	(o)	Attribute:	Port0 Tx Delay
6.2.2	(o)	Attribute:	Port1 Tx Delay
6.2.3	(o)	Attribute:	Port2 Tx Delay
6.2.4	(o)	Attribute:	Port3 Tx Delay
6.2.5	(o)	Attribute:	Port0 Rx Delay
6.2.6	(o)	Attribute:	Port1 Rx Delay
6.2.7	(o)	Attribute:	Port2 Rx Delay
6.2.8	(o)	Attribute:	Port3 Rx Delay
6.3	(o)	Attribute:	Delay Correcion 3
6.3.1	(o)	Attribute:	Forward Port 0 to next Port
6.3.2	(o)	Attribute:	Forward non-Port 0 to next Port
6.3.3	(o)	Attribute:	Additive forward time closed Port
7.	(o)	Attribute:	Categories
7.1	(o)	Attribute:	List of Strings
7.1.1	(o)	Attribute:	Strings
7.2	(o)	Attribute:	General information
7.2.1	(o)	Attribute:	Group Name
7.2.2	(o)	Attribute:	Image
7.2.3	(o)	Attribute:	Order information
7.2.4	(o)	Attribute:	Name
7.2.5	(o)	Attribute:	Physical Layer
7.2.6	(o)	Attribute:	CoE Details
7.2.7	(o)	Attribute:	FoE Details
7.2.8	(o)	Attribute:	EoE Details
7.2.9	(o)	Attribute:	NoLWR
7.2.10	(o)	Attribute:	Power Budget
7.3	(o)	Attribute:	List of FMMU
7.3.1	(o)	Attribute:	Entry
7.4	(o)	Attribute:	List of Sync Manager
7.4.1	(o)	Attribute:	Sync Manager
7.5	(o)	Attribute:	List of TxPDO
7.5.1	(o)	Attribute:	Index
7.5.2	(o)	Attribute:	Related Sync Manager
7.5.3	(o)	Attribute:	Reference to DC
7.5.4	(o)	Attribute:	Name
7.5.5	(o)	Attribute:	List of Entries
7.5.5.1	(o)	Attribute:	Index
7.5.5.2	(o)	Attribute:	Subindex
7.5.5.3	(o)	Attribute:	Name

7.5.5.4	(o)	Attribute:	Data type
7.5.5.5	(o)	Attribute:	Data type
7.6	(o)	Attribute:	List of RxPDO
7.6.1	(o)	Attribute:	Index
7.6.2	(o)	Attribute:	Related Sync Manager
7.6.3	(o)	Attribute:	Reference to DC
7.6.4	(o)	Attribute:	Name
7.6.5	(o)	Attribute:	List of Entries
7.6.5.1	(o)	Attribute:	Index
7.6.5.2	(o)	Attribute:	Subindex
7.6.5.3	(o)	Attribute:	Name
7.6.5.4	(o)	Attribute:	Data type
7.6.5.5	(o)	Attribute:	Data Length
7.7	(o)	Attribute:	DC Settings
7.7.1	(o)	Attribute:	DC area

SERVICES:

1.	(o)	OpsService:	Read SII
2.	(o)	OpsService:	Write SII
3.	(o)	OpsService:	Reload SII

Objects for structuring and additional access can be found in CoE ASE.

6.1.2.2.2 Attributes

Implicit

The attribute Implicit indicates that the SII object is implicitly addressed by the services.

PDI control
This object is composed of the following elements:

PDI type
This attribute, defined in 6.2.2, specifies the initialization value for the PDI Type of AR ASE.

AL state control
This attribute, defined in 6.2.2, specifies the initialization value for the AL State Control of AR ASE.

Specific settings
This optional attribute, defined in 6.2.2, specifies the initialization value for the Specific Settings of AR ASE.

Sync impulse length
This optional attribute specifies the value for the Sync Impulse.

Configured station address
This optional attribute specifies the initialization value for the configured station address, as specified in IEC 61158-3-12.

Device identification
This object is composed of the following elements:

Vendor ID
This attribute specifies the initialization Vendor ID as detailed in 6.1.4.1.2.3.6. Its length is four octets.

Product code
This attribute specifies the initialization Product Code as detailed in 6.1.4.1.2.3.6. Its length is four octets.

Revision number
This attribute specifies the initialization Revision Number as detailed in 6.1.4.1.2.3.6. Its length is four octets.

Serial number
This attribute specifies the initialization Serial Number as detailed in 6.1.4.1.2.3.6. Its length is four octets.

Mailbox configuration
This object is composed of the following elements:

Bootstrap
This optional object is composed of the following elements:

Receive memory offset
This attribute specifies the offset to the memory area. Its permissible range is 0x1000 to 0xFFDA.

Receive size
This attribute specifies the receive mailbox size. Its permissible range is 38 to 1486 octets.

Send memory offset
This attribute specifies the offset to the memory area. Its permissible range is 0x1000 to 0xFFDA.

Send size
This attribute specifies the receive mailbox size. Its permissible range is 38 to 1486 octets.

Standard
This optional object is composed of the following elements:

Receive memory offset
This attribute specifies the offset to the memory area. Its permissible range is 0x1000 to 0xFFDA.

Receive size
This attribute specifies the receive mailbox size. Its permissible range is 38 to 1486 octets.

Send memory offset
This attribute specifies the offset to the memory area. Its permissible range is 0x1000 to 0xFFDA.

Send size
This attribute specifies the receive mailbox size. Its permissible range is 38 to 1486 octets.

Mailbox protocols
This attribute specifies the mailbox protocols supported. Its allowed values are EoE, CoE, FoE, and profile and application specific encodings.

SII information
This object is composed of the following elements:

Size
This attribute specifies the size of the SII structure in units of 1 Kibit (1 024 bits). Its range is 1 to 65 536 units of 1 Kibit each, reperesented as 0 to 0xFFFF.

Version
This attribute specifies the SII structure version; its value is 1

Additional parameter
This object is composed of the following elements:

Delay correction 1
This attribute consists of the following elements, each with a granularity of 100 ps.

Port0 delay
This attribute specifies a correction factor for line delay to be added if master is behind Port 0.

Port1 delay
This attribute specifies a correction factor for line delay to be added if master is behind Port 1.

Delay correction 2
This attribute consists of the following elements, each with a granularity of 100 ps.

Port0 Tx delay
This attribute specifies a correction factor for delay to be added if Port0 as sender is involved in a path. It represents the time difference between signal at the PhL media and the entry at PhL.

Port1 Tx delay
This attribute specifies a correction factor for delay to be added if Port1 as sender is involved in a path. It represents the time difference between signal at the PhL media and the entry at PhL.

Port2 Tx delay
This attribute specifies a correction factor for delay to be added if Port2 as sender is involved in a path. It represents the time difference between signal at the PhL media and the entry at PhL.

Port3 Tx delay
This attribute specifies a correction factor for delay to be added if Port3 as sender is involved in a path. It represents the time difference between signal at the PhL media and the entry at PhL.

Port0 Rx delay
This attribute specifies a correction factor for delay to be added if Port0 as receiver is involved in a path. It represents the time difference between the entry at PhL and signal at the PhL media.

Port1 Rx delay
This attribute specifies a correction factor for delay to be added if Port1 as receiver is involved in a path. It represents the time difference between the entry at PhL and signal at the PhL media.

Port2 Rx delay
This attribute specifies a correction factor for delay to be added if Port2 as receiver is involved in a path. It represents the time difference between the entry at PhL and signal at the PhL media.

Port3 Rx delay
This attribute specifies a correction factor for delay to be added if Port3 as receiver is involved in a path. It represents the time difference between the entry at PhL and signal at the PhL media.

Delay correction 3
This attribute consists of the following elements, each with a granularity of 100 ps.

Forward port 0 to next port
This attribute specifies a correction factor for delay to be added if Port0 as receiver is involved in a path. It represents the time difference between PhL to DL at Port 0 and that of DL to PhL at the next port.

Forward non-port 0 to next port
This attribute specifies a correction factor for delay to be added if a port other than Port0 as receiver is involved in a path. It represents the time difference between PhL to DL at the port that is not Port 0 and that of DL to PhL at the next port.

Additive forward time closed port
This attribute specifies a correction factor for delay to be added if a port is in the closed state. The basic time is “Additive forward time closed Port” if Port 0 is involved as virtual receiver.

Categories
This object is composed of the following elements:

List of strings
This attribute consists of the following list elements:

Strings
This attribute specifies textual elements that are reference by their list position in the following category elements.

General information
This attribute consists of the following elements:

Group name
This attribute specifies a reference to a string in the Strings collection. It is used to specify a group name for the device.

Image
This attribute specifies a reference to a string in the Strings collection. It is used to specify a name of a pictorial representation for the device.

Order information
This attribute specifies a reference to a string in the Strings collection. It is used to specify the order information for the device.

Name
This attribute specifies a reference to a string in the Strings collection. It is used to specify the name information for the device.

Physical layer
This attribute specifies the main physical layer technology of the device. Its permissible values are "not used", E-Bus, 100BASE-TX, 100BASE-FX

CoE details
This attribute specifies a structure with CoE information.

FoE details
This attribute specifies a structure with FoE information.

NoLRW
This Boolean attribute is true if use of the DL-service LogicalReadWrite is not supported by this device.

Power budget
This attribute specifies the operating current required by this device. Negative values indicate that the device provides power. This attribute's range is -128 to +127 mA.

List of FMMU
This attribute consists of the following list elements:

Entry
This attribute specifies the preferred use of an FMMU. A maximum of three uses can be supported.

List of Sync Manager
This attribute consists of the following list elements:

Sync Manager
This attribute specifies the preferred use of a Sync Manager, as defined in IEC 61158-3-12.

List of TxPDO
This optional attribute is present if there are a limited number of PDO selections that are always available for this type of device. It consists of the following list of elements:

Index
This attribute specifies the Index of a PDO. Its permissible range is 0x1A00 to 0x1BFF.

Related Sync Manager
This attribute specifies the allocation of the PDU to a Sync Manager. Its permissible range is 0 to 15.

Reference to DC sync
This attribute specifies one of up to sixteen Sync Methods as specified in the CoE object dictionary.

Name
This attribute specifies the Name as Reference to a string

List of entries
This attribute consists of the following list elements:

Index
This attribute specifies the Index of an object.

Subindex
This attribute specifies the Subindex of an object.

Name
This attribute specifies the Name as Reference to a string

Data type
This attribute specifies the data type as index to a CoE object directory.

Data length
This attribute specifies the length of the object in bits.

List of RxPDO
This optional attribute is present if there are a limited number of PDO selections that are always available for this type of device. It consists of the following list of elements:

Index
This attribute specifies the Index of a PDO in the range 0x1600 to 0x17FF

Related Sync Manager
This attribute specifies the allocation to a Sync Manager; its range is 0 to 15

Reference to DC sync
This attribute specifies the Sync Method as specified in CoE object dictionary; its range is 0 to 15

Name
This attribute specifies the Name as Reference to a string

List of entries
This attribute consists of the following list elements:

Index
This attribute specifies the Index of an object.

Subindex
This attribute specifies the Subindex of an object.

Name
This attribute specifies the Name as Reference to a string

Data type
This attribute specifies the data type as index to a CoE object directory.

Data length
This attribute specifies the length of the object in bits.

DC settings
This attribute consists of the following element:

DC area
This attribute specifies an octet string the contents will be defined in the device profiles.

6.1.2.2.3 Services

Write SII
This optional service is used by masters to convey 2 octets of data to the slave SII.

Read SII
This optional service is used by masters to convey 4 octets of data from the slave SII to the master.

Reload SII
This optional service is used by masters to convey 2 octets of data to the slave SII and update the mirror register in the Slave DL.

6.1.2.3 SII ASE service specification

6.1.2.3.1 Supported services

The SII ASE defines the services

Read SII

Write SII

Reload SII

6.1.2.3.2 SII read service

6.1.2.3.2.1 Service overview

The SII Read service is used to transfer of up to 4 octets of data from the server to client. The server answers with the result of the read operation.

6.1.2.3.2.2 Service primitives

Table 4 shows the service primitives and parameter of the SII Read service.

Table 4 – SII read

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Word Index	M	M (=)		
Result(+)			S	S
Data			M	M (=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the slave.

Word Index
This parameter specifies the word index to the SII area which is to be read.

Result(+)
This selection type parameter indicates that the service request succeeded.

Data
This parameter specifies the object value read.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.2.3.3 SII write service

6.1.2.3.3.1 Service overview

The SII Write service is used to transfer of up to 2 octets of data from the client to server. The server answers with the result of the write operation.

6.1.2.3.3.2 Service primitives

Table 5 shows the service primitives and parameter of the SII Write service.

Table 5 – SII write

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Word Index	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the slave.

Word Index
This parameter specifies the word index to the SII area which is to be written.

Data
This parameter specifies the object value which is to be written.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.2.3.4 SII reload service

6.1.2.3.4.1 Service overview

The SII Reload service is used to transfer of up to 2 octets of data from the client to server and refresh the DL registers associated with the new SII values. The server answers with the result of the reload operation.

6.1.2.3.4.2 Service primitives

Table 6 shows the service primitives and parameter of the SII Reload service.

Table 6 – SII reload

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Word Index	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the slave.

Word index
This parameter specifies the word index to the SII area which is to be written.

Data
This parameter specifies the object value which is to be written.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.3 Isochronous ASE

6.1.3.1 Overview

In the Type 12 application layer environment, an application process of a slave may have a isochronous PDI which specifies information needed for synchronous operation of several devices. The isochronous specification assumes an interface that consists of two synchronization signals that are related to the system clock, as defined in IEC 61158-3 and IEC 61158-4. Additionally support for latching input signals may be supported.

The isochronous PDI registers define the access to the slave information interface if isochronous operation is supported by the slave controller (ESC).

The formal model of the isochronous ASE is described next, the services are local read and write services of the DL and local events.

6.1.3.2 Isochronous class specification

6.1.3.2.1 Formal model

The isochronous object is described by the following template:

ASE:			**Isochronous ASE**
CLASS:			**Isochronous**
CLASS ID:			not used
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Implicit
2.	(o)	Attribute:	Sync Parameter
2.1	(o)	Attribute:	Cyclic operation enable
2.2	(o)	Attribute:	SYNC0 activate
2.3	(o)	Attribute:	SYNC1 activate
2.4	(o)	Attribute:	Sync Impulse Length
2.5	(o)	Attribute:	Interrupt 0 Status
2.6	(o)	Attribute:	Interrupt 1 Status
2.7	(o)	Attribute:	Cyclic Operation Start Time
2.8	(o)	Attribute:	SYNC0 cycle time
2.9	(o)	Attribute:	SYNC1 cycle time
3.	(m)	Attribute:	Latch Parameter
3.1	(m)	Attribute:	Latch 0 positive edge
3.2	(m)	Attribute:	Latch 0 negative edge
3.3	(m)	Attribute:	Latch 1 positive edge
3.4	(o)	Attribute:	Latch 1 negative edge
3.1	(m)	Attribute:	Latch 0 positive event
3.2	(m)	Attribute:	Latch 0 negative event
3.3	(m)	Attribute:	Latch 1 positive event
3.4	(o)	Attribute:	Latch 1 negative event
3.1	(m)	Attribute:	Latch 0 positive edge value
3.2	(m)	Attribute:	Latch 0 negative edge value
3.3	(m)	Attribute:	Latch 1 positive edge value
3.4	(o)	Attribute:	Latch 1 negative edge value
SERVICES:			
1.	(o)	OpsService:	DL-read local
2.	(o)	OpsService:	DL-write local
1.	(o)	OpsServices:	DL-read
2.	(o)	OpsServices:	DL-write

Service description will be found in IEC 61158-3-12.

6.1.3.2.2 Attributes

Implicit

The attribute Implicit indicates that the isochronous object is implicitly addressed by the services.

Sync parameter
This object is composed of the following elements:

Cyclic operation enable
This Boolean attribute enables cyclic Sync 0 or 1 operation.

Sync0 activate
This Boolean attribute enables the Sync 0 operation.

Sync1 activate
This Boolean attribute enables the Sync 1 operation.

Sync Impulse Length
This attribute specifies the value of Sync Impulse Length as specified in SII ASE.

Specific Settings
This optional attribute, defined in 6.2.2, specifies the initialization value for the Specific Settings of AR ASE.

SyncImpulseLen
This optional attribute specifies the value for the Sync Impulse in multiples of 10 ns.

Interrupt 0 Status
This optional Boolean attribute indicates an active Sync0 interrupt.

Interrupt 1 Status
This optional Boolean attribute indicates an active Sync1 interrupt.

Cyclic Operation Start Time
This optional attribute sets a start time related to system time for cyclic operation.

Sync0 cycle time
This attribute set the cycle time of Sync0 in multiples of 1ns for Sync0.

Sync1 cycle time
This attribute set the cycle time of Sync1 in multiples of 1ns for Sync0.

Latch Parameter
This object is composed of the following elements:

Latch 0 positive edge
This Boolean attribute enables Latch 0 operation for a single event (true) or continuous latching.

Latch 0 negative edge
This Boolean attribute enables Latch 0 operation for a single event (true) or continuous latching.

Latch 1 positive edge
This Boolean attribute enables Latch 1 operation for a single event (true) or continuous latching.

Latch 1 negative edge
This Boolean attribute enables Latch 1 operation for a single event (true) or continuous latching.

Latch 0 positive event
This Boolean attribute indicates Latch 0 positive edge event.

Latch 0 negative event
This Boolean attribute indicates Latch 0 negative edge event.

Latch 1 positive event
This Boolean attribute indicates Latch 1 positive edge event.

Latch 1 negative event
This Boolean attribute indicates Latch 1 negative edge event.

Latch 0 positive edge value
This attribute stores the system time value in case of a Latch 0 positive edge event.

Latch 0 negative edge value
This attribute stores the system time value in case of a Latch 0 negative edge event.

Latch 0 positive edge value
This attribute stores the system time value in case of a Latch 0 positive edge event.

Latch 0 negative edge value
This attribute stores the system time value in case of a Latch 0 negative edge event.

6.1.3.2.3 Services

DL-read local
This optional service is used by the slave application to read out the isochronous parameter. The value of the parameter memory address should be in the range of 0x980 to 0x9FF.

DL-write local
This optional service is used by the slave application to write isochronous parameter. The value of the parameter memory address should be in the range of 0x980 to 0x9FF. Only a subset of the registers can be written.

DL-read
This optional service is used by the master to read out the isochronous parameter. The value of the parameter memory address should be in the range of 0x980 to 0x9FF.

DL-write
This optional service is used by the master to write isochronous parameter. The value of the parameter memory address should be in the range of 0x980 to 0x9FF. Only a subset of the registers can be written.

6.1.4 CoE ASE

6.1.4.1 Overview

6.1.4.1.1 General

CANopen is a communication protocol originally developed for CAN-based systems. It is a standardized embedded network with highly flexible configuration capabilities.

NOTE CANopen (CiA DS 301) is standardized as EN 50325-4.

Type 12 uses the CAN application protocol service definition.

The Object Dictionary contains parameters, application data and the mapping information between process data interface and application date (PDO mapping). Its entries can be accessed via Service Data Objects (SDO), as shown in

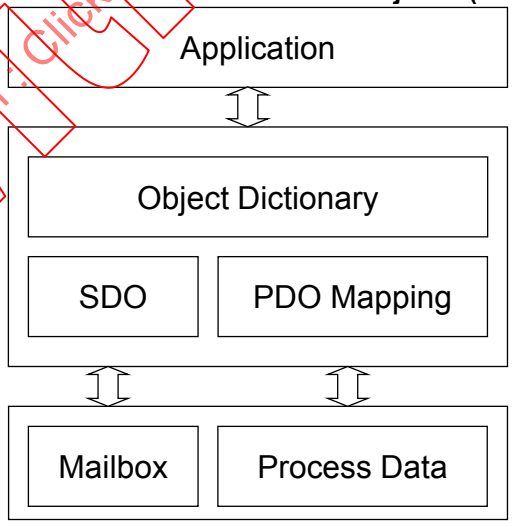


Figure 10.

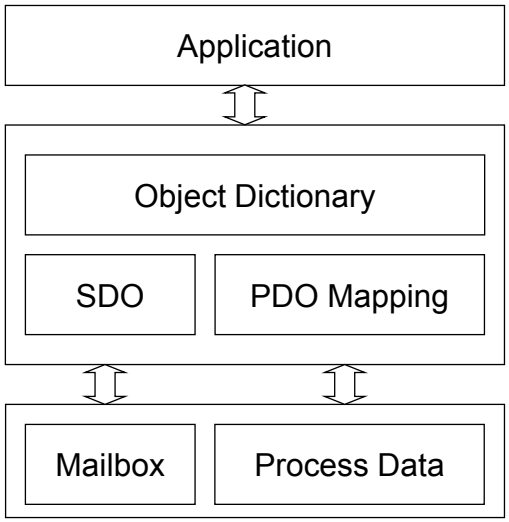


Figure 10 – CoE server model

6.1.4.1.2 Object dictionary

6.1.4.1.2.1 Object dictionary structure

The Object Dictionary contains all the CoE related data objects of the device in a standardized way. It is a collection of the device parameters data structures that can be accessed with the SDO Upload and SDO Download services.

With the SDO information services the available entries of the object dictionary and the entry description of an entry in the object dictionary can be read.

The object dictionary consists of the areas as described in Table 7

Table 7 – Allocation of SDO areas

Area	Contents
Data type Area:	Definition of the data types
CoE Communication Area:	Definition of the variables which may be used for all servers and for dedicated communication purposes
Manufacturer Specific Area:	Definition of manufacturer specific variables
Device Profile Area:	Definition of the variables defined in a device profile
Reserved Area:	reserved for future use

6.1.4.1.2.2 Data type area

The Data type Area consists of the following parts:

Static Data types
    Definition of general simple data types
Complex Data types
    Definition of general structured data types
Manufacturer Specific Complex Data types
    Definition of manufacturer specific structured data types
Device Profile Specific Static Data types
    Definition of device profile specific simple data types
Device Profile Specific Complex Data types
    Definition of device profile specific structured data types
Enumeration Data types
    Definition of device specific enumeration data types

6.1.4.1.2.3 CoE Communication area

6.1.4.1.2.3.1 Device type

The Device Type object has the following parameter:

Parameter
    Device profile
        This parameter specifies the device profile that is used of the device.
    Additional information
        This parameter specifies additional information defined by the device profile.

6.1.4.1.2.3.2 Error register

The optional Error Register object has the following parameter:

Parameter
    Generic error
        This parameter is set to true if a generic error is present.
    Current error
        This parameter is set to true if a current error is present.
    Voltage error
        This parameter is set to true if a voltage error is present.
    Temperature error
        This parameter is set to true if a temperature error is present.
    Communication error
        This parameter is set to true if a communication error is present.
    Device profile specific error
        This parameter is set to true if a device profile specific error is present.
    Manufacturer specific error
        This parameter is set to true if a manufacturer specific error is present.

6.1.4.1.2.3.3 Manufacturer device name

The Manufacturer Device Name object has the following parameter:

Parameter

Device name

This parameter specifies the device name of the device.

6.1.4.1.2.3.4 Hardware version

The Hardware Version object has the following parameter:

Parameter

Hardware version

This parameter specifies the hardware version of the device.

6.1.4.1.2.3.5 Software version

The Software Version object has the following parameter:

Parameter

Software version

This parameter specifies the software version of the device.

6.1.4.1.2.3.6 Identity

The Identity object has the following parameter:

Parameter

Vendor ID

This parameter specifies the vendor ID of the device.

Product code

This parameter specifies the product code of the device.

Major Revision Number

This parameter specifies the major revision number of the device which identifies the functionality of the device.

Minor Revision Number

This parameter specifies the minor revision number of the device which identifies different version with the same functionality.

Serial Number

This parameter specifies the serial number of the device.

6.1.4.1.2.3.7 Receive PDO mapping

The Receive PDO Mapping object has the following parameter:

Parameter

PDO Number

This parameter specifies the number of the PDO.

Number of mapping entries

This parameter specifies the number of mapping entries.

List of mapping entries

This parameter specifies a list of mapping entries.

6.1.4.1.2.3.8 Transmit PDO mapping

The Transmit PDO Mapping object has the following parameter:

Parameter

PDO Number

This parameter specifies the number of the PDO.

Number of mapping entries

This parameter specifies the number of mapping entries.

List of mapping entries

This parameter specifies a list of mapping entries.

6.1.4.1.2.3.9 Sync manager communication Type

The Sync Manager Communication Type object has the following parameter:

Parameter

Number of Used Sync Manager entities

This parameter specifies the number of used Sync Manager entities.

List of Sync Manager Communication Types

This parameter specifies the communication type for each used Sync Manager entity.

6.1.4.1.2.3.10 Sync manager PDO assignment

The Sync Manager PDO assignment object has the following parameter:

Parameter

Channel number

This parameter specifies the number of the Sync Manager channel.

Number of assigned PDOs

This parameter specifies the number of assigned PDOs.

List of assigned PDOs

This parameter specifies the list of assigned PDOs.

6.1.4.1.2.3.11 Synchronization timing

The Sync Manager Synchronization object has the following parameter:

Parameter

Channel number

This parameter specifies the number of the Sync Manager channel.

Synchronization type

This parameter specifies the method of synchronization.

Cycle time

This parameter specifies the target length of a cycle.

Shift time

This parameter specifies an offset parameter for synchronized action.

6.1.4.1.3 SDO interactions

6.1.4.1.3.1 SDO services

With the SDO services entries of the Object Dictionary can be read or written. The SDO transport protocol allows transmitting objects of any size. The SDO protocol is equivalent to the CANopen SDO protocol in order to support reuse of existing protocol stacks.

The first octet of the first segment specifies the necessary flow control information. The next three octets of the first segment specify the index and sub-index of the Object Dictionary entry to be read or written. The next octets of the first segment are available for user data. The second and the following segments contain the control octet and user data. The receiver confirms each segment or a block of segments, so that a peer-to-peer communication (client/server) takes place.

In legacy mode the SDO protocol consists of 8 octets only to match the CAN data size. In enhanced mode the payload data is simply enlarged without changing the protocol headers. In this way the larger data size of the mailbox is applied to the SDO protocol, the transmission of large data is accelerated accordingly.

Furthermore, a mode is added that allows one to transfer the entire data located at one index of the object dictionary at one go. The data of all sub-indices is then transferred subsequently.

The confirmed services (SDO Upload, SDO Download, Download SDO Segment, and Upload SDO Segment) and the unconfirmed service (Abort SDO Transfer) are used for Service Data Objects performing the segmented/expedited transfer. The SDO Download service class is split up in SDO Download Expedited and Initiate SDO Download Normal services. The SDO Upload service class is split up in SDO Upload Expedited and Initiate SDO Upload Normal services.

The primitives of the SDO services are mapped to the primitives of the mailbox transmission services.

6.1.4.1.3.2 SDO download sequence

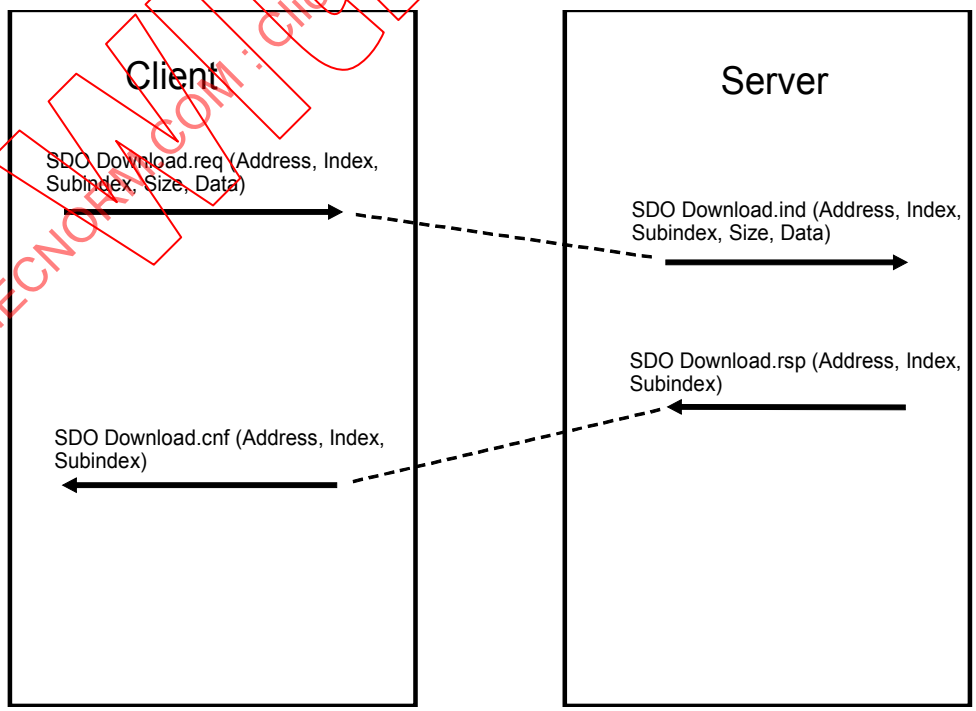


Figure 11 shows the primitives between client and server in case of a successful single SDO Download sequence.

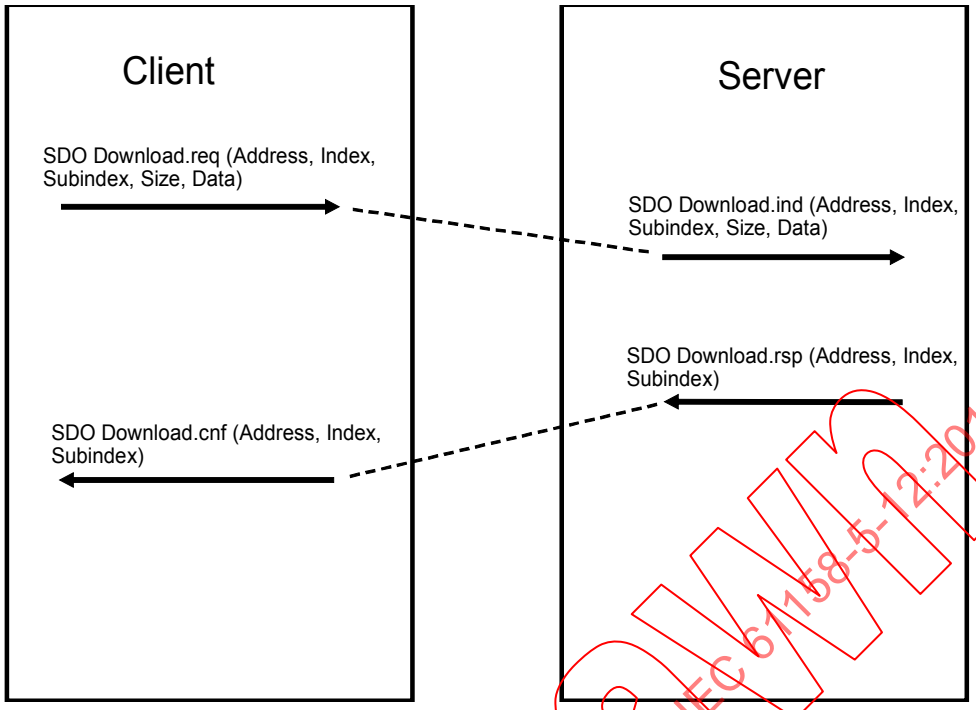


Figure 11 – Successful single SDO-Download sequence

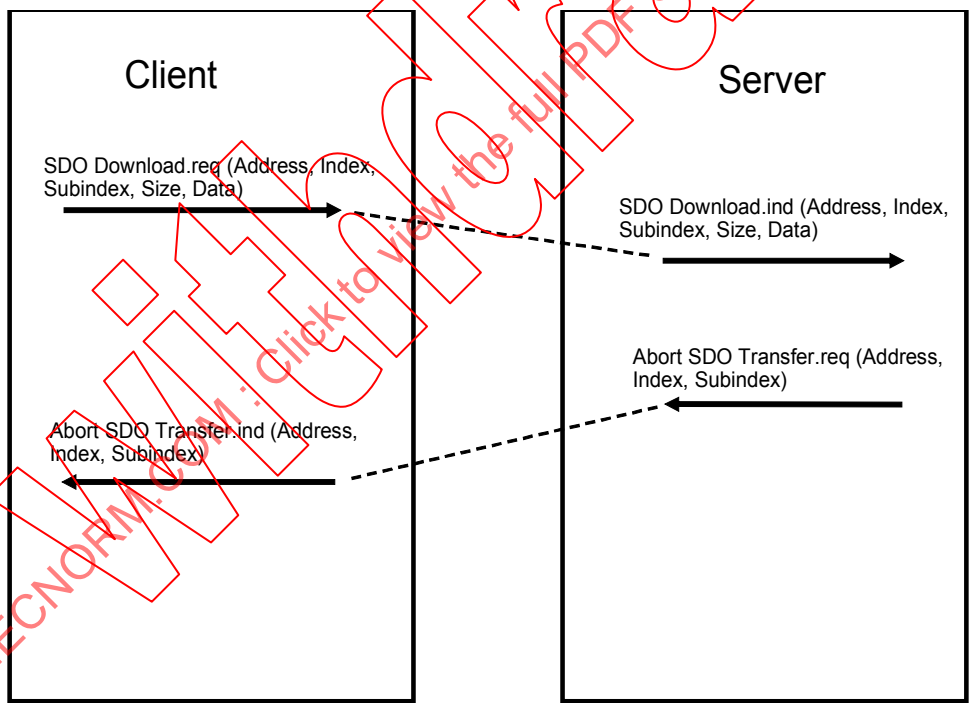


Figure 12 shows the primitives between client and server in case of an unsuccessful SDO Download sequence. The service parameter Address, Index and Subindex should be the same in the Download request and the Abort request.

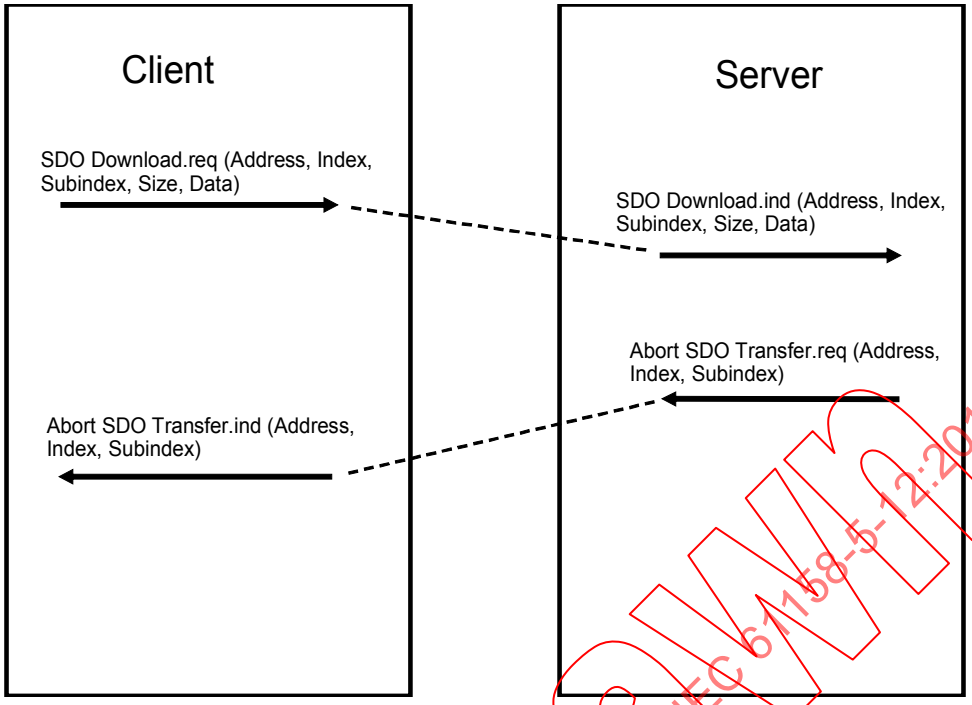


Figure 12 – Unsuccessful single SDO Download sequence

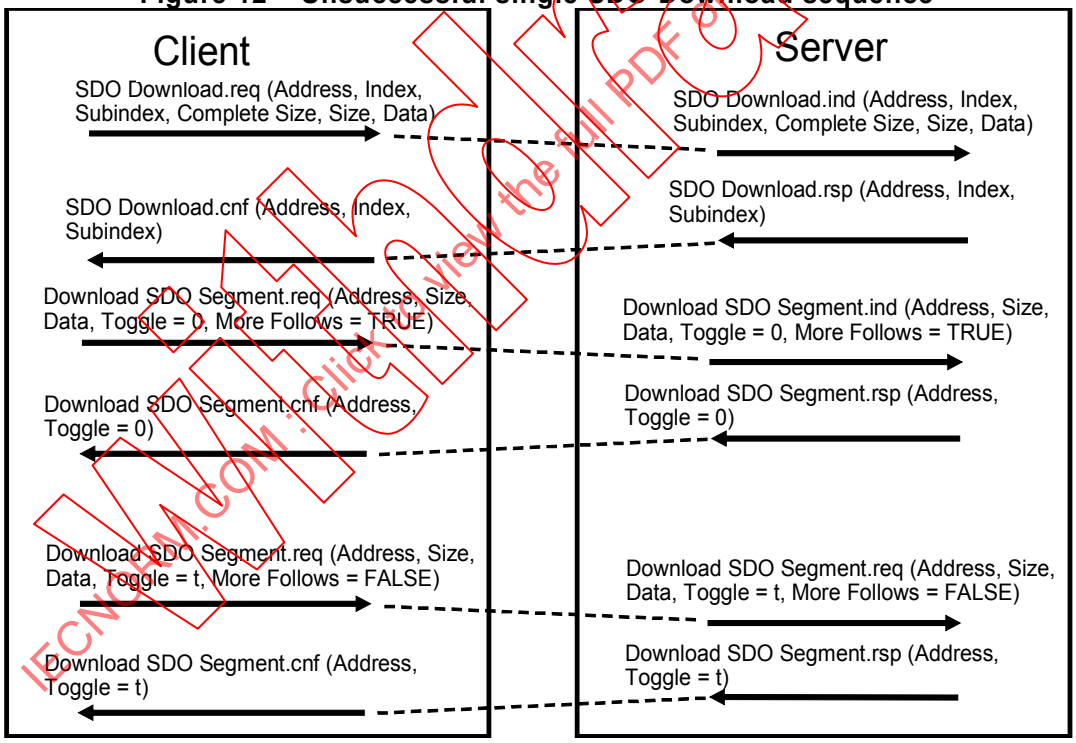


Figure 13 shows the primitives between client and server in case of a successful segmented SDO Download sequence.

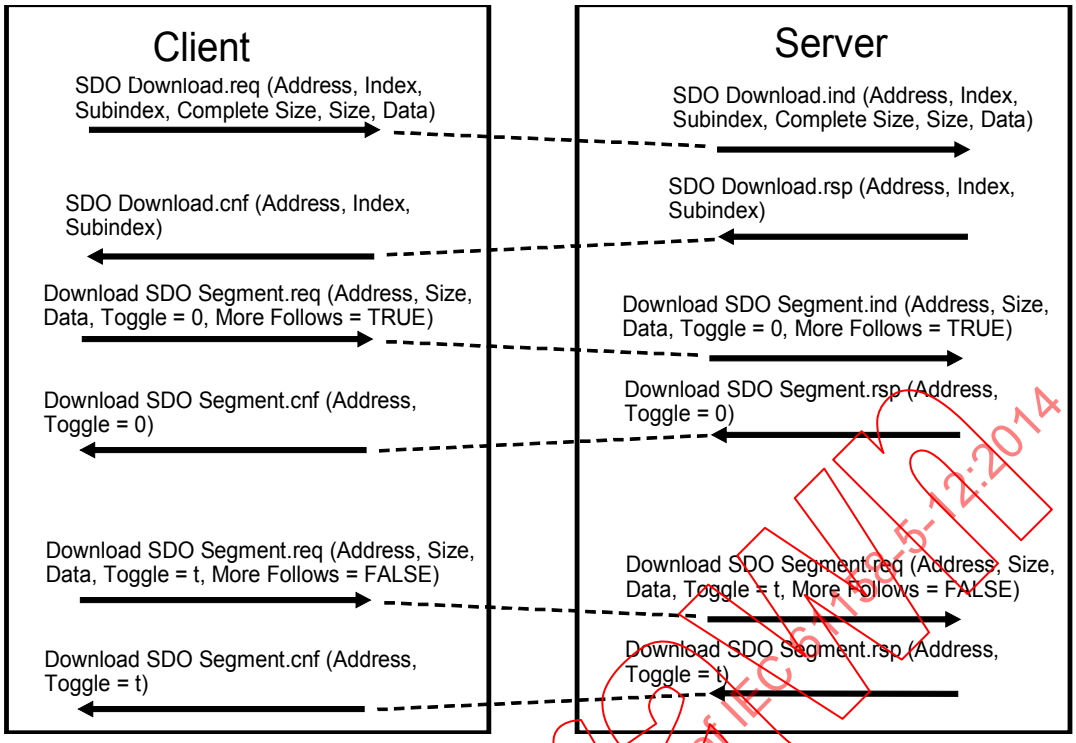


Figure 13 – Successful segmented SDO Download sequence

6.1.4.1.3.3 SDO Upload Sequence

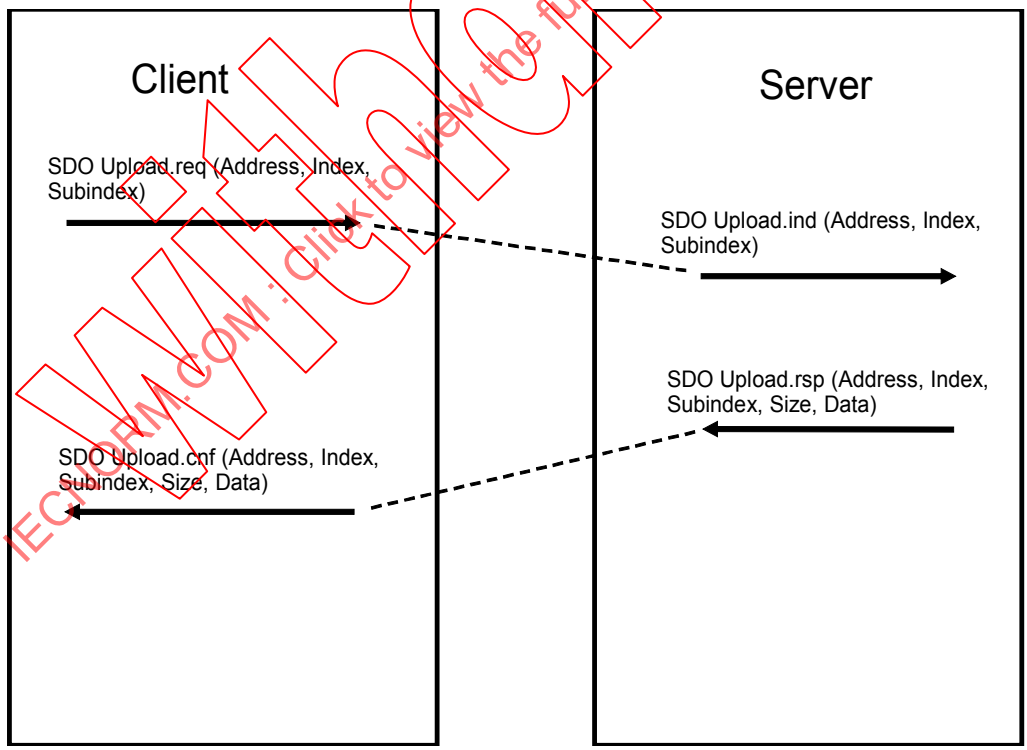


Figure 14 shows the primitives between client and server in case of a successful single SDO Upload sequence.

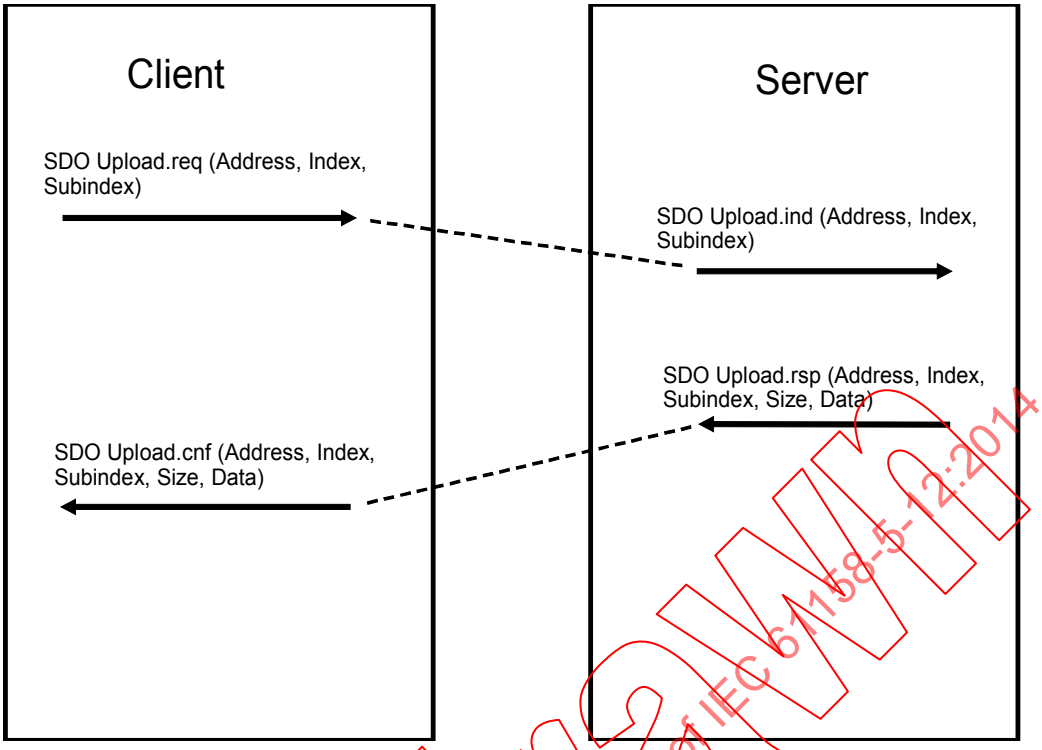


Figure 14 – Successful single SDO-Upload sequence

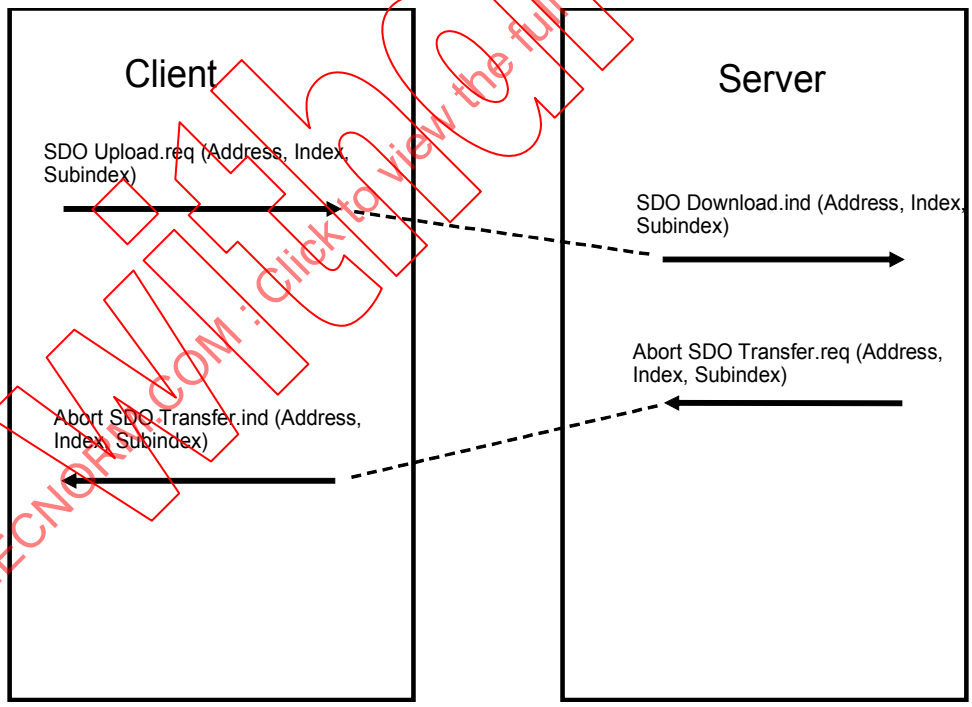


Figure 15 shows the primitives between client and server in case of a unsuccessful SDO Upload sequence. The service parameter Address, Index and Subindex should be the same in the Upload request and the Abort request.

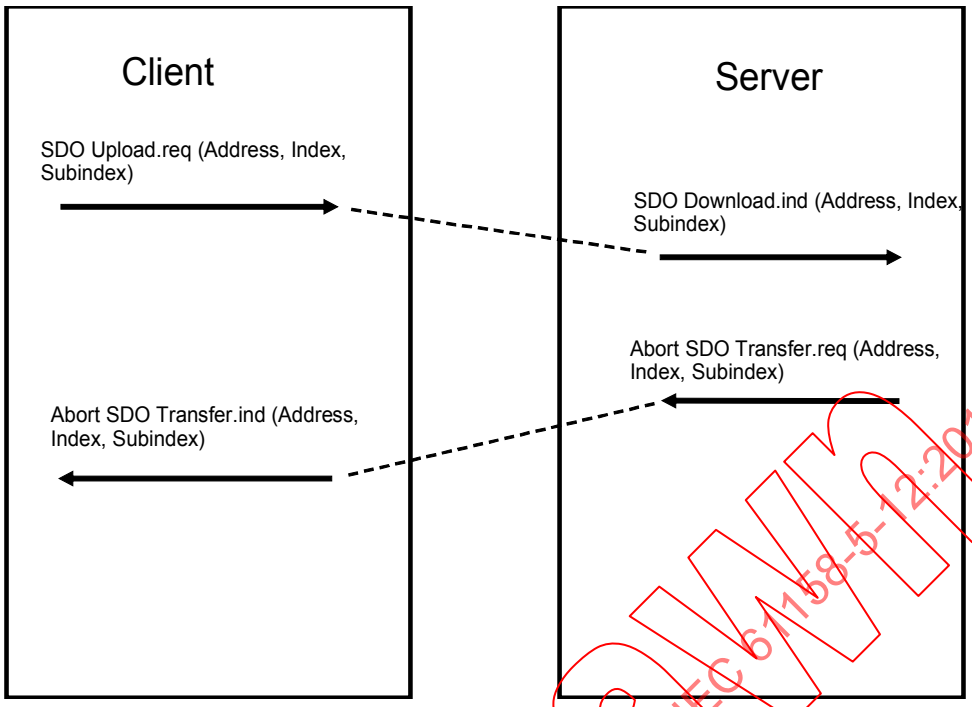


Figure 15 – Unsuccessful single SDO-Upload sequence

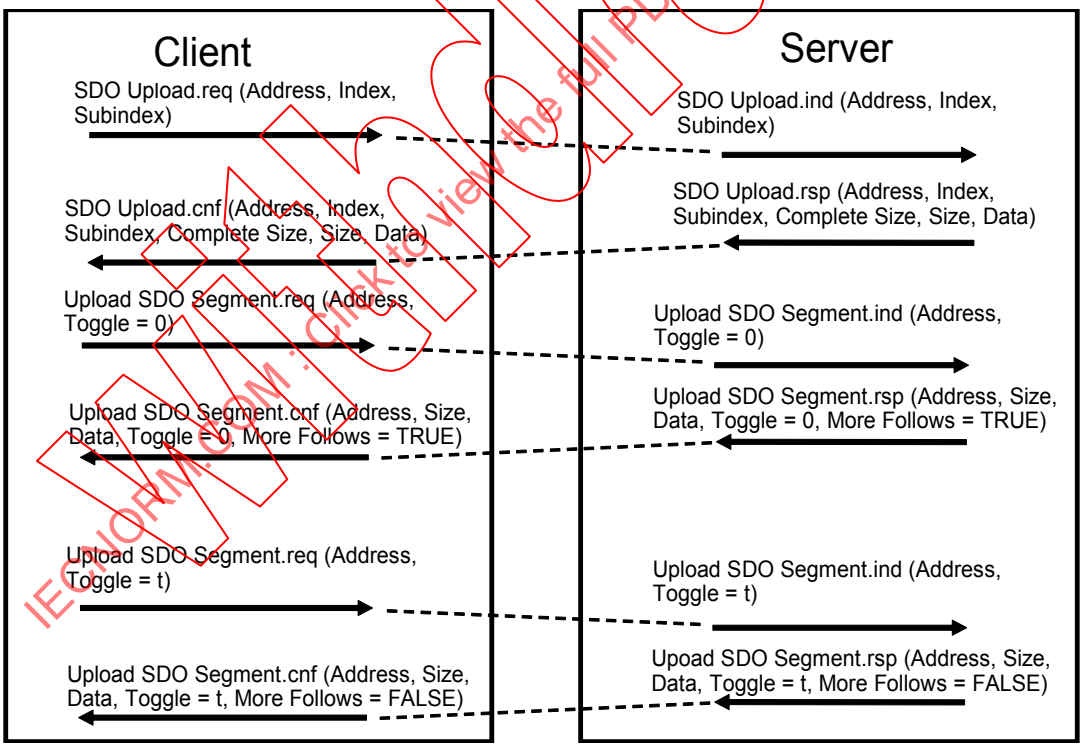


Figure 16 shows the primitives between client and server in case of a successful segmented SDO Upload sequence.

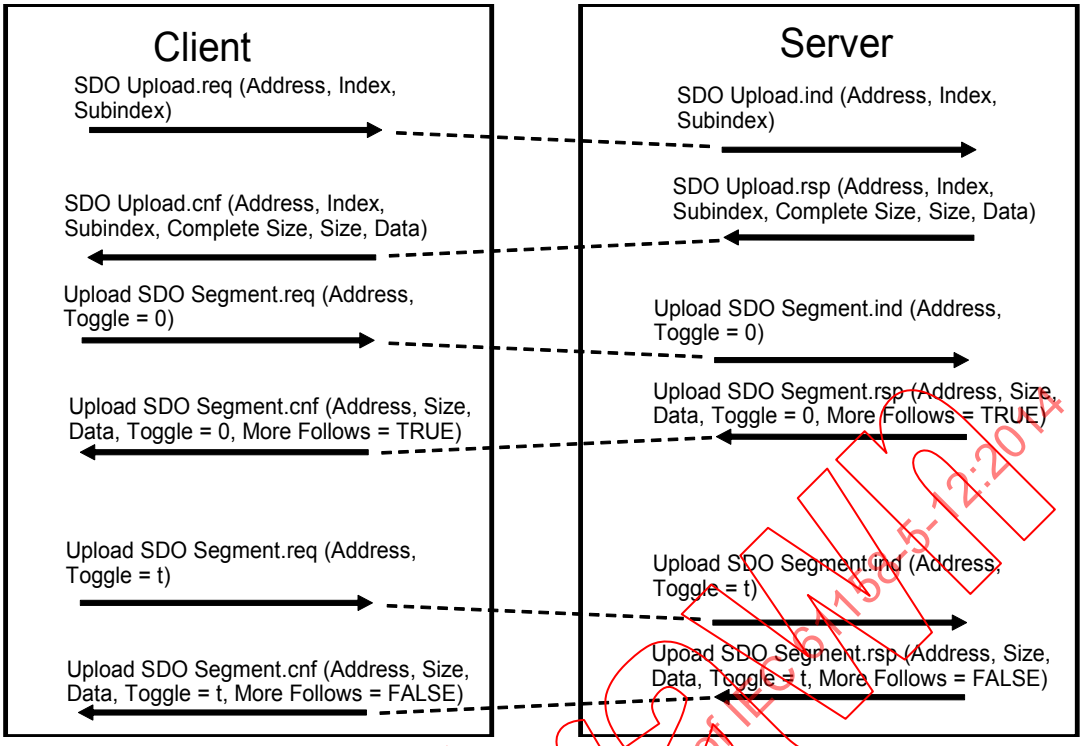


Figure 16 – Successful segmented SDO-Upload sequence

6.1.4.1.3.4 Information services

With the SDO information services the object dictionary of a server can be read by a client. The primitives of the SDO information services are mapped to the primitives of the mailbox transmission services.

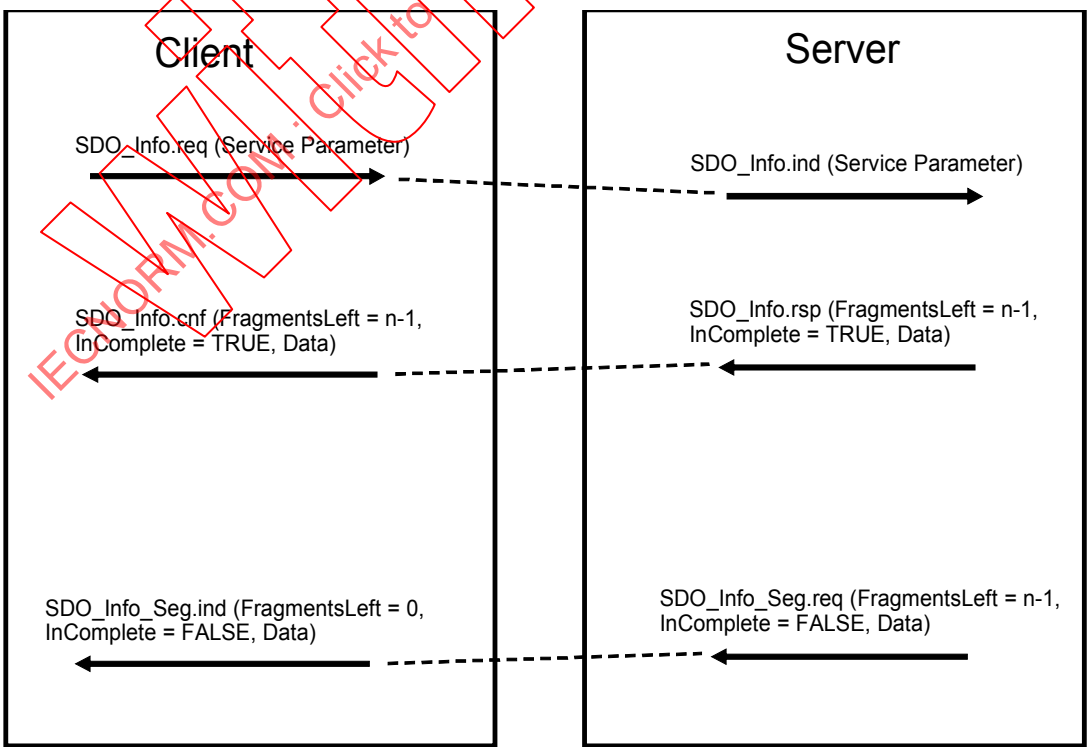


Figure 17 shows the primitives between client and server which is the same for all SDO information services named as SDO_Info. The fragmentation is done with an SDO_Info_Seg service.

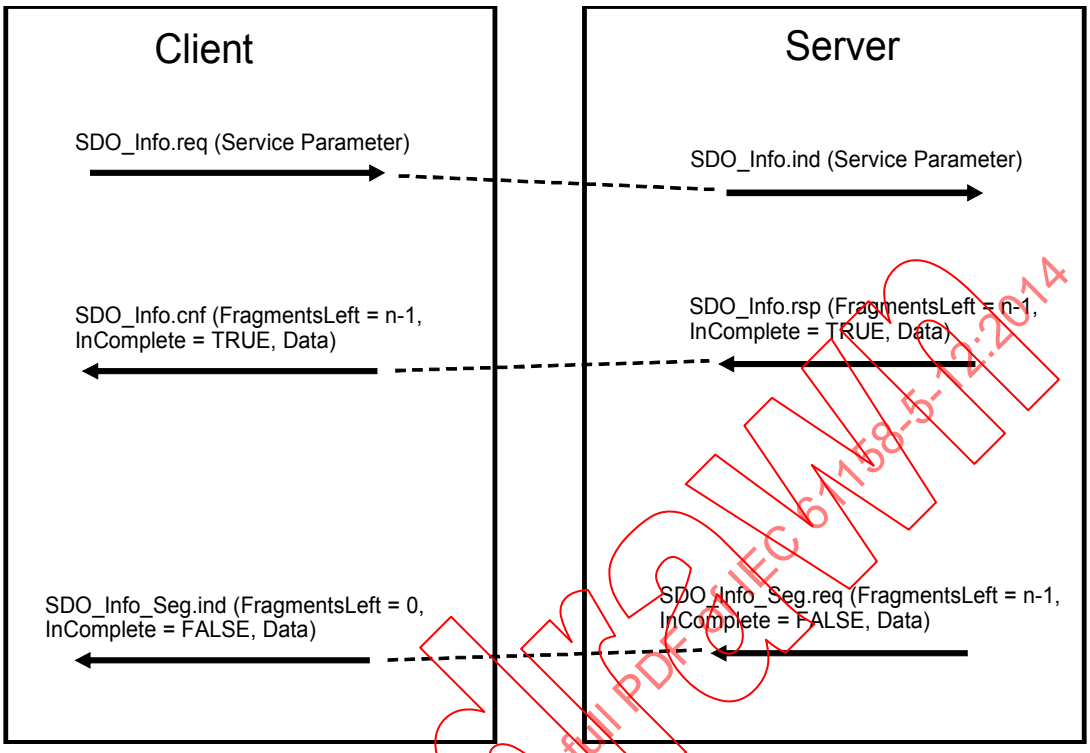


Figure 17 – SDO information sequence

6.1.4.1.4 Emergency interaction

Emergency messages are triggered by the occurrence of a device internal error situation. The transmission is executed via the mailbox interface.

The primitives of the Emergency service are mapped to the primitives of the mailbox transmission services.

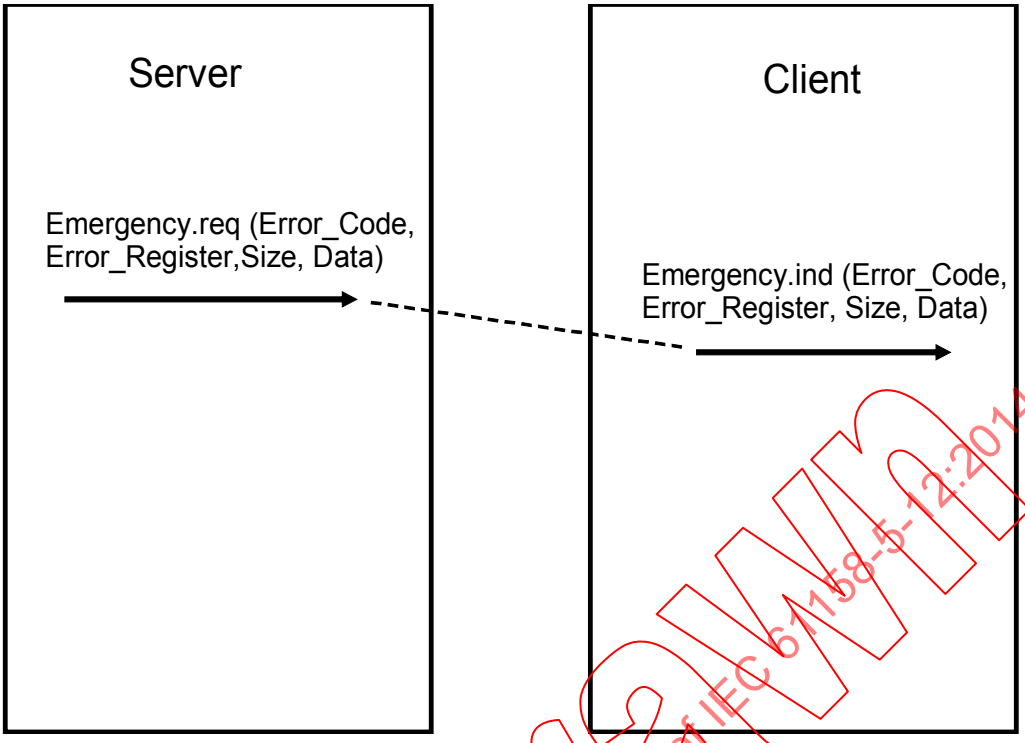


Figure 18 shows the primitives between server and client in case of an Emergency service.

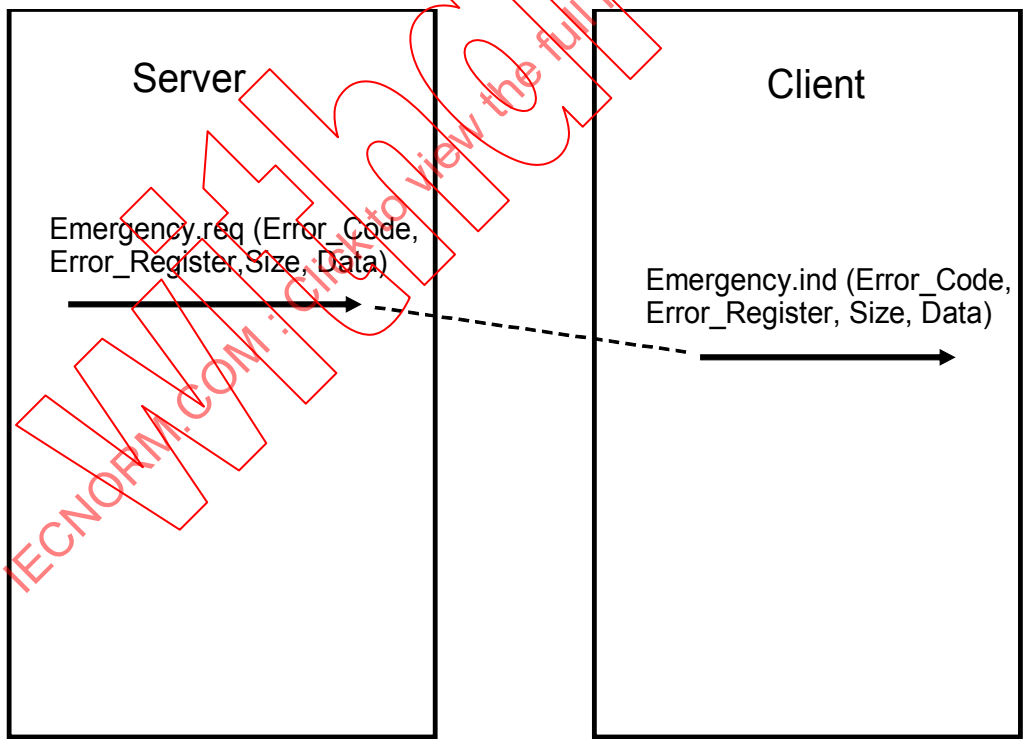


Figure 18 – Emergency service

6.1.4.1.5 Command

Command objects can be used for operations where data has to be sent in the request and the response.

An operation will be started in the server by a writing subindex 1 of the command object with the command request data by a SDO Download service. The status and the response data of the operation will be read with a SDO Upload to subindex 3 of the command object.

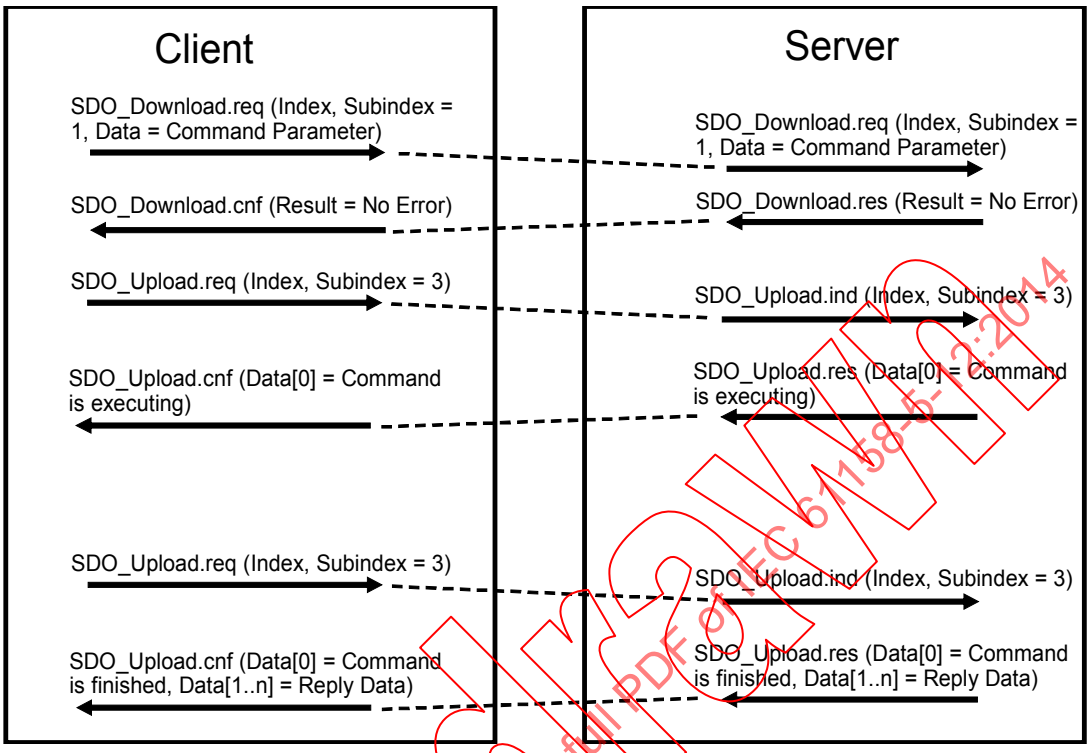


Figure 19 shows the primitives between server and client in case of a Command service.

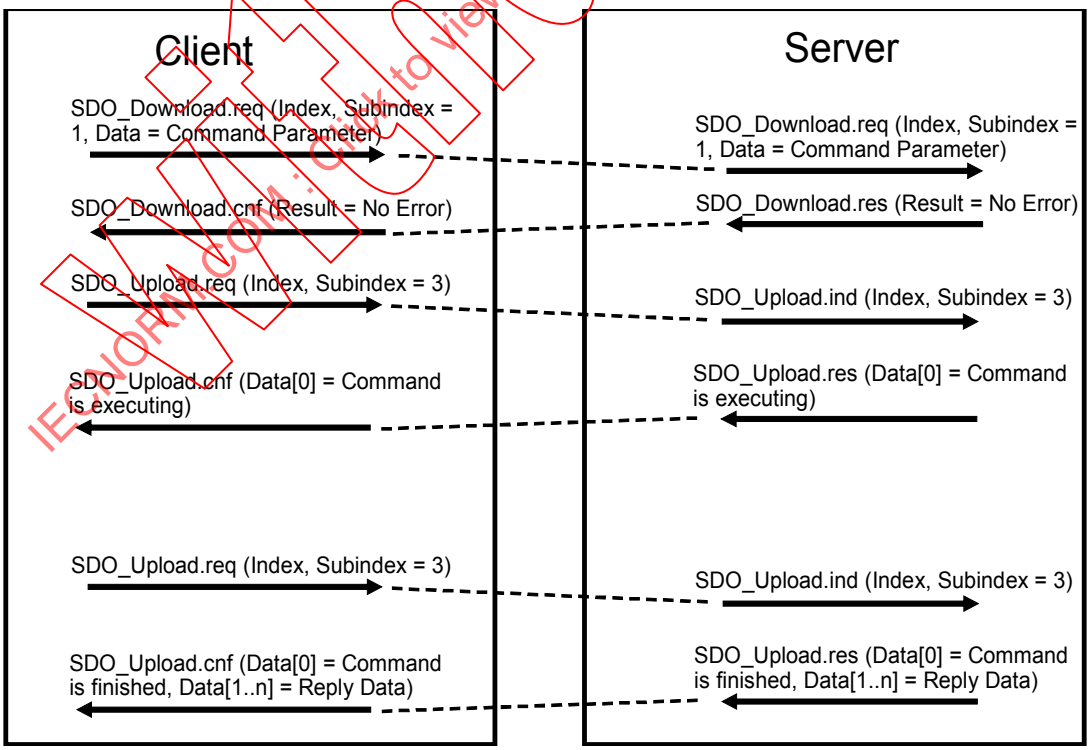


Figure 19 – Command sequence

6.1.4.1.6 Process data interaction

6.1.4.1.6.1 Mapping of objects to process data

The process data of a slave are specified in process data ASE. Each Sync Manager channel PDO assignment object describes a consistent area inside the process data ASE and consists of several (1 to 255) process data objects.

Process data objects (PDO) consists of objects in the object dictionary which can be mapped to PDOs. The PDO mapping objects contains a list of object references together with the length in bit.

PDO mapping refers to mapping of the application objects (real time process data) from the object dictionary to the PDOs. The device profiles provide a default mapping for every device type which is appropriate for most applications. E.g. the default mapping for digital I/O simply represents the inputs and outputs in their topological order in the transmit and receive PDOs.

The current mapping can be read by means of corresponding entries in the object directory. These are known as the mapping tables. The first location in the mapping table (sub-index 0) specifies the number of mapped objects that are listed after it. The tables are located in the object directory at index 0x1600 to 0x17FF for the RxPDOs and at 0x1A00 to 0x1BFF for the TxPDOs.

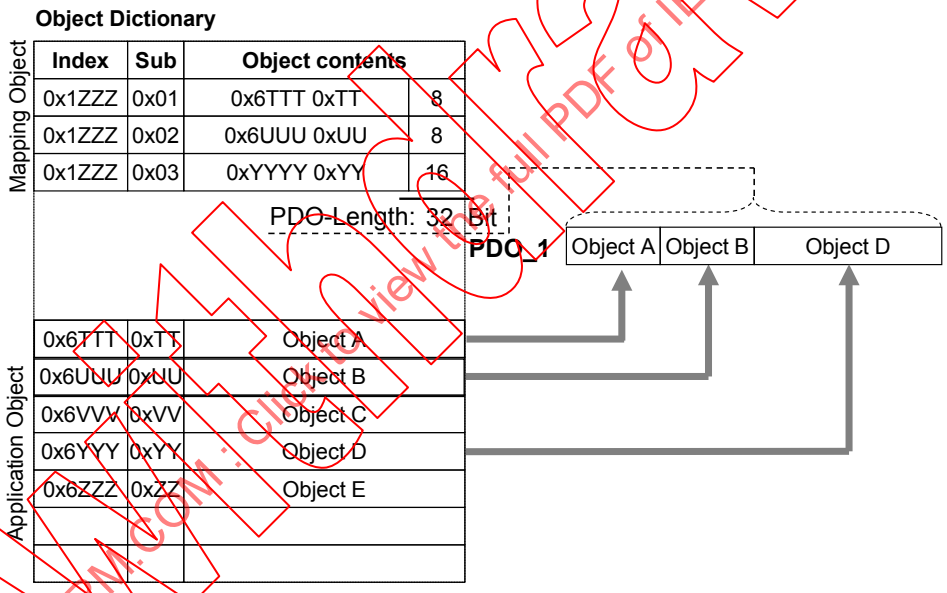


Figure 20 shows an example of a PDO mapping.

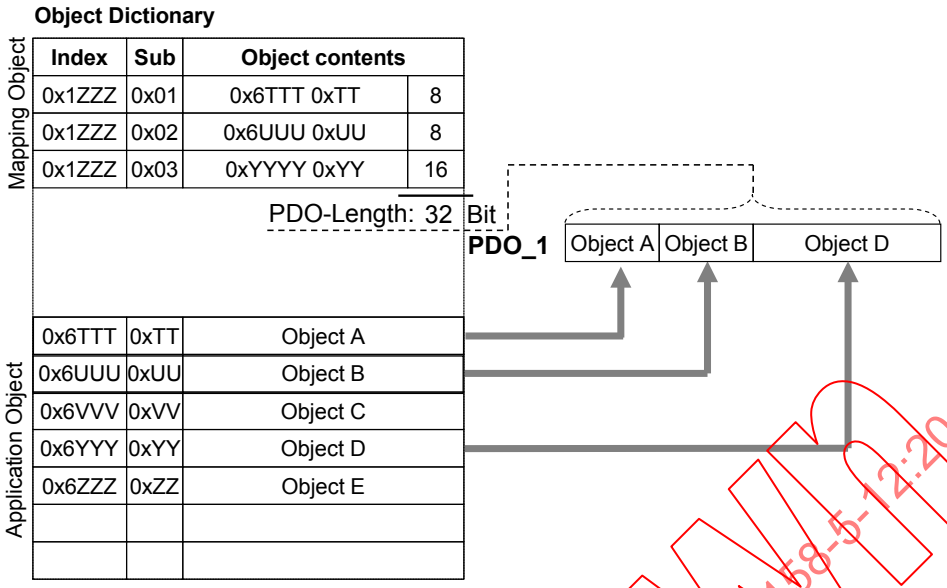


Figure 20 – PDO mapping

Sync Manager channel consists of several PDOs. The Sync Manager PDO assign objects describes how these PDOs are related to a Sync Manager.

For devices with a configurable mapping or devices allowing an access to the process data by different master tasks, the Sync Manager PDO assign objects should be writable.

The current assignment can be read by means of corresponding entries in the object directory. The first location in the Sync Manager PDO assignment table (sub-index 0) specifies the number of assigned PDOs that are listed after it. The tables are located in the object directory at index 0x1C10 to 0x1C2F.

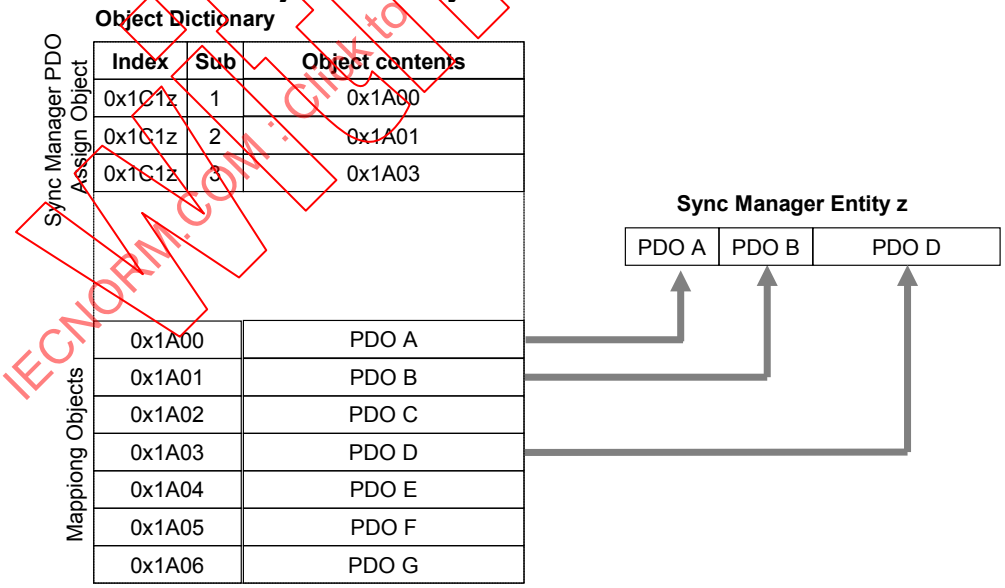


Figure 21 shows an example of a PDO assignment.

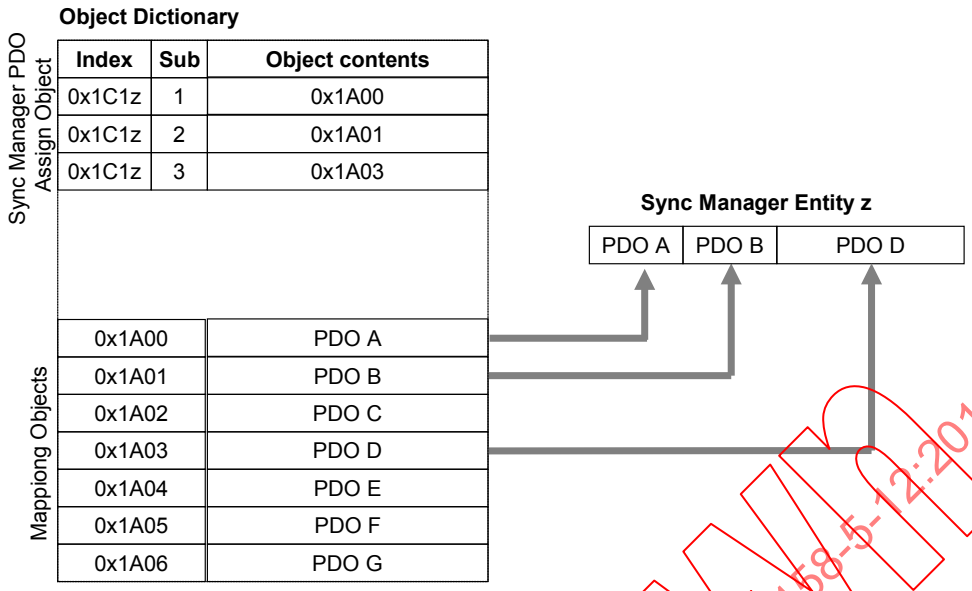


Figure 21 – Sync manager PDO assigment

Slaves with mailbox communication support the reading of the PDO mapping and the Sync Manager PDO assign objects. For slaves with a configurable mapping there are two possibilities how this configurable mapping should be supported. In the first possibility the slave supports different mappings described in the PDO mapping objects which are only readable. The master configures the actual mapping by selecting the desired PDO mapping objects when writing the Sync Manager PDO assign objects. The second possibility for more enhanced devices allows the master to write the PDO mapping objects to get a more flexible mapping configuration possibility.

6.1.4.1.6.2 Process data objects (PDO)

Each process data object has the following parameters.

Parameter

Number

This parameter specifies the number to identify the PDO.

Direction

This parameter specifies the direction Rx (master to slave, client to server) or Tx (slave to master, server to client) of the PDO.

Number of mapped Objects

This parameter specifies the number of mapped objects which are related to the PDO.

List of mapped Objects

This parameter specifies the related objects which define the mapping of the PDO.

6.1.4.1.6.3 Sync manager PDO assignment

Each Sync Manager channel object has the following parameters.

Parameter

Number

This parameter specifies the number to identify the Sync Manager channel.

Direction

This parameter specifies the direction Output (master to slave) or Input (slave to master) of the Sync Manager channel.

Number of assigned PDOs

This parameter specifies the number of assigned PDOs which are related to the Sync Manager channel.

List of assigned PDOs

This parameter specifies the assigned PDOs to the Sync Manager channel.

6.1.4.1.6.4 PDO transmission with CoE

With the PDO services process data objects can be transmit via the mailbox Interface acyclically from the client to the server (RxPDO) or from the server to the client (TxPDO).

The primitives of the PDO services are mapped to the primitives of the mailbox transmission services.

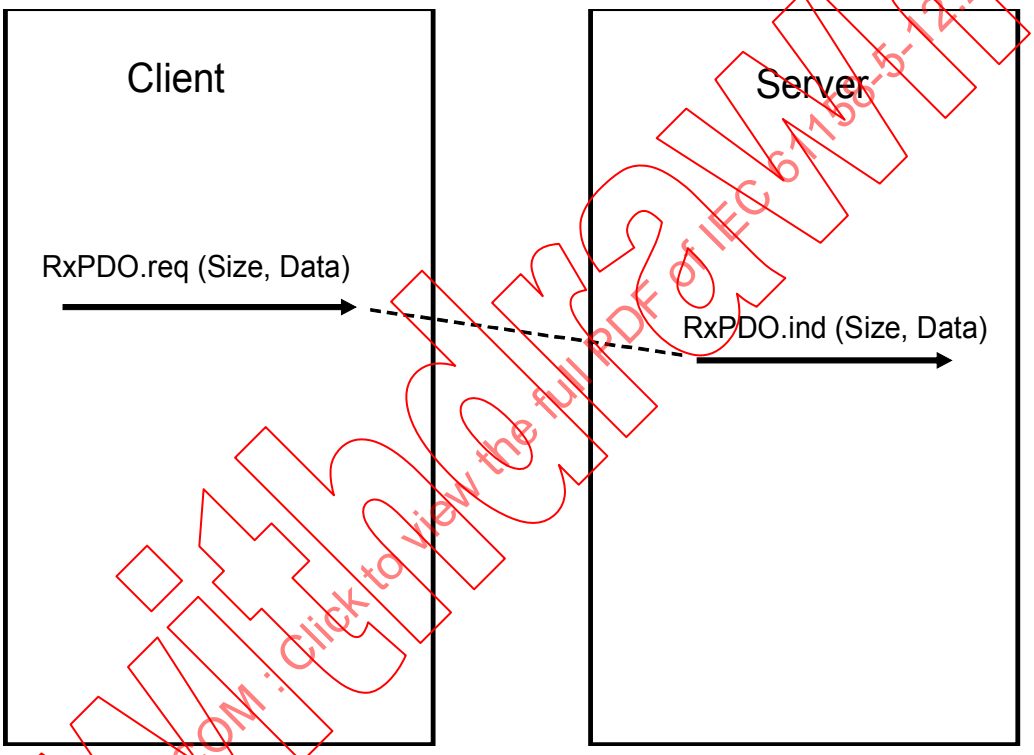


Figure 22 shows the primitives between client and server in case of a RxPDO service.

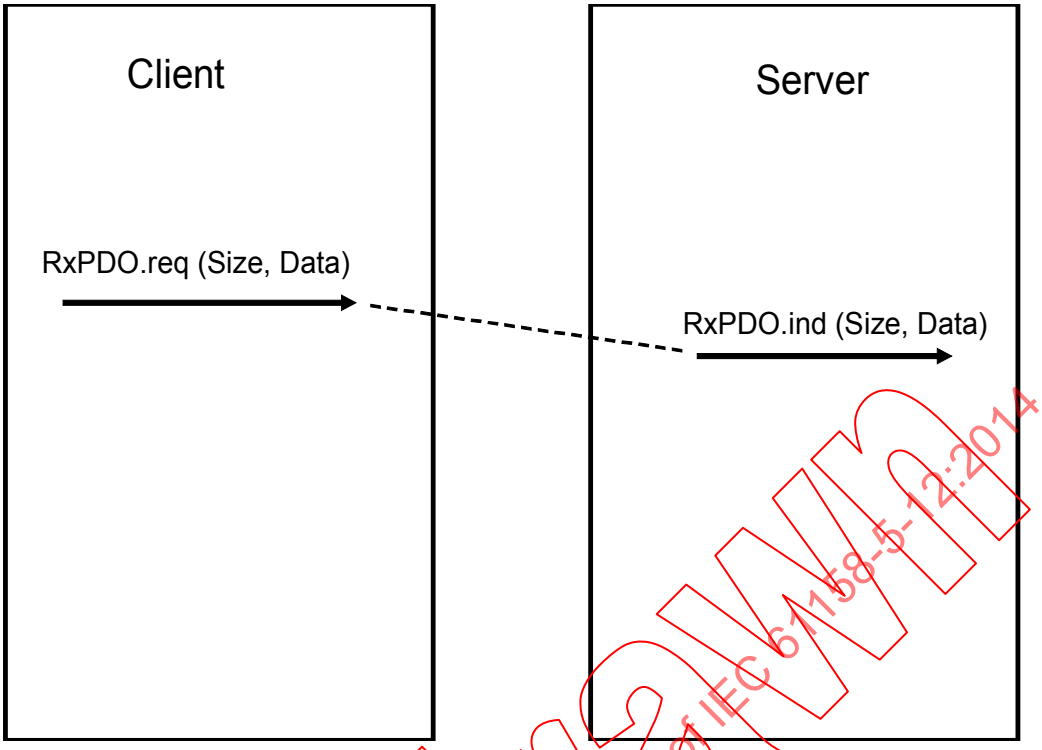


Figure 22 – RxPDO service

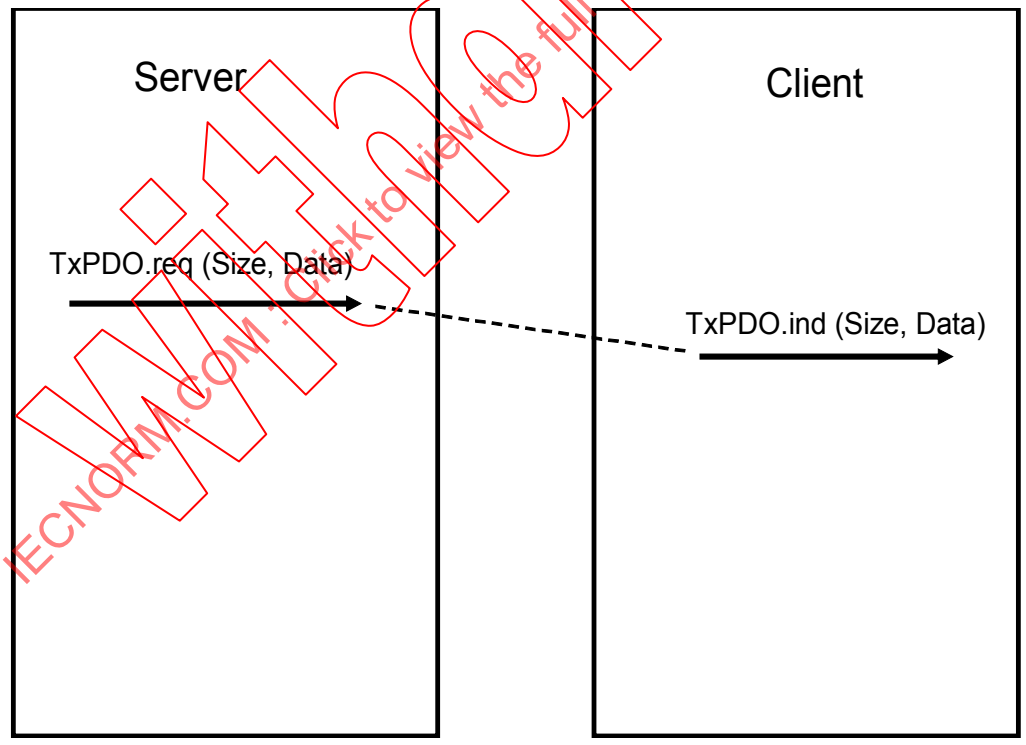


Figure 23 shows the primitives between server and client in case of a TxPDO service.

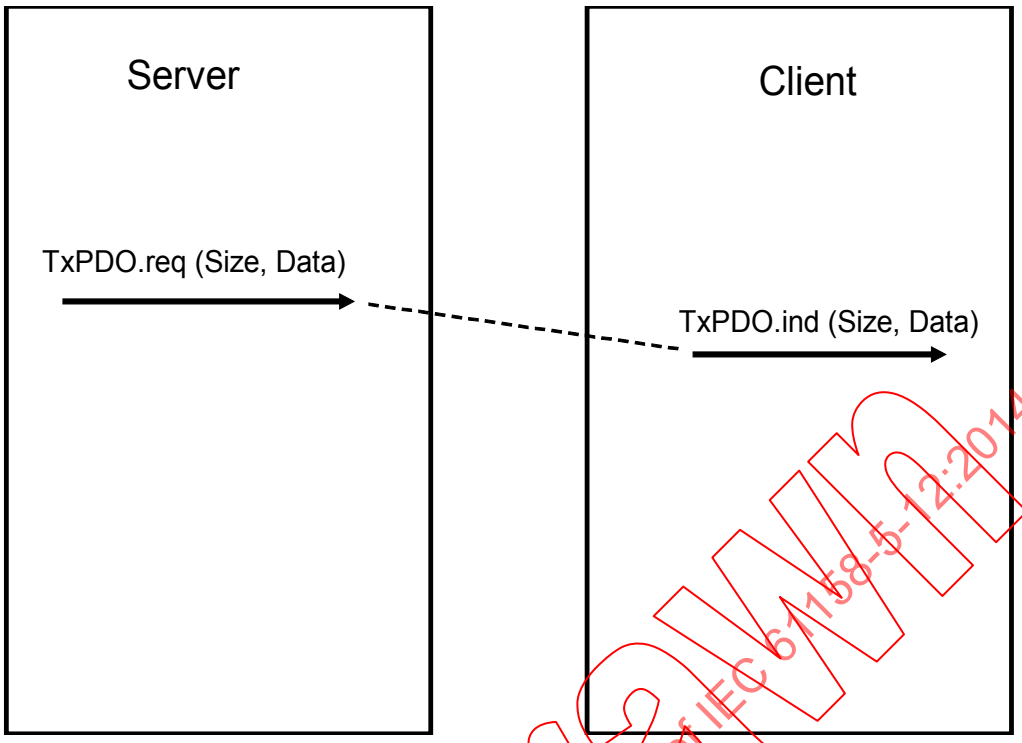


Figure 23 – TxPDO service

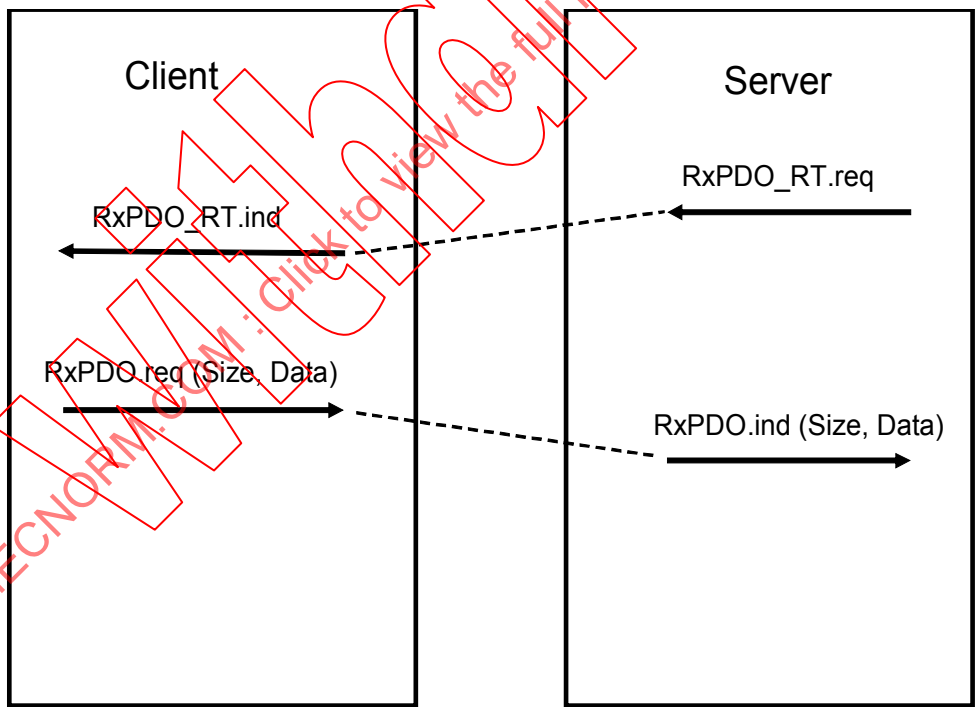


Figure 24 shows the primitives between client and server in case of a RxPDO Remote Transmission service.

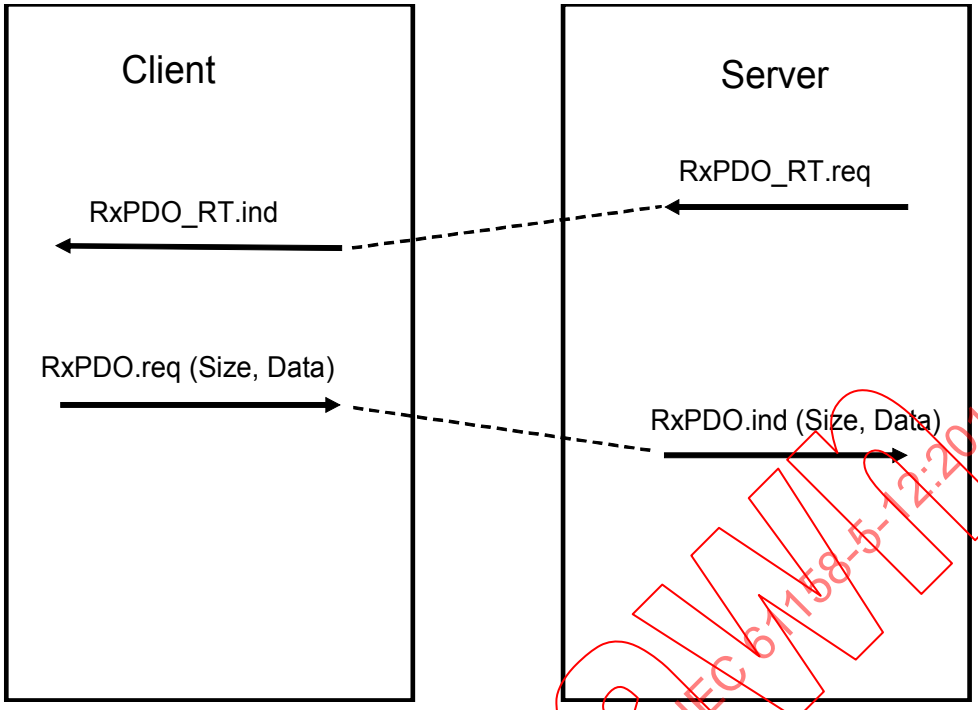


Figure 24 – RxPDO remote transmission sequence

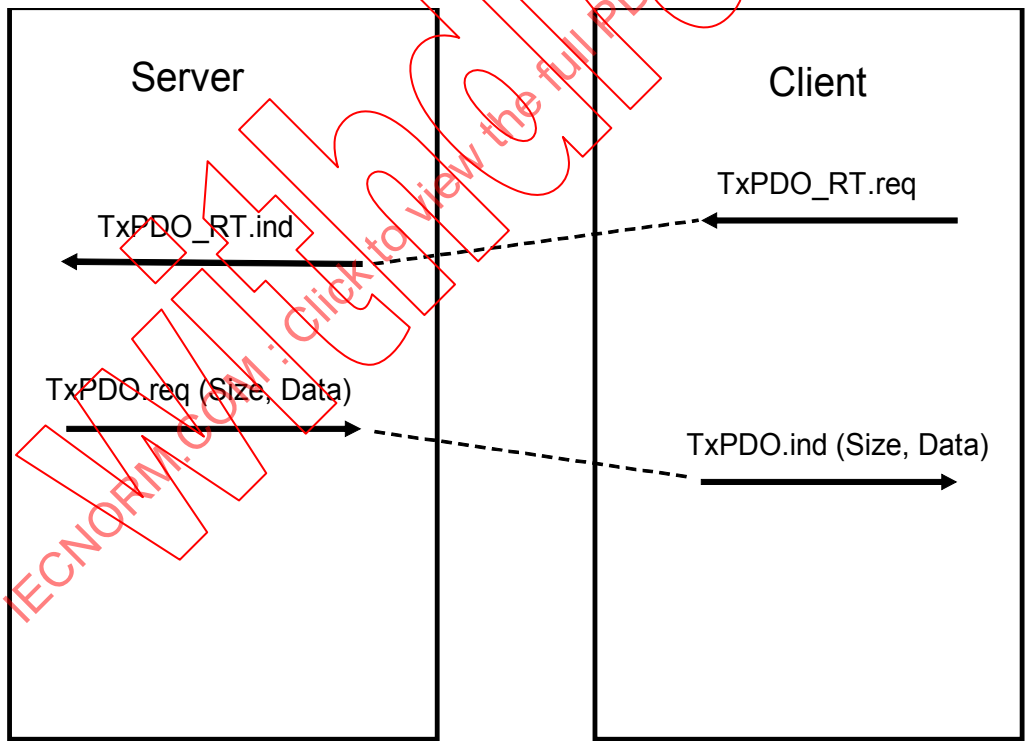


Figure 25 shows the primitives between client and server in case of a TxPDO Remote Transmission service.

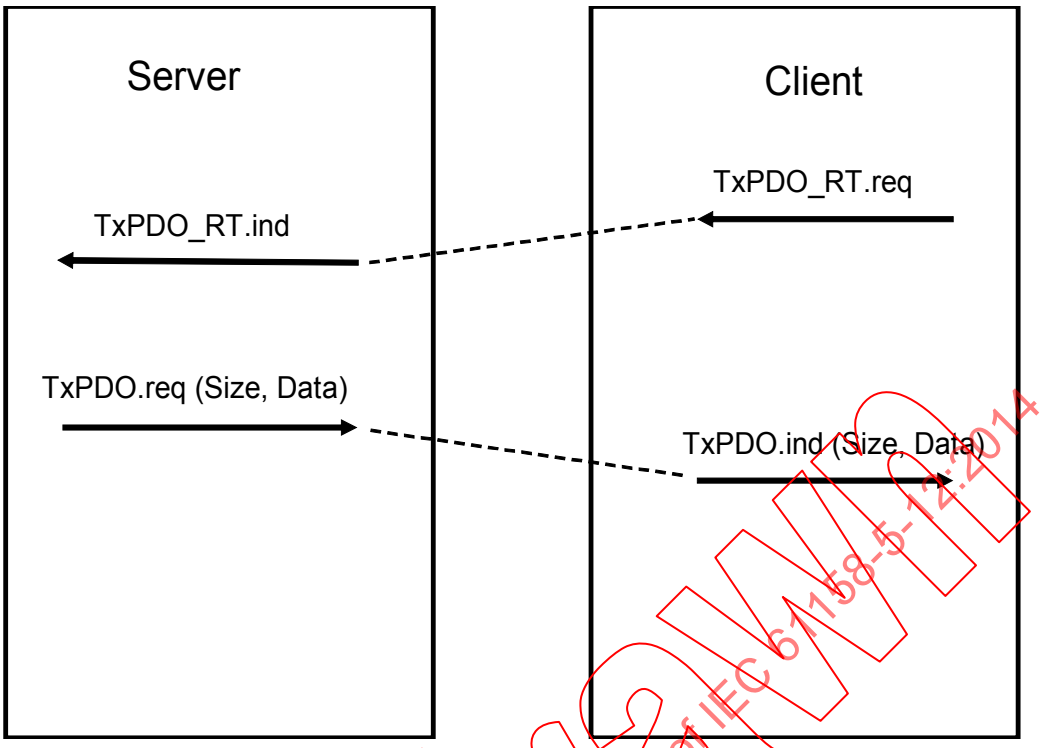


Figure 25 – TxPDO remote transmission sequence

6.1.4.2 CoE class specification

6.1.4.2.1 Formal model

The CoE Object Dictionary object is described by the following template:

ASE:			**CoE**
CLASS:			**CoE Object Dictionary**
CLASS ID:			not used
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Implicit
2.	(m)	Attribute:	List of Objects
2.1	(m)	Attribute:	Index
2.2	(m)	Attribute:	List of Entry
2.2.1	(m)	Attribute:	Subindex
2.2.2	(m)	Attribute:	Bitlen
2.2.3	(o)	Attribute:	Access
2.2.4	(o)	Attribute:	DataType
2.2.5	(o)	Attribute:	DefaultValue
2.2.6	(o)	Attribute:	MinValue
2.2.7	(o)	Attribute:	MaxValue
2.2.8	(o)	Attribute:	Description
SERVICES:			
1.	(o)	OpsService:	SDO Download Expedited
2.	(o)	OpsService:	SDO Download Normal
3.	(o)	OpsService:	Download SDO Segment
4.	(m)	OpsService:	SDO Upload Expedited
5.	(m)	OpsService:	SDO Upload Normal
6.	(o)	OpsService:	Upload SDO Segment
7.	(m)	OpsService:	Abort SDO Transfer
8.	(m)	OpsService:	Get OD List
9.	(o)	OpsService:	Get Object Description
10.	(o)	OpsService:	Get Entry Description
11.	(o)	OpsService:	OD List Segment
12.	(o)	OpsService:	Object Entry Segment
13.	(o)	OpsService:	Emergency
14.	(o)	OpsService:	RxPDO
15.	(o)	OpsService:	TxPDO
16.	(o)	OpsService:	RxPDO Remote Transmission
17.	(o)	OpsService:	TxPDO Remote Transmission

6.1.4.2.2 Attributes

Implicit

The attribute Implicit indicates that the CoE Object Dictionary object is implicitly addressed by the services.

List of objects

One Object is composed of the following list elements:

Index

This attribute specifies a numerical index of the object.

List of Entry
This attribute consists of the following attributes.

Subindex
This attribute specifies to the reference to the object entry.

Bitlen
This attribute specifies to the length in bits of the object entry value.

For PDOs, the length should be ≤ 11 888.

Access
This attribute specifies the access rights to that entry. Its allowed values are

- RP (Read in Pre-Operational)
- RS (Read in Safe-Operational)
- RO (Read in Operational)
- WP (Write in Pre-Operational)
- WS (Write in Safe-Operational)
- WO (Write in Operational)
- RX (usable for RxPDO)
- TX (usable for TXPDO)

DataType
This optional attribute specifies to the data type of the object entry value.

DefaultValue
This optional attribute specifies to the default value of the object entry value.

MinValue
This optional attribute specifies to the smallest possible value of the object entry value.

MaxValue
This optional attribute specifies to the largest possible value of the object entry value.

Description
This optional attribute gives a description of the object entry value.

6.1.4.2.3 Services

SDO downloadexpedited
Writes up to four octets to the slave.

SDO download normal
Writes up to to a negotiated number of octets to the slave.

Download SDO segment
Writes additional data if the object size is greater than the negotiated number of octets.

SDO upload expedited
Reads up to four octets from the slave.

SDO upload normal
Reads p to to a negotiated number of octets from the slave.

Upload SDO segment
Reads additional data if the object size is greater than the negotiated number of octets.

Abort SDO transfer
Server abort of service in case of an errorneous condition.

Get OD list
Reads a list of available indices.

Get object description
Reads details of an index.

Get entry description
Reads details of a subindex.

OD list segment
Continuation of the list of available indices.

Object entry segment
Continuation of an entry description.

Emergency
Report of unexpected conditions.

RxPDO
Unsolicited transfer of RxPDO Data from master to slave.

TxPDO
Unsolicited transfer of TxPDO Data from slave to master.

RxPDO remote transmission
Unsolicited request from slave to master to transfer of RxPDO Data.

TxPDO remote transmission
Unsolicited request from master to slave to transfer of TxPDO Data.

6.1.4.3 CoE service specification

6.1.4.3.1 Supported services

The CoE ASE defines the services

SDO Download Expedited
SDO Download Normal
Download SDO Segment
SDO Upload Expedited
SDO Upload Normal
Upload SDO Segment
Abort SDO Transfer
Get OD List
Get Object Description
Get Entry Description
OD List Segment
Object Entry Segment
Emergency
RxPDO
TxPDO
RxPDO Remote Transmission
TxPDO Remote Transmission

6.1.4.3.2 SDO download expedited service

The SDO Download Expedited service is used for an expedited transfer of up to 4 octets of data from the client to server. The server answers with the result of the download operation. Table 8 shows the service primitives and parameter of the SDO Download Expedited service.

Table 8 – SDO download expedited

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Index	M	M (=)		
Subindex	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object which it to be downloaded. Its range is 0 to 0x9FFF

Subindex
This parameter specifies the subindex in the object dictionary of the object which is to be downloaded.

Data
This parameter specifies the object value, of length 1 to 4 octets, to be downloaded.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.3 SDO download normal service

The SDO Download service will be used for a single normal transfer of data from the client to the server, if the number of octets to be downloaded fit in the mailbox. Otherwise a segmented transfer will be started. Table 9 shows the service primitives and parameter of the SDO Download Normal service.

Table 9 – SDO download normal

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Index	M	M (=)		
Subindex	M	M (=)		
Complete Access	M	M (=)		
Complete Size	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.				

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object which is to be downloaded. Its range is 0 to 0x9FFF.

Subindex
This parameter specifies the subindex in the object dictionary of the object which is to be downloaded.

Complete access
This Boolean parameter is set to true to indicate that all subindices should be written on that request.

Complete size
This parameter indicates the number of octets to be downloaded to the server. If the Complete Size is greater than the Size parameter the difference number of octets will be downloaded in the subsequent Download SDO Segment services.

Data
This parameter specifies the part of the object value, of length 1 to 1 477 octets, to be downloaded.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.4 Download SDO segment service

The Download SDO Segment service is used to download the additional data from the client to the server, which did not fit in the mailbox with SDO Download Normal service. The master will start as many Download SDO Segment services as data is available. Table 10 shows the service primitives and parameter of the Download SDO Segment service.

Table 10 – Download SDO segment

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Toggle	M	M (=)		
More Follows	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Toggle
This Boolean parameter should be toggled with every Download SDO Segment service, starting with zero.

More follows
This Boolean parameter is set to false if this service is the last Download SDO Segment service of a SDO download sequence. It is set to true if at least one Download SDO Segment service of a SDO download sequence is following.

Data
This parameter specifies the part of the object value, of length 1 to 1 477 octets, to be downloaded.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.5 SDO upload expedited service

The SDO Upload Expedited service can be used for an expedited transfer of up to 4 octets of data from the server to client. The server answers with the result of the upload operation and the requested data in case of a successful operation. Table 11 shows the service primitives and parameter of the SDO Upload Expedited service.

Table 11 – SDO upload expedited

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Index	M	M (=)		
Subindex	M	M (=)		
Result(+)			S	S
Data			M	M (=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object which is to be uploaded. Its range is 0 to 0x9FFF.

Subindex
This parameter specifies the subindex in the object dictionary of the object which is to be uploaded.

Result(+)
This selection type parameter indicates that the service request succeeded.

Data
This parameter specifies the object value, of length 1 to 4 octets, that is being uploaded to the server.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.6 SDO upload normal service

The SDO Upload Normal service is used for a single normal transfer of data from the server to the client, if the number of octets to be uploaded fit in the mailbox, or to start a segmented transfer with more octets. The server answers with the result of the upload operation and the requested data in case of a successful operation. Table 12 shows the service primitives and parameter of the SDO Upload Normal service.

Table 12 – SDO upload normal

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Index	M	M (=)		
Subindex	M	M (=)		
Complete Access	M	M (=)		
Result(+)			S	S
Complete Size			M	M (=)
Data			M	M (=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object which is to be uploaded. Its range is 0 to 0x9FFF

Subindex
This parameter specifies the subindex in the object dictionary of the object which is to be uploaded.

Complete access
This Boolean parameter is set to true to indicate that all subindices should be read on that request.

Result(+)
This selection type parameter indicates that the service request succeeded.

Complete size
This parameter indicates the number of octets to be uploaded to the server. If the Complete Size is greater than the Size parameter the difference number of octets will be uploaded in the subsequent Upload SDO Segment services.

Data
This parameter specifies the part of the object value, of length 1 to 1 477 octets, that is being uploaded to the server.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.7 Upload SDO segment service

The Upload SDO Segment service is used to upload the additional data from the server to the client, which did not fit in the mailbox with the SDO Upload service response. The client will start so many Upload SDO Segment services until all data is uploaded from the server. Table 13 shows the service primitives and parameter of the Upload SDO Segment service.

Table 13 – Upload SDO segment

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Toggle	M	M (=)		
Result(+)			S	S
More Follows			M	M (=)
Data			M	M (=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Toggle
This Boolean parameter should be toggled with every Upload SDO Segment service, starting by zero.

Result(+)
This selection type parameter indicates that the service request succeeded.

More follows
This Boolean parameter is set to false if this service is the last Upload SDO Segment service of a SDO upload sequence. It is set to true if at least one Upload SDO Segment service of a SDO upload sequence is following.

Data
This parameter specifies the part of the object value, of length 1 to 1 477 octets, that is being uploaded to the server.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.8 Abort SDO transfer service

The Abort SDO Transfer service is used from the server instead of a service response to a download or upload service in case of an error. Table 14 shows the service primitives and parameter of the Abort SDO Transfer service.

Table 14 – Abort SDO transfer

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Index	M	M (=)
Subindex	M	M (=)
Reason	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object whose transfer is to be aborted. Its range is 0 to 0x9FFF

Subindex
This parameter specifies the subindex in the object dictionary of the object whose transfer is to be aborted.

Reason
This parameter specifies the abort reason.

6.1.4.3.9 Get OD list service

With the Get OD List service lists of objects existing in the object dictionary of the server can be read. If the response does not fit in the mailbox the response is sent with multiple fragments as described above. Table 15 shows the service primitives and parameter of the Get OD service.

Table 15 – Get OD list

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
List Type	M	M (=)		
Result(+)			S	S
Incomplete			M	M (=)
Fragments Left			M	M (=)
Index List Entry			M	M (=)
Index List			M	M (=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

List type
This parameter should restrict the Index List Entries to objects that have at least one entry with the appropriate object access attribute set. Its allowed values are

- ALL
- RX (usable for RxPDO)
- TX (usable for TXPDO)

- BU (Back up values)
- ST (Setting values)

Result(+)
This selection type parameter indicates that the service request succeeded.

Incomplete
This Boolean parameter is set to false if the index list is completely contained in the data parameter and set to true if OD List Segment services are following.

Fragments left
This parameter is set to zero if the index list is completely contained in the data parameter and set to the appropriate number of following OD List Segment services.

Index list entry
This parameter indicates the number of entries in the Index list, between 1 and 736.

Index list
This array parameter specifies the first part of the object value which is to be uploaded.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.10 OD list segment service

With the OD List Segment service additional lists of objects existing in the object dictionary of the server can be transferred to the client. Table 15 shows the service primitives and parameter of the OD List Segment service.

Table 16 – OD list segment

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Incomplete	M	M (=)
Fragments Left	M	M (=)
List Type	M	M (=)
Index List Entry	M	M (=)
Index List	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Incomplete
This Boolean parameter is set to false if the index list is completely contained in the data parameter and set to true if OD List Segment services are following.

Fragments left
This parameter is set to zero if the index list is completely contained in the data parameter and set to the appropriate number of following OD List Segment services.

List type
This parameter should restrict the Index List Entries to objects that have at least one entry with the appropriate object acces attribute set. Its allowed values are

- ALL
- RX (usable for RxPDO)

- TX (usable for TXPDO)
- BU (Back up values)
- ST (Setting values)

Index list entry
This parameter indicates the number of entries in the Index list, between 1 and 736.

Index list
This array parameter specifies the first part of the object value which is to be uploaded.

6.1.4.3.11 Get object description service

With the Get Object Description service an object description existing in the object dictionary of the server can be read. If the response does not fit in the mailbox the response is sent with multiple fragments as described above. Table 17 shows the service primitives and parameter of the Get Object Description service.

Table 17 – Get object description

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Index	M	M (=)		
Result(+)			S	S
Incomplete			M	M (=)
Fragments left			M	M (=)
Max Subindex			M	M (=)
Object Code			M	M (=)
Name			M	M (=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object description which is to be read. Its range is 0 to 0x9FFF

Result(+)
This selection type parameter indicates that the service request succeeded.

Incomplete
This Boolean parameter has the value false.

Fragments left
This parameter is set to zero.

Max subindex
This parameter is set to the greatest subindex number.

Object code
This parameter specifies the general type information. Its allowed values are

- VAR
- ARRAY
- RECORD

Name
This parameter specifies the Name of the object.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.12 Get entry description service

With the Get Entry Description service an entry description existing in the object dictionary and addressed by index and subindex of the server can be read. If the response does not fit in the mailbox the response is sent with multiple fragments as described above. Table 18 shows the service primitives and parameter of the Get Entry Description service. If the response parameters do not fit in a single transfer unit of the underlying communication system, the optional parameters can be transmitted in following Entry Description Segment services.

Table 18 – Get entry description

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Index	M	M (=)		
Subindex	M	M (=)		
Request Access Rights	M	M (=)		
Request Object Category	M	M (=)		
Request Mapping Info	M	M (=)		
Request Unit Type	M	M (=)		
Request Default Value	M	M (=)		
Request Min Value	M	M (=)		
Request Max Value	M	M (=)		
Request Description	M	M (=)		
Result(+)			S	S
Incomplete			M	M (=)
Fragments left			M	M (=)
Data type			M	M (=)
Bit Length			M	M (=)
Access Rights			O	O(=)
Object Category			O	O(=)
Mapping Info			O	O(=)
Unit Type			O	O(=)
Default Value			O	O(=)
Min Value			O	O(=)
Max Value			O	O(=)
Description			O	O(=)
Result(–)			S	S

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Index
This parameter specifies the index in the object dictionary of the object entry description which is to be read. Its range is 0 to 0x9FFF

Subindex
This parameter specifies the subindex of the object entry description which is to be read.

Request access rights
This Boolean parameter is set to true if the parameter Access Rights is requested to be included in the response.

Request object category
This Boolean parameter is set to true if the parameter Object Category is requested to be included in the response.

Request mapping info
This Boolean parameter is set to true if the parameter Mapping Info is requested to be included in the response.

Request unit type
This Boolean parameter is set to true if the parameter Unit Type is requested to be included in the response.

Request default value
This Boolean parameter is set to true if the parameter Default Value is requested to be included in the response.

Request min value
This Boolean parameter is set to true if the parameter Min Value is requested to be included in the response.

Request max value
This Boolean parameter is set to true if the parameter Max Value is requested to be included in the response.

Request description
This Boolean parameter is set to true if the parameter Description is requested to be included in the response.

Result(+)
This selection type parameter indicates that the service request succeeded.

Incomplete
This Boolean parameter is set to false if the object entry is completely contained in the data parameter and set to true if Object Entry Segment services are following.

Fragments left
This parameter is set to zero if the object entry is completely contained in the data parameter and set to the appropriate number of following Object Entry Segment services.

Data type
This parameter points to the index of the data type in the object dictionary.

Bit length
This parameter defines the length in bits of the object entry value.

Access rights
This optional parameter defines the access right to the entry.

Object category
This optional parameter defines the object category of the object entry. Its allowed values are

- M (mandatory)
- O (optional)

Unit type
This optional parameter points to the index of the data type in the object dictionary.

Default value
This optional parameter shows the default value for this object entry.

Min value
This optional parameter shows the smallest possible value for this object entry.

Max value
This optional parameter shows the largest possible value for this object entry.

Description
This optional parameter specifies a textual description of the object entry.

Result(–)
This selection type parameter indicates that the service request failed.

6.1.4.3.13 Object entry segment service

With the Object Entry Segment service additional lists of objects existing in the object dictionary of the server can be transferred to the client. Table 19 shows the service primitives and parameter of the Object Entry Segment service. It is recommended that a continuation should follow the parameter boundary.

Table 19 – Object entry segment

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Incomplete	M	M (=)
Fragments left	M	M (=)
Index	M	M (=)
Subindex	M	M (=)
Unit Type	O	O(=)
Default Value	O	O(=)
Min Value	O	O(=)
Max Value	O	O(=)
Description	O	O(=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Incomplete
This Boolean parameter is set to false if the object entry is completely contained in the data parameter and set to true if Object Entry Segment services are following.

Index
This parameter specifies the index in the object dictionary of the object entry description which has been read. Its range is 0 to 0x9FFF

Subindex
This parameter specifies the subindex of the object entry description which has been read.

Fragments left
This parameter is set to zero if the index list is completely contained in the data parameter and set to the appropriate number of following Object Entry Segment services.

Unit type
This optional parameter points to the index of the data type in the object dictionary.

Default value
This optional parameter shows the default value for this object entry.

Min value
This optional parameter shows the smallest possible value for this object entry.

Max value
This optional parameter shows the largest possible value for this object entry.

Description
This optional parameter specifies a textual description of the object entry.

6.1.4.3.14 Emergency service

The Emergency service is used from the server to transmit diagnostic messages to the client. Each diagnostic event which was transmitted by the server to the client should be transmitted again with the reset error code when the diagnostic event disappears. Table 20 shows the service primitives and parameter of the Emergency service.

Table 20 – Emergency

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Error Code	M	M (=)
Error Register	C	C (=)
Data	O	O(=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Error code
This parameter specifies the emergency error code.

Error register
This conditional parameter specifies the emergency error register.

Data
This optional parameter specifies the emergency error details.

6.1.4.3.15 RxPDO service

The RxPDO service is used from the client to transmit output data to the server. The mapping and the size of the data is defined in the PDO mapping objects. Table 21 shows the service primitives and parameter of the RxPDO service.

Table 21 – RxPDO

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Index	M	M (=)
Data	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Index
This parameter specifies the index in the object dictionary of the RxPDO object to be written. It's permitted range is 0x1600 to 0x17FF.

Data
This parameter specifies the object value, of length 1 to 1 472 octets, to be written to the server.

6.1.4.3.16 TxPDO service

The TxPDO service is used from the server to transmit input data to the client. The mapping and the size of the data is defined in the PDO mapping objects. Table 22 shows the service primitives and parameter of the TxPDO service.

Table 22 – TxPDO

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Index	M	M (=)
Size	M	M (=)
Data	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Index
This parameter specifies the index in the object dictionary of the TxPDO object which is to be transfered. Its range is 0x1A00 to 0x1BFF.

Size
This parameter indicates the number of octets to be transferred to the client. Its range is 1 to 1 472.

The size should be identical to that of the PDO mapping object specified by the index parameter.

Data
This parameter specifies the object value which is to be transferred to the client.

6.1.4.3.17 RxPDO remote transmission service

The RxPDO_RT service is used from the server to request output data from the client. Table 23 shows the service primitives and parameter of the TxPDO Remote Transmission service.

Table 23 – RxPDO remote transmission

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Index	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Index
This parameter specifies the index in the object dictionary of the RxPDO object which is to be transmitted. Its range is 0x1600 to 0x17FF.

6.1.4.3.18 TxPDO remote transmission service

The TxPDO_RT service is used from the client to request input data from the server. Table 24 shows the service primitives and parameter of the TxPDO Remote Transmission service.

Table 24 – TxPDO remote transmission

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Index	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Index
This parameter specifies the index in the object dictionary of the TxPDO object which is to be transmitted. Its range is 0x1A00 to 0x1BFF.

6.1.5 EoE ASE

6.1.5.1 Overview

In addition to the already described addressing mode for the communication with the devices, an Ethernet fieldbus is also expected to feature standard IP-based protocols such as TCP/IP, UDP/IP and all higher protocols based on these (HTTP, FTP, SNMP etc.). Ideally, individual Ethernet frames should be transferred transparently, since this avoids restrictions with regard to the protocols to be transferred.

There are two different basic approaches for transferring acyclic Ethernet frames in cyclic fieldbus mode. In the first variant, an appropriate time slice is allocated, in which the acyclic Ethernet frames can be embedded. This time slice has to be chosen sufficiently large to be able to accommodate complete Ethernet frames. The maximum frame size of ISO/IEC 8802-3 is 1 526 octets (1 530 in case of tagged frames), corresponding to approximately 125 µs

transmission time (including inter-frame gap) at 100 Mbit/s. The minimum resulting cycle time for systems using this variant is approximately 200-250 µs. A reduction of the maximum frame size often leads to problems. While the IP protocol allows fragmentation in principle, this is often not recommended, and there may be protocols that do not support fragmentation. Data consistency problems may also occur, particularly in UDP/IP transfers.

Type 12 utilizes the second variant, in which the Ethernet frames are tunnelled and re-assembled in the associated device, before being relayed as complete Ethernet frames. This procedure does not restrict the achievable cycle time, since the fragments can be optimized according to the available bandwidth (Ethernet fragmentation instead of IP fragmentation). In this case, Type 12 defines a standard channel, which in principle enables any device to participate in the normal Ethernet traffic. In an intelligent drive controller that exchanges process data with a cycle time of 100 µs, for example, an HTTP server can be integrated that features its own diagnostics interface in the form of web pages.

Another application for transferred Ethernet frames will be devices with non Type 12 Ethernet ports. They offer standard Ethernet ports at any location within the system, through which any Ethernet device may be connected. For example, this may be a service computer that communicates directly with the control and queries the web page of an intelligent device, or simply routes it to the intranet or internet via the control. The master also features software-integrated switch functionality, which is responsible for the routing of the individual Ethernet frames from and to the devices and the IP stack of the host operating system. The switch functionality is identical with that of a standard layer 2 switch and responds to the Ethernet addresses used irrespective of the protocol.

There are special EoE Parameters needed for EoE end devices within a Type 12 segment. This parameter can be conveyed with the EoE services Set IP Parameter and Set MAC Filter.

The primitives of the EoE services are mapped to the primitives of the mailbox transmission services.

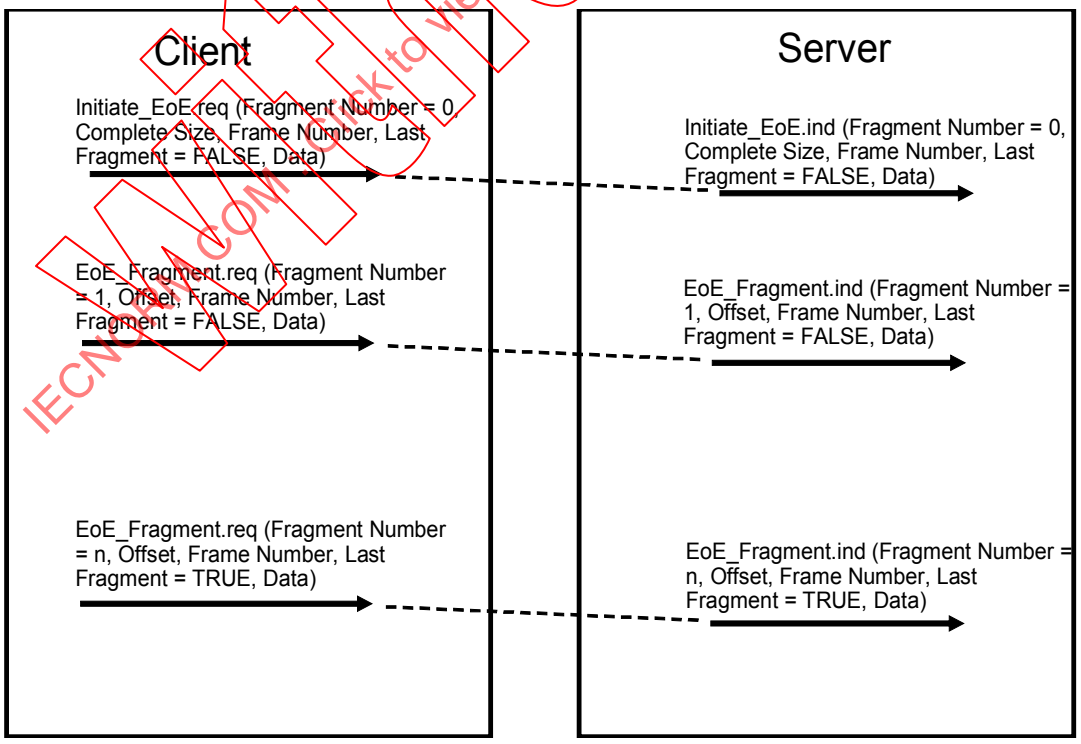


Figure 26 shows the primitives between client and server in case of a successful EoE sequence. The frame number has to be the same for all Ethernet fragments in an EoE sequence.

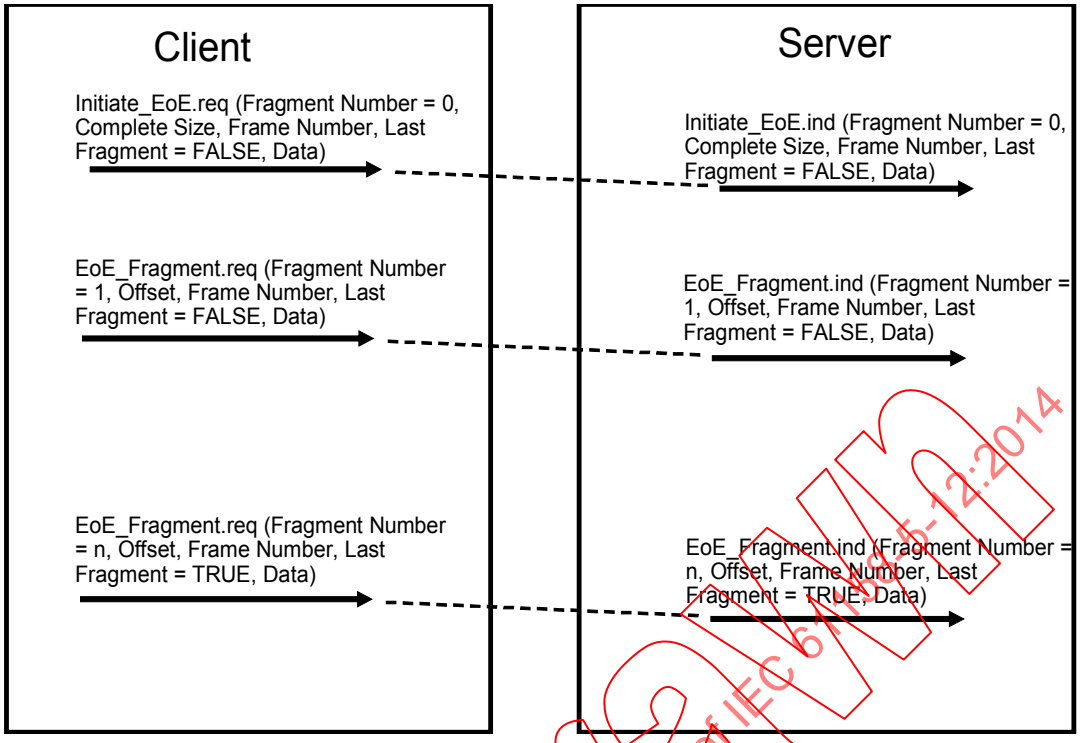


Figure 26 – EoE sequence

6.1.5.2 EoE class specification

6.1.5.2.1 Formal model

The EoE object is described by the following template:

ASE:			**EoE**
CLASS:			**EoE Parameter**
CLASS ID:			not used
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Implicit
2.	(m)	Attribute:	Virtual MAC Address
3.	(o)	Attribute:	IP Address Info
3.1	(o)	Attribute:	IP Address
3.2	(o)	Attribute:	IP Subnet mask
3.3	(o)	Attribute:	Default Gateway
3.4	(o)	Attribute:	DNS IP Address
3.5	(o)	Attribute:	Domain Name
4.	(o)	Attribute:	Filter Info
4.1	(o)	Attribute:	Broadcast Forwarding
4.2	(o)	Attribute:	List of MAC Address Filter
4.2.1	(o)	Attribute:	MAC Address
4.3	(o)	Attribute:	List of MAC Filter Mask
4.3.1	(o)	Attribute:	Filter Mask
5.	(m)	Attribute:	List of Transmit Frames
5.1	(m)	Attribute:	Frame
5.2	(o)	Attribute:	Port
5.3	(o)	Attribute:	Timestamp
6.	(m)	Attribute:	List of Receive Frames
6.1	(m)	Attribute:	Frame

6.2	(o)	Attribute:	Port
6.3	(o)	Attribute:	Timestamp

SERVICES:

1.	(m)	OpsService:	Initiate EoE
2.	(m)	OpsService:	EoE Fragment
3.	(o)	OpsService:	Set IP Parameter
4.	(o)	OpsService:	Set MAC Filter

6.1.5.2.2 Attributes

Implicit

The attribute Implicit indicates that the EoE Parameter object is implicitly addressed by the services.

Virtual MAC address
This attribute specifies a MAC Address according to ISO/IEC 8802-3 used for identifying a node in an Ethernet network.

IP Address info
One Object is composed of the following list elements:

IP address
This optional attribute consists of an IP address according to RFC 791 to identify an IP node in the internet.

Subnet mask
This optional attribute an Subnet mask specify a group of addresses belonging to an IP subnet.

Default gateway
This optional attribute consist of an IP address according to RFC 791 identifying the next router in the internet.

DNS IP address
This optional attribute consists of an IP address of a station which supports the translation of Domain names to IP addresses.

Domain name
This optional attribute is used to store the domain name of the node.

Filter Info
One Object is composed of the following list elements:

Broadcast forwarding
This optional Boolean attribute is set to true if broadcast PDUs should be forwarded

List of MAC address filter
This attribute consists of the following attributes.

MAC address
This attribute specifies a MAC Address according to ISO/IEC 8802-3 used for forwarding frames with a match in the MAC Address filter match.

List of MAC filter mask
This attribute consists of the following attributes.

MAC address
This attribute specifies a MAC Address mask according to ISO/IEC 8802-3 used to form a MAC address group for address filtering.

List of transmit frames
This attribute consists of the following attributes.

Frame
This attribute specifies a complete Ethernet frame.

Port
This optional attribute specifies the send port of the frame. Its range is 1 to 15.

Timestamp
This optional attribute specifies the timestamp in units of ns of the local time at the send of a frame.

List of receive frames
This attribute consists of the following attributes.

Frame
This attribute specifies a complete Ethernet frame that is error free.

Port
This optional attribute specifies the receive port of the frame. Its range is 1 to 15.

Timestamp
This optional attribute specifies the timestamp in units of ns of the local time at the receipt of a frame.

6.1.5.2.3 Services

Initiate EoE
Start to transfer Ethernet frame – a timestamp response can be requested

EoE fragment
Continuation of an Ethernet frame that can not be sent within one Type 12 Datagram

Set IP parameter
Set up IP parameters such as MAC address and IP address.

Set MAC filter
Set up Filters for MAC addresses or Groups of MAC addresses.

6.1.5.3 EoE service specification

6.1.5.3.1 Supported services

The EoE ASE defines the services

Initiate EoE

EoE Fragment

Set IP Parameter

Set MAC Filter

6.1.5.3.2 Initiate EoE

The Initiate EoE service is used to transmit the first fragment of an Ethernet frame. Table 25 shows the service primitives and parameter of the EoE Initiate service.

Table 25 – Initiate EoE

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Port	M	M (=)		
Time Appended	M	M (=)		
Time Requested	M	M (=)		
Frame Number	M	M (=)		
Complete Size	M	M (=)		
Last Fragment	M	M (=)		
Data	M	M (=)		
Timestamp	C	C (=)		
Result			C	C (=)
Timestamp			M	M (=)

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Port
This parameter specifies a receiving/transmitting port number of the frame. Its range is 0 to 15; 0 indicates that the port is unspecified.

Time appended
This Boolean parameter is set to true if time stamp of the time of receipt (beginning of first bit of DA) is appended. It is set to false otherwise.

Time requested
This Boolean parameter is set to true if the time stamp of the time of sending (beginning of first bit of DA) should be sent in the response. It is set to false otherwise.

Frame number
This parameter specifies a sequence number of the frame modulo 8.

Complete size
This parameter indicates the number of 32 octet blocks to be transferred with this frame (excluding Preamble, SFD, FCS). Its range is 2 to 48.

Last fragment
This Boolean parameter is set to true if this service is the last fragment of the Ethernet frame. It is set to false if at least one EoE Fragment service is following.

Data
This parameter specifies the object value, of length 64 to 1 472 octets, to be transferred to the client.

TimeStamp
This conditional parameter specifies a timestamp of the receipt of the frame in ns (begin of DA is timestamp-point)

Result
This parameter indicates an optional service response.

TimeStamp
This parameter specifies a timestamp of the transmission of the frame with the specified frame number

6.1.5.3.3 EoE Fragment

The EoE Fragment service is used to transmit the fragments of an Ethernet frame following the Initiate EoE service if the Ethernet frame to be transmitted is larger than the data parameter of the Initiate EoE service. Table 26 shows the service primitives and parameter of the EoE Fragment service.

Table 26 – EoE fragment

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Port	M	M (=)
Time Appended	M	M (=)
Frame Number	M	M (=)
Offset	M	M (=)
Last Fragment	M	M (=)
Data	M	M (=)
Time Stamp	C	C (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server.

Port
This parameter specifies a receiving/transmitting port number of the frame. Its range is 0 to 15; 0 indicates that the port is unspecified.

Time appended
This Boolean parameter is set to true if time stamp of the time of receipt (beginning of first bit of DA) is appended. It is set to false otherwise.

Frame number
This parameter is the same as the parameter Frame Number in the related Initiate EoE service.

Offset
This parameter multiplied with 32 indicates the offset in the Ethernet frame of the first data octet to be transferred with this frame (excluding Preamble, SFD). Its range is 2 to 47.

Last fragment
This Boolean parameter is set to true if this service is the last fragment of the Ethernet frame. It is set to false if at least one EoE Fragment service is following.

Data
This parameter specifies the object value, of length 64 to 1 472 octets, to be transferred to the client.

TimeStamp
This conditional parameter specifies a timestamp of the receipt of the frame in ns (begin of DA is timestamp-point)

6.1.5.3.4 Set IP parameter

The Set IP Parameter service is used transfer IP Parameters from the client to server. The server answers with the result of the set operation. Table 27 shows the service primitives and parameter of the Set IP Parameter service.

Table 27 – Set IP parameter

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
MAC Address	O	O(=)		
IP Address	O	O(=)		
Subnet Mask	O	O(=)		
Default Gateway	O	O(=)		
DNS Server	O	O(=)		
DNS Name	O	O(=)		
Result(+)			S	S (=)
Result(–)			S	S (=)
Reason			M	M (=)
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.				

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

MAC address
This optional parameter specifies a MAC Address according to ISO/IEC 8802-3 used for identifying a node in an Ethernet network.

IP address
This optional parameter consists of an IP address according to RFC 791 to identify an IP node in the internet.

Subnet mask
This optional parameter consists of an IP subnet mask according to RFC 791 to identify an IP subnet in the internet.

Default gateway
This optional parameter consists of an IP address according to RFC 791 to identify the standard router of this subnet to the internet.

DNS IP address
This optional parameter consists of an IP address of a station which supports the translation of Domain names to IP addresses.

Domain name
This optional parameter is used to store the domain name of the node.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

Reason
This parameter indicates the reason for the service failure.

6.1.5.3.5 Set address filter

The Set Address Filter service is used transfer Address Filters from the client to server. The server answers with the result of the set operation. Table 28 shows the service primitives and parameter of the Set Address Filter service.

Table 28 – Set address filter

Parameter name	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Broadcast Forwarding	O	O(=)		
MAC Address Filter1	O	O(=)		
..	O	O(=)		
MAC Address Filter16	O	O(=)		
MAC Filter Mask1	O	O(=)		
..	O	O(=)		
MAC Filter Mask4	O	O(=)		
Result(+)			S	S (=)
Result(–)			S	S (=)
Reason			M	M (=)
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.				

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the source station if a master is the client or the station address of the destination if a slave is the client to allow slave to slave communication.

Broadcast forwarding
This optional Boolean parameter is set to true to enable the broadcast filter.

MAC address Filter 1 to 16
These optional parameters contain a MAC Addresses according to ISO/IEC 8802-3 used for filtering frames not dedicated to this set of addresses in an Ethernet network.

MAC filter mask 1 to 4
These optional parameters contain a MAC Addresses according to ISO/IEC 8802-3 used to mask unused bits. The Mask coresponds to the Filter with the same index.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

Reason
This parameter indicates the reason for the service failure.

6.1.6 FoE ASE

6.1.6.1 Overview

The file access mailbox command specifies a standard way to download a firmware or any other files from a client to a server or to upload a firmware or any other files from a server to a client. The protocol is similar to the TFTP protocol (trivial file transfer protocol). Both sides can initiate a read or write request via a read or write command.

The primitives of the FoE services are mapped to the primitives of the mailbox transmission services.

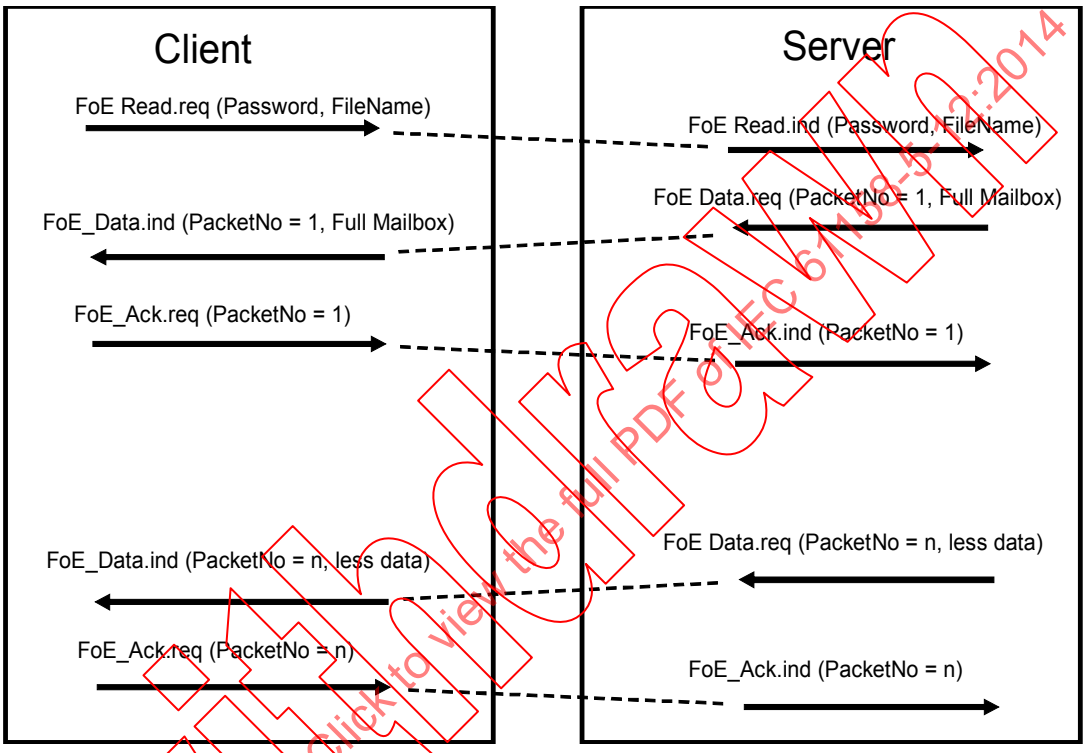


Figure 27 shows the primitives between client and server in case of a successful FoE Read sequence.

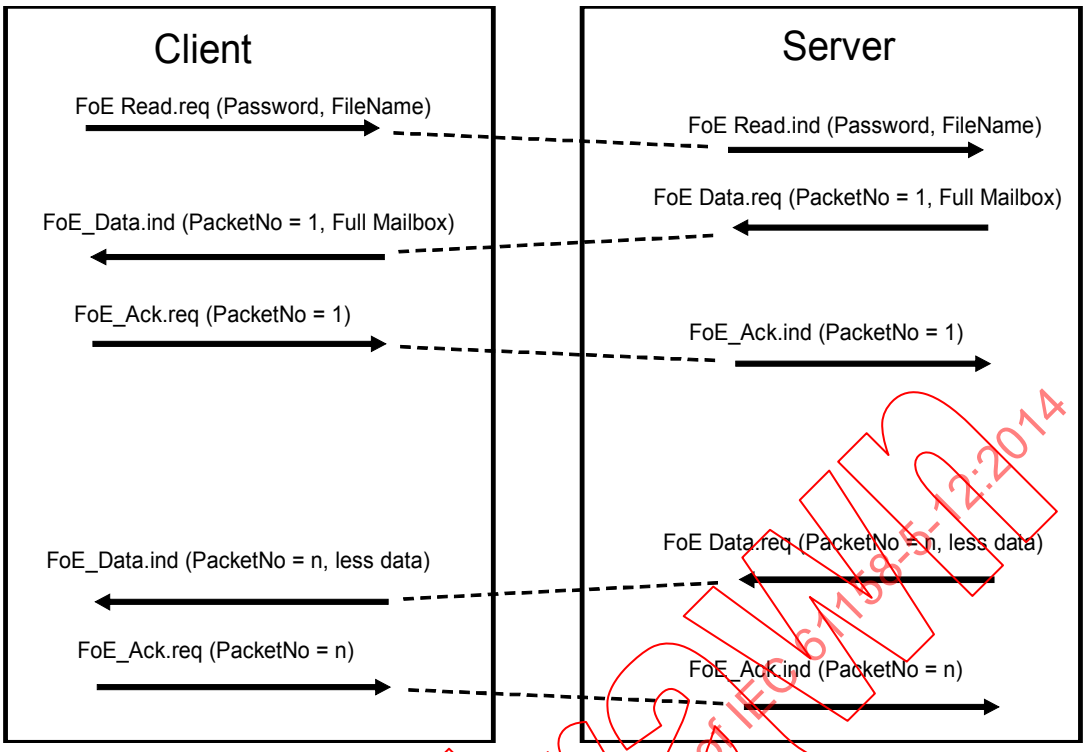


Figure 27 – FoE read sequence with success

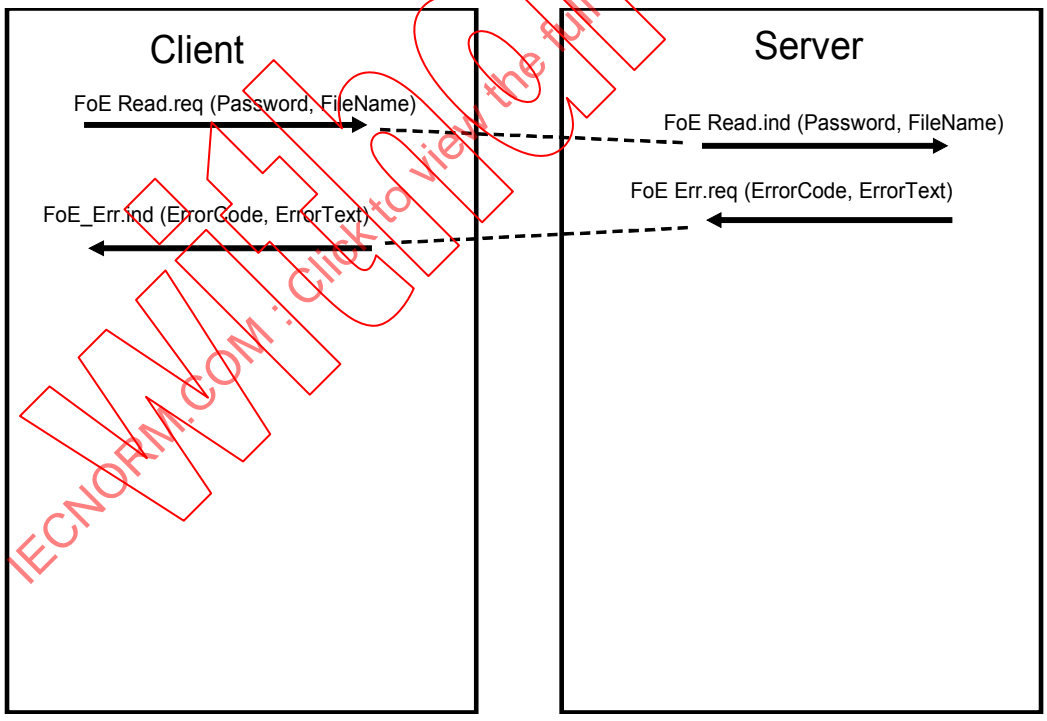


Figure 28 shows the primitives between client and server in case of an unsuccessful FoE Read sequence.

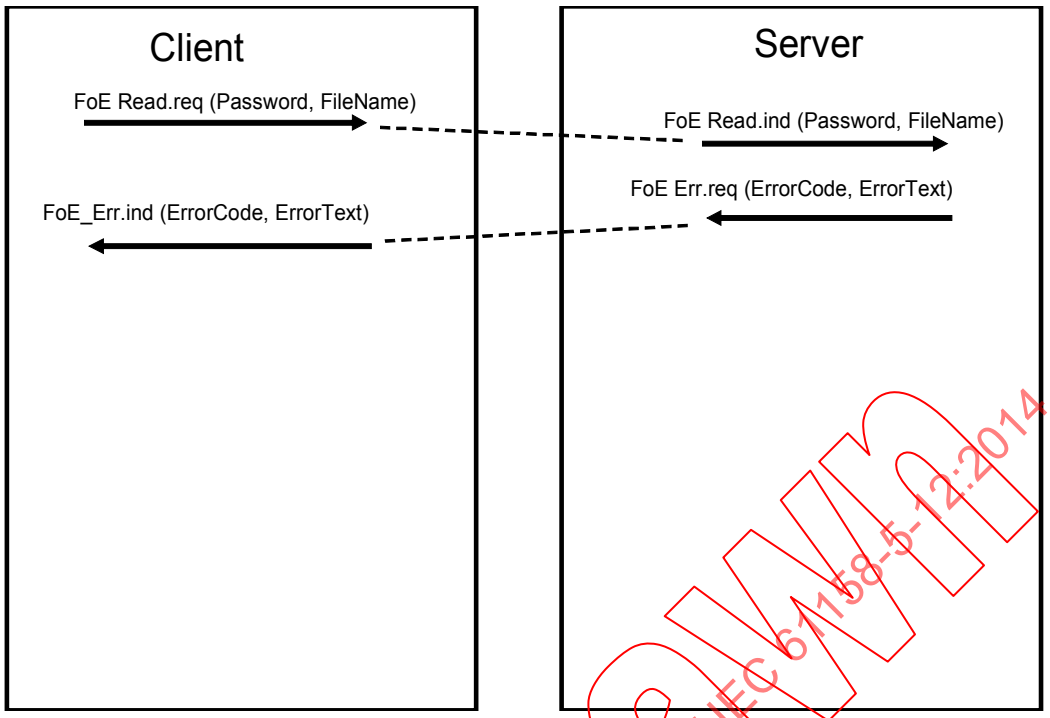


Figure 28 – FoE read sequence with error

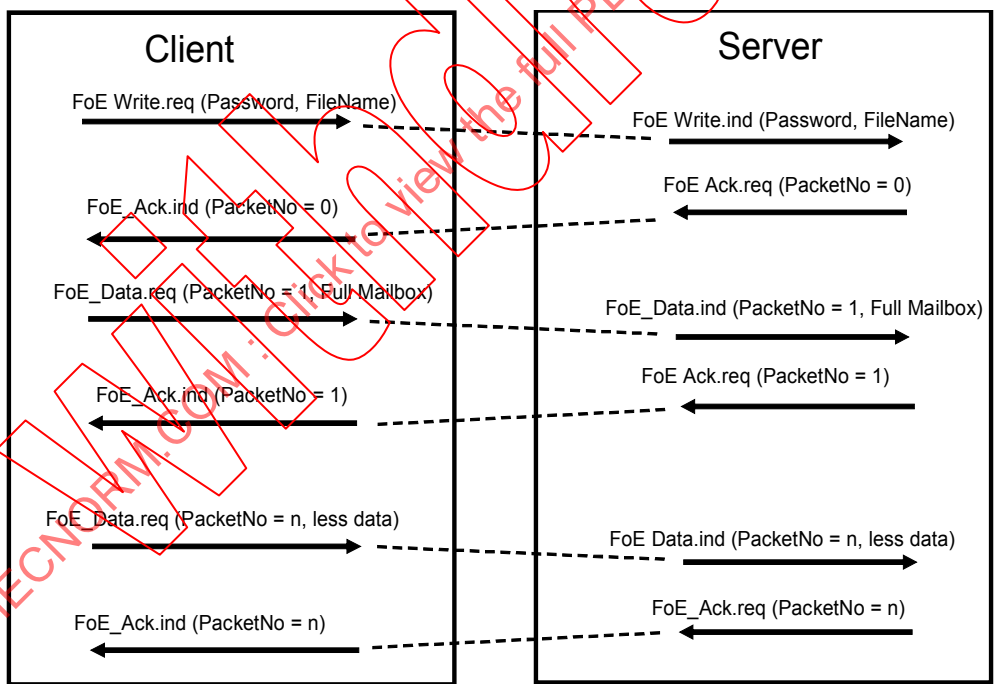


Figure 29 shows the primitives between client and server in case of a successful FoE Write sequence.

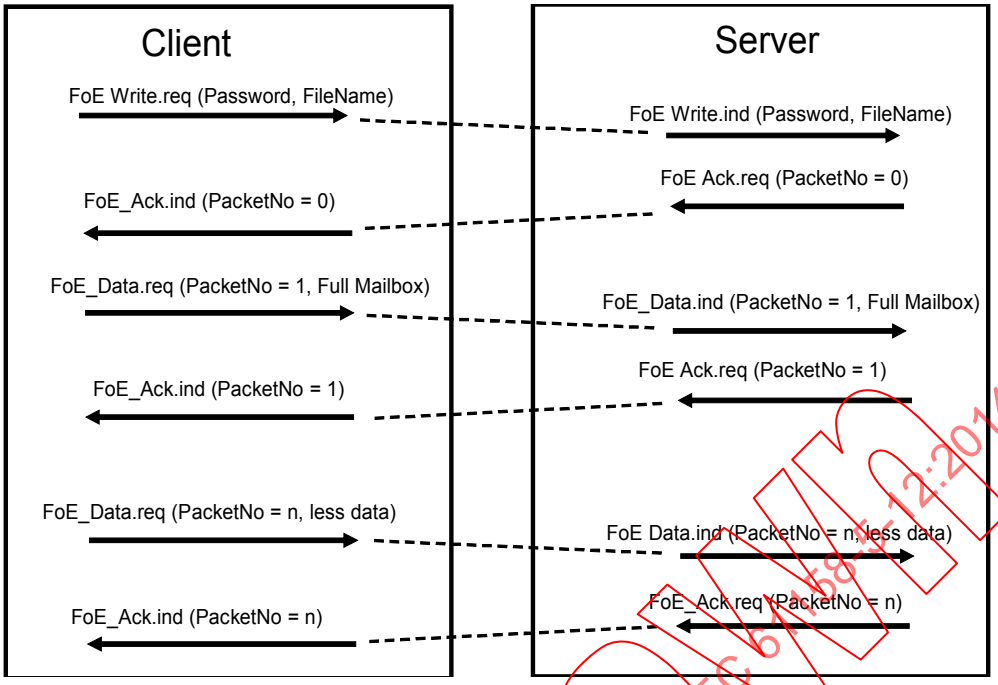


Figure 29 – FoE write sequence with success

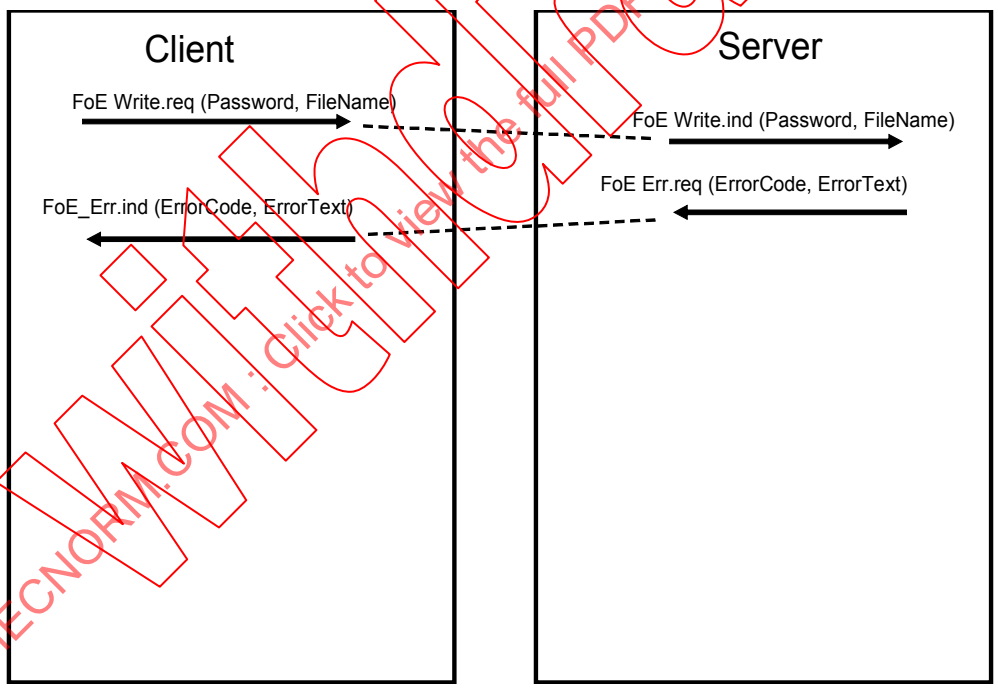


Figure 30 shows the primitives between client and server in case of an unsuccessful FoE Write sequence.

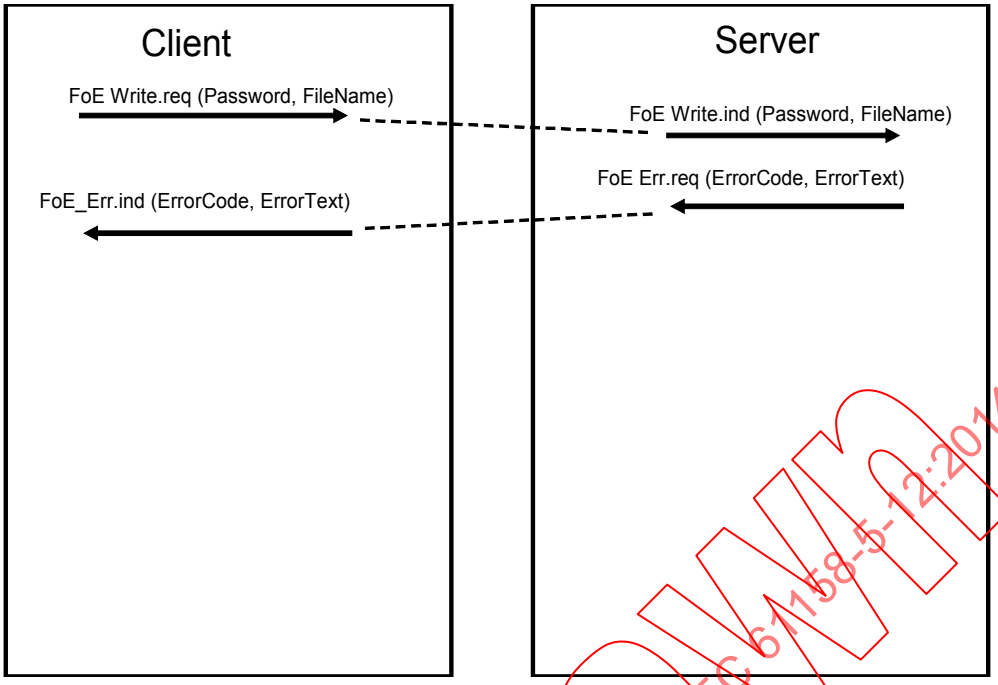


Figure 30 – FoE write sequence with error

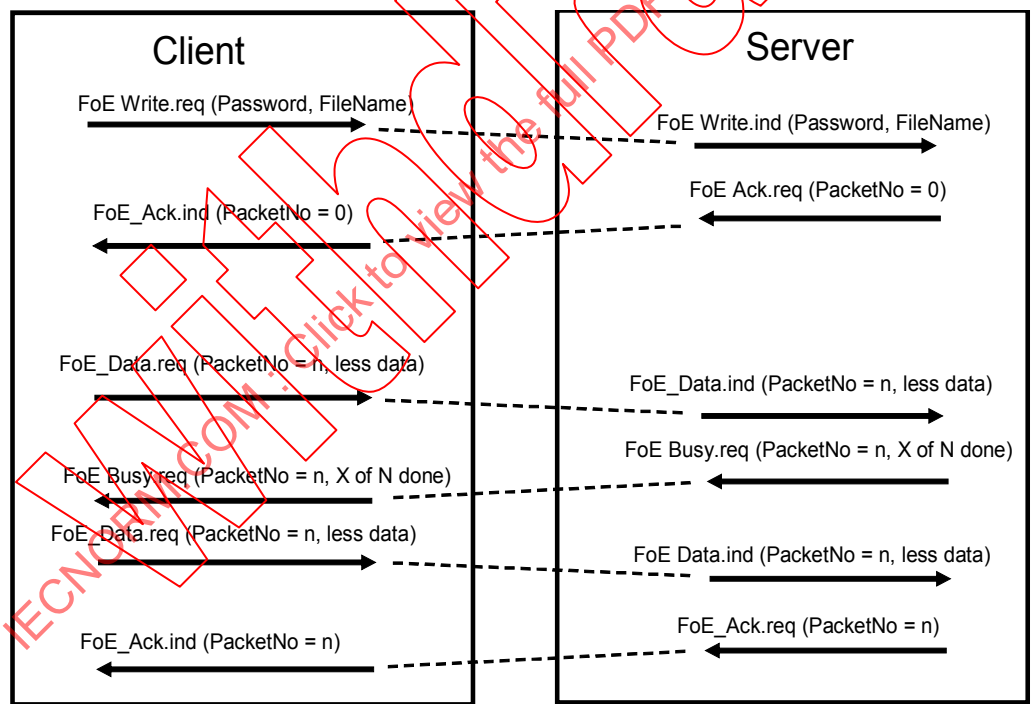


Figure 31 shows the primitives between client and server in case of a FoE Write sequence with a busy Server interaction.

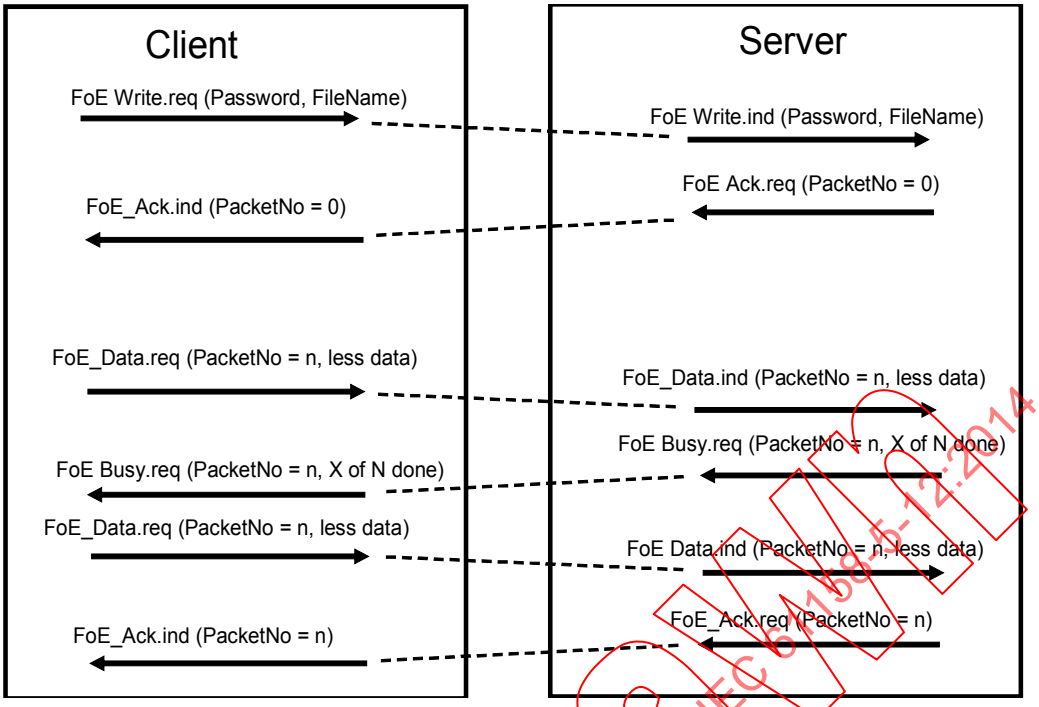


Figure 31 – FoE write sequence with busy

6.1.6.2 FoE class specification

6.1.6.2.1 Formal model

The FoE file object is described by the following template:

ASE: **FoE**
CLASS: **FoE File**
CLASS ID: not used
PARENT CLASS: TOP
ATTRIBUTES:
1. (m) Key Attribute: File Name
2. (o) Attribute: Password
3. (m) Attribute: Download Active
5. (m) Attribute: File Info

SERVICES:
1. (o) OpsService: FoE Read
2. (o) OpsService: FoE Write

6.1.6.2.2 Attributes

File name
This attribute specifies a file name to identify a file object.

Password
This optional attribute specifies a Password to prevent a file to be overwritten accidentally.

Download active
This Boolean attribute is set to true if a download is in progress and set to false otherwise.

File data
This attribute specifies a file data.

6.1.6.2.3 Services

FoE read
Initiates data transfer from the server to the client

FoE write
Initiates data transfer from the client to the server

FoE data
Conveys a piece of data

FoE ack
Acknowledge the conveyance of data

FoE busy
Reports a delay in delivery of data or acknowledgement

FoE error
Reports an error in an FoE service

6.1.6.3 FoE service specification

6.1.6.3.1 Supported services

The FoE ASE defines the services

FoE Read

FoE Write

6.1.6.3.2 FoE read

The FoE Read service is used to read a file from a server. Table 29 shows the service primitives and parameter of the FoE Read service.

Table 29 – FoE read

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Password	O	O(=)
File Name	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Password
This optional parameter specifies a password for the read access.

File name
This parameter indicates the name of the file to be read.

6.1.6.3.3 FoE write

The FoE Write service is used to write a file to a server. Table 30 shows the service primitives and parameter of the FoE Write service.

Table 30 – FoE write

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Password	O	O(=)
File Name	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Password
This optional parameter specifies a password for the write access.

File name
This parameter indicates the name of the file to be written.

6.1.6.3.4 FoE data

The FoE Data service is used to transmit the file data from the server in case of a read or from the client in case of a write. The first FoE Data always starts with a Packet Number of one incrementing with every following FoE Data. Table 31 shows the service primitives and parameter of the FoE Data service.

Table 31 – FoE data

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Packet Number	M	M (=)
Size	M	M (=)
Data	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Packet number
This parameter specifies a sequence number of the frame starting with 1 with the first FoE Data service of a sequence. The PacketNumber is used as a sequence number.

Size
This parameter indicates the number of octets to be transferred to the client. Its range is 0 to 1 472.

Data
This parameter specifies the object value which is to be transferred to the client.

6.1.6.3.5 FoE ack

The FoE Ack service is used to acknowledge a FoE Data. The Packet Number will be always the same as sent with the corresponding FoE Data before. A FoE Write will be acknowledged with a FoE Ack with a Packet Number of zero before the client starts with the first FoE Data. Table 32 shows the service primitives and parameter of the FoE Ack service.

Table 32 – FoE ack

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Packet Number	M	M (=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Packet number
This parameter specifies a sequence number of the frame starting with 1 with the first FoE Data service of a sequence. The PacketNumber is used as a sequence number.

6.1.6.3.6 FoE busy

The FoE Busy service is used to indicate the client, that the server is busy to store the written file data. The client will repeat the corresponding FoE Data until the server answers with a FoE Ack. Table 33 shows the service primitives and parameter of the FoE Busy.

Table 33 – FoE busy

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Done	M	M (=)
Entire	M	M (=)
Busy Detail	O	O(=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Done
This parameter specifies a percentage of the time elapsed before the FoE Ack can be issued. Its range is zero to 100 (percent).

Entire
This parameter specifies zero or the 100 percent value of done otherwise.

Busy detail
This optional parameter specifies detail information of busy.

6.1.6.3.7 FoE Error

The FoE Error service is used to indicate the client that an error has happened during file operation. Table 34 shows the service primitives and parameter of the FoE Error.

Table 34 – FoE error

Parameter name	Req	Ind
Argument		
Address	M	M (=)
Error Code	M	M (=)
Error Text	O	O(=)

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Error code
This parameter indicates the type of error.

Error text
This optional parameter provides an explanation of the error.

6.1.7 MBX ASE

6.1.7.1 Overview

The mailbox commands specify a standard way to access mailbox by profile standards.

6.1.7.2 MBX class specification

6.1.7.2.1 Formal model

The MBX object is described by the following template:

ASE: **MBX**
CLASS: **MBX**
CLASS ID: not used
PARENT CLASS: TOP
ATTRIBUTES:
1. (m) Key Attribute: implicit
2. (m) Attribute: Write area
3. (m) Attribute: Read area

SERVICES:
1. (m) OpsService: Mailbox Read
2. (m) OpsService: Mailbox Write
3. (m) OpsService: Mailbox Read Upd

6.1.7.2.2 Attributes

Write area
This attribute specifies a memory area to store data send from the client to the server.

Read area
This attribute specifies a memory area to store data send from the server to the client.

6.1.7.2.3 Services

Mailbox read
Initiates data transfer from the server to the client

Mailbox write
Initiates data transfer from the client to the server

Mailbox read upd
Updates Read area in the server.

6.1.7.3 MBX service specification

6.1.7.3.1 Supported services

The MBX ASE defines the services

MBX Read

MBX Write

MBX Read Upd

6.1.7.3.2 MBX read

The MBX Read service is used to read data from a server. It can be used for profiles to convey data from the slave to the master. Table 35 shows the service primitives and parameter of the MBX Read service.

Table 35 – MBX read

Parameter name	Req	Ind	Cnf
Argument			
Address	M	M (=)	
Channel	O	O(=)	
Priority	O	O(=)	
Type	M	M (=)	
Cnt	M	M (=)	
Result(+)			S
Data			M
Result(–)			S
Reason			M
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.			

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the master.

Channel
This optional parameter specifies a a channel number in the range 0 to 63

Priority
This optional parameter specifies one of four priorities

Type
This parameter specifies the type of the mailbox service.

Cnt
This parameter specifies a service counter

Result(+)
This selection type parameter indicates that the service request succeeded.

Data
This parameter specifies the data that was read

Result(–)
This selection type parameter indicates that the service request failed.

Reason
This parameter reports the reason of the failure. Possible values are

- No Slave Reaction
- No Reply Message

6.1.7.3.3 MBX write

The MBX Write service is used to write data to a server. It can be used for profiles to convey data from the master to the slave. Table 36 shows the service primitives and parameter of the MBX Write service.

Table 36 – MBX write

Parameter name	Req	Ind	Cnf
Argument			
Address	M	M (=)	
Channel	O	O(=)	
Priority	O	O(=)	
Type	M	M (=)	
Cnt	M	M (=)	
Length	M	M (=)	
Service Data	M	M (=)	
Result(+)			S
Result(–)			S
Reason			M

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Channel
This optional parameter specifies a channel number.

Allowed values: 0 to 63

Priority
This optional parameter specifies one of four priorities.

Type
This parameter specifies the type of the mailbox service.

Cnt
This parameter specifies a service counter

Length
This parameter specifies the length of the service data. Its range is 0 to 1 480.

Service data
This parameter specifies the user data.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

Reason
This parameter reports the reason for the failure. Possible values are

- No Slave Reaction
- No Reply Message

6.1.7.3.4 MBX read upd

The MBX Read Upd service is used to write data from the slave application to the mailbox read area. It can be used for profiles to convey data from the slave to the master in conjunction with the MBX Read service. Table 37 shows the service primitives and parameter of the MBX Read Upd service.

Table 37 – MBX read upd

Parameter name	Req	Ind	Cnf
Argument			
Address	M	M (=)	
Channel	O	O(=)	
Priority	O	O(=)	
Type	M	M (=)	
Cnt	M	M (=)	
Length	M	M (=)	
Service Data	M	M (=)	
Result(+)			S
Result(–)			S
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.			

Argument
The argument conveys the service specific parameters of the service request.

Address
This parameter specifies the station address of the server if the data are sent to the slave and the address of the master otherwise.

Channel
This optional parameter specifies a channel number. Its range is 0 to 63.

Priority
This optional parameter specifies one of four priorities.

Type
This parameter specifies the type of the mailbox service.

Cnt
This parameter specifies a service counter

Length
This parameter specifies the length of the user data. Its range is 0 to 1 480.

Service data
This parameter specifies the service data.

Result(+)
This selection type parameter indicates that the service request succeeded.

Result(–)
This selection type parameter indicates that the service request failed.

6.2 AR

6.2.1 Overview

6.2.1.1.1 General

There is only one AR endpoint in a Slave. This endpoint is controlled by the Type 12 State Machine (ESM) which describes the states and state changes of the slave application. The actual state of a slave application is reflected in the AL Status Register by the application and requested state changes are indicated in the AL Control Register by the master.

The ESM is logically located between the Type 12 slave controller (ESC) and the application.

The ESM defines four states:

- Init,
- Pre-Operational,
- Safe-Operational, and
- Operational.

All state changes are possible except for the 'Init' state, where only the transition to the 'Pre-Operational' state is possible and for the 'Pre-Operational' state, where no direct state change to 'Operational' exists.

State changes are normally requested by the master. The master requests a write to the AL Control register which results in a Register Event 'AL Control' indication in the slave. The slave responds to the change in AL Control through a local AL Status write service after a successful or a failed state change. If the requested state change failed, the slave responds with the error flag set.

The Bootstrap state is optional and there is only a transition from or to the Init state. The only purpose of this state is to download the device's firmware. In Bootstrap state the mailbox is active but restricted to the file access over Type 12 services (FoE) protocol.

6.2.1.2 State services

6.2.1.2.1 Service overview

The primitives of the Type 12 DL-services are mapped to the AL management primitives described in the DL as specified in Table 38.

If the slave does not support mailbox services, the slave controller should be configured from the master to confirm the state change immediately in the AL Status register.

Table 38 – AL management and ESM service primitives

AL management primitives	master service	slave service
AL Control.req/ind	Write (AL Control) – all types of write or RW allowed	Event(AL Control), ReadLocal(AL Control)
AL Control.rsp/cnf	Read (AL Status) – all types of read or RW allowed	WriteRegisterLocal(AL Status)
AL State Changed.req/ind	Read (AL Status) – all types of read or RW allowed	WriteRegisterLocal(AL Status)

6.2.1.2.2 AL control sequence

The primitives between master and slave in case of a successful AL Control sequence are specified in

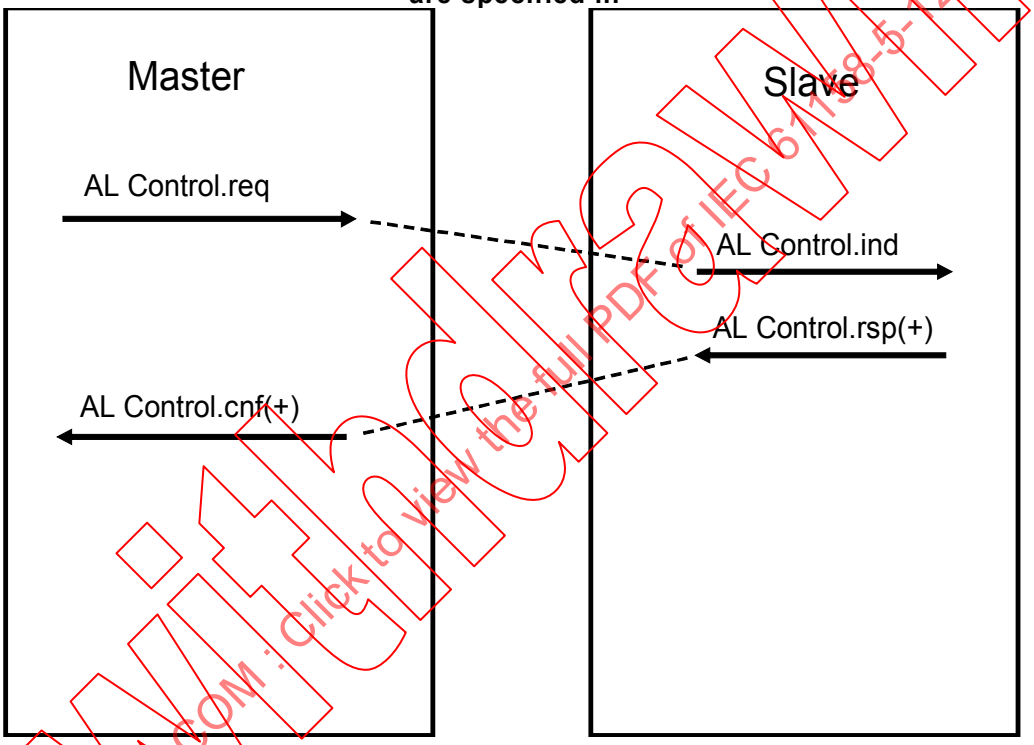


Figure 32.

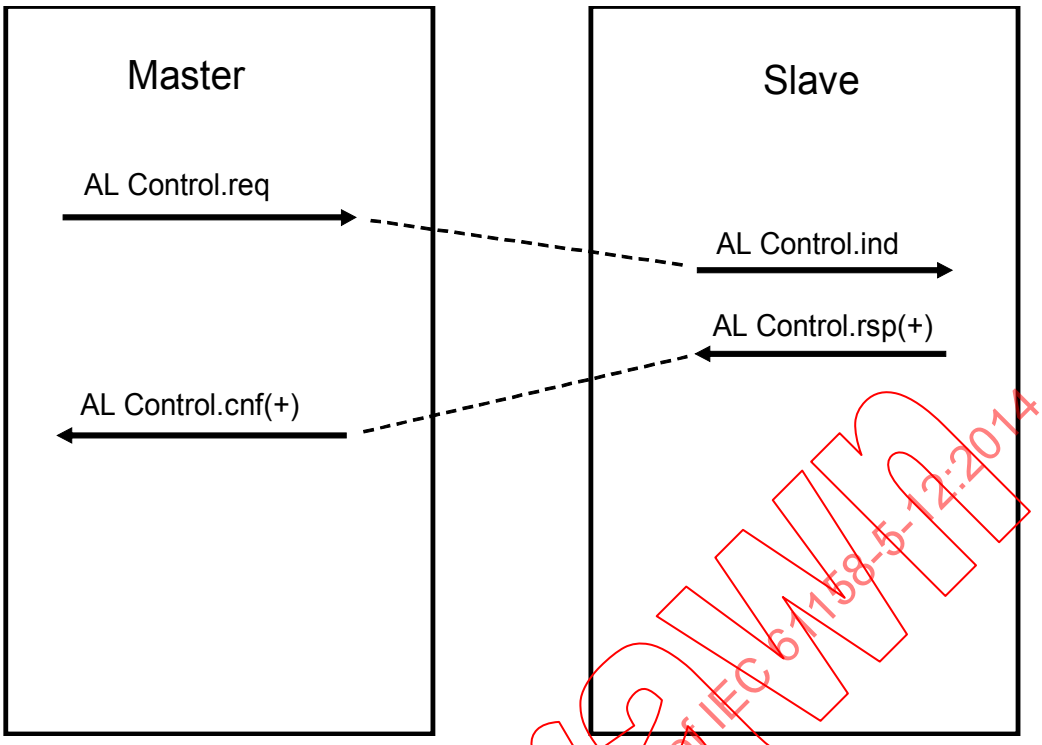


Figure 32 – Successful AL control sequence

The primitives between master and slave in case of a unsuccessful AL Control sequence are specified in

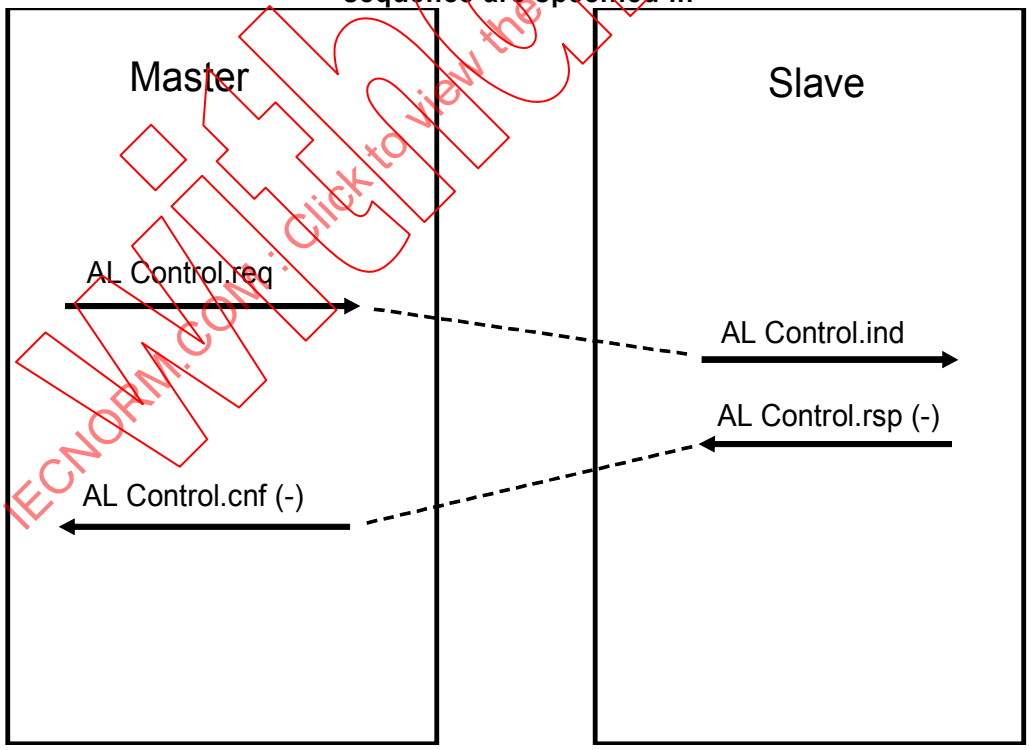


Figure 33.

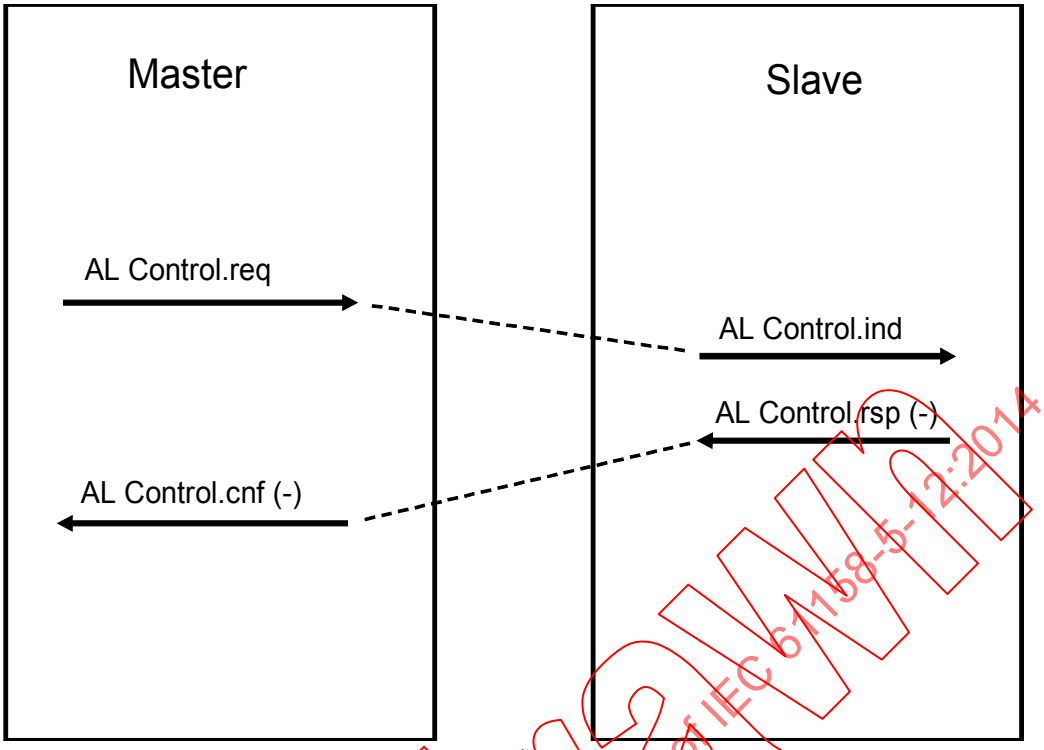


Figure 33 – Unsuccessful AL control sequence

6.2.1.2.3 AL state changed sequence

The primitives between master and slave in case of an AL State Changed sequence are specified in

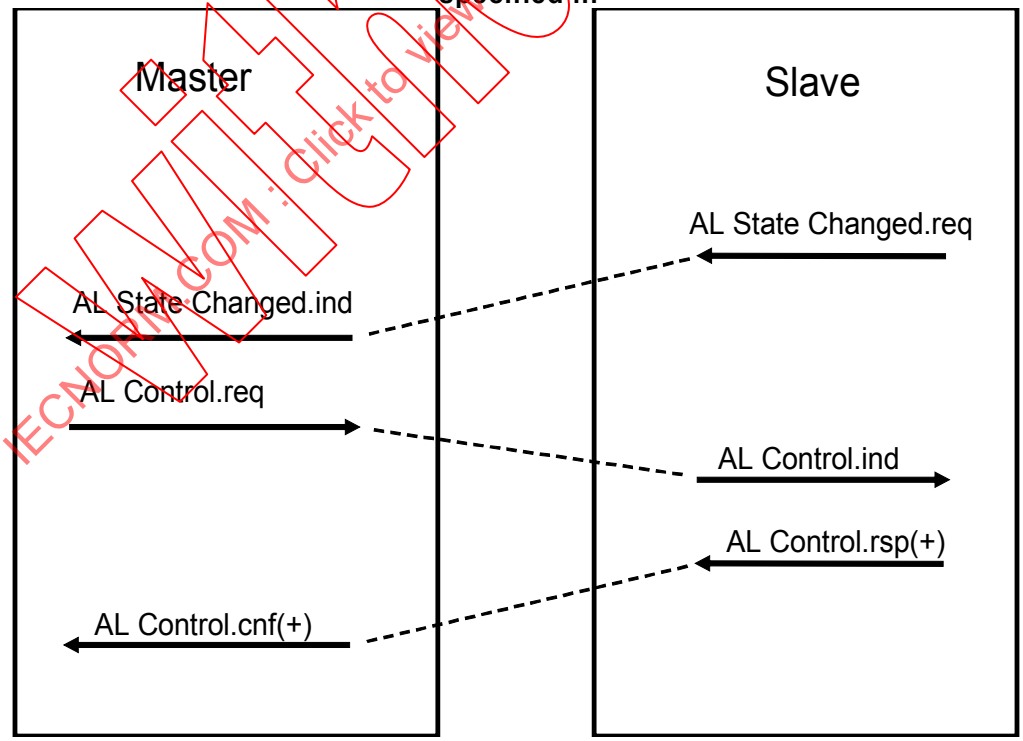


Figure 34.

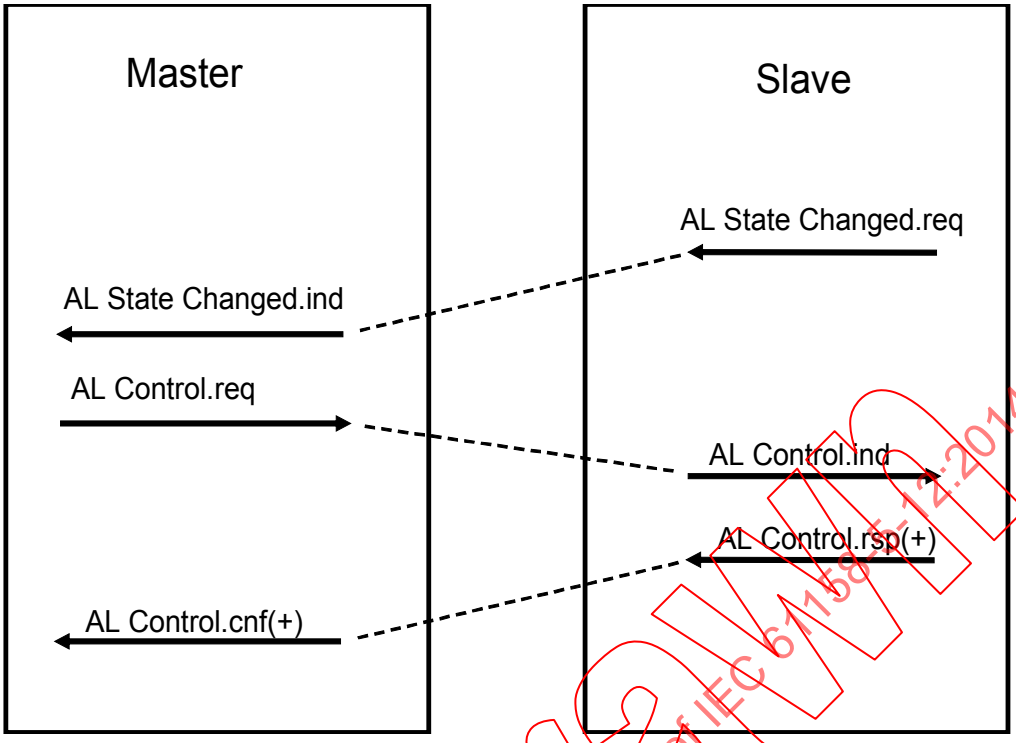


Figure 34 – AL state changed sequence

6.2.1.3 Local management services

6.2.1.3.1 Start mailbox communication

The Start Mailbox Communication Event will be issue if in the state 'Init' a valid AL Control service and requested AL Control State 'Pre-Operational' is indicated.

The master should configure the DL registers and the Sync Manager channels for the mailbox before requesting this service. Then the master should wait for the confirmation of the slave before sending other commands to the slave.

If the slave supports mailbox services, the slave's application should read the settings of the Sync Manager and initialize its mailbox handler appropriately before confirming the state change by writing the AL Status register of the slave controller. If the Sync Manager's configuration is correct, the slave provides a positive confirmation to this service.

6.2.1.3.2 Stop mailbox communication

The Stop Mailbox Communication Event will be issue if the slave application will enter the state 'Init' from 'Pre-Operational', 'Safe-Operational' or 'Operational'.

If the slave supports mailbox services, the slave's application has to stop its mailbox handler before issuing the state change by writing the AL Status register of the slave controller. The slave always confirms this service as successful.

6.2.1.3.3 Start input update

The Start Input Update service will be issued on a valid AL Control service and requested AL Control State the 'Safe-Operational' state, if the slave's application is in the state 'Pre-Operational'.

The master should configure the Sync Manager for the process data and the FMMU before requesting this service. After requesting the state transition the master should begin the

transmission of the process data services, but should ignore the input data until the slave has confirmed the state transition.

If the slave supports mailbox services, the slave's application reads the configuration of the Sync Manager channels configured for process data transfer and checks if these settings match to its local process data configuration. If the checking was successful, the slave's application delivers valid input data before confirming the state transition. The output data of the slave should stay in a safe state. If the checking of the Sync Manager configuration was unsuccessful, the slave should confirm this service with the 'Pre-Operational' state and set the Error Flag parameter TRUE.

Start Input Update behavior is supported by complex slaves.

6.2.1.3.4 Stop input update

The Stop input update Event will be issue if the slave application will enter the state 'Pre-Operational' or 'Init' from 'Safe-Operational' or 'Operational'.

The master should stop transmission of process data requests before requesting the state transition.

If the slave supports mailbox services, the slave's application should stop updating the input data before confirming the state transition. The slave always confirms this service as successful.

Stop Input Update behavior is supported by complex slaves.

The slave's application can use this service to indicate a local state transition, for example because of an unexpected error.

6.2.1.3.5 Start output update

The Start Output Update service will be issued on a valid AL Control service and requested AL Control State the 'Operational' state, if the slave's application is in the state 'Safe-Operational'.

The master should deliver valid output data in the process data services before requesting the state transition.

Start Output Update behavior is supported by complex slaves.

The slave's application should activate the valid output data received with the process data service before confirming the state transition. If the activation of the output data is not possible, the slave should confirm this service with the 'Safe-Operational' state and set the Error Flag parameter TRUE.

6.2.1.3.6 Stop output update

The Stop output update Event will be issue if the slave application will leave the state 'Operational'.

If the slave supports mailbox services, the slave's application should set the output in the safe state before confirming the state transition. The slave always confirms this service as successful.

Stop Output Update behavior is supported by complex slaves.

The slave's application can use this service to indicate a local state transition, for example because of an unexpected error.

6.2.1.3.7 Start bootstrap mode

The Start Mailbox Communication Event will be issue if in the state 'Init' a valid AL Control service and requested AL Control State 'Bootstrap' is indicated.

The master should configure the Sync Manager channels for the mailbox before requesting this service. The Sync Manager configuration for the mailbox in Bootstrap state can differ from the configuration in the other states. Then the master should wait for the confirmation of the slave before sending other commands to the slave.

The slave's application should read the settings of the Sync Manager and initialize its mailbox handler appropriately before confirming the state change by writing the AL Status register of the slave controller. If the slave's application supports the Bootstrap state and the configuration of the Sync Manager are correct for the Bootstrap state, the slave will confirm this service as successful.

6.2.1.3.8 Stop bootstrap mode

The Stop Mailbox Communication Event will be issue if the slave application will re-enter the state 'Init'.

If the slave supports mailbox services, the slave's application has to stop its bootstrap handler before confirming the state change by writing the AL Status register of the slave controller. The slave always confirms this service as successful.

If the slave does not support mailbox services, the slave controller should be configured from the master to confirm the state change immediately in the AL Status register.

6.2.2 AR control class specification

6.2.2.1 Formal model

The AR control object is described by the following template:

ASE:			**AR**
CLASS:			**AR control**
CLASS ID:			not used
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Implicit
2.	(m)	Attribute:	AL Control
2.1	(m)	Attribute:	AL State
2.2	(m)	Attribute:	Acknowledge
2.3	(o)	Attribute:	State info
3.	(m)	Attribute:	AL Status
3.1	(m)	Attribute:	AL State
3.2	(m)	Attribute:	Error Indication
3.3	(o)	Attribute:	ID Indication
3.4	(o)	Attribute:	State info
3.5	(m)	Attribute:	AL Status Code
4.	(m)	Attribute:	PDI Control
4.1	(m)	Attribute:	PDI Type
4.2	(m)	Attribute:	AL State Control
4.3	(o)	Attribute:	Specific settings
5.	(o)	Attribute:	Sync Control
5.1	(o)	Attribute:	Signal conditioning Sync 0
5.2	(o)	Attribute:	Enable Sync 0
5.3	(o)	Attribute:	Signal conditioning Sync 1
5.4	(o)	Attribute:	Enable Sync 1
6.	(m)	Attribute:	Events
6.1	(m)	Attribute:	AL control write event
6.2	(o)	Attribute:	Latch event
6.3	(o)	Attribute:	Sync 0 event
6.4	(o)	Attribute:	Sync 1 event
6.5	(m)	Attribute:	List of SM event
6.5.1	(m)	Attribute:	SM event
6.6	(o)	Attribute:	AL control write event mask
6.7	(o)	Attribute:	Latch event mask
6.8	(o)	Attribute:	Sync 0 event mask
6.9	(o)	Attribute:	Sync 1 event mask
6.10	(o)	Attribute:	List of SM event mask
6.10.1	(o)	Attribute:	SM event mask
7.	(o)	Attribute:	SM Settings
7.1	(o)	Attribute:	SM0
7.2	(o)	Attribute:	SM1
7.1	(o)	Attribute:	SM2
7.2	(o)	Attribute:	SM3
SERVICES:			
1.	(m)	OpsService:	AL Control
2.	(m)	OpsService:	AL State Change

NOTE The attribute are read/written by DL-services as described in IEC 61158-3-12 as this ASE will be the agent for management services.

6.2.2.2 Attributes

Implicit

The attribute Implicit indicates that the AR Control object is implicitly addressed by the services.

AL Control
One Object is composed of the following elements:

AL state
This parameter specifies the requested AL state. Possible values are

- Init
- Pre-operational
- Safe-operational
- Operational
- Bootstrap

Acknowledge
This Boolean attribute is set to confirm an error indication of the AL.

ID Request
This Boolean attribute is set to request the Identification value of the AL.

State info
This optional attribute consists of application-specific information set by the master.

AL Status
One Object is composed of the following elements:

AL state
This attribute contain the AL State set by the local instance. Possible values are

- Init
- Pre-operational
- Safe-operational
- Operational
- Bootstrap

Error indication
This Boolean attribute consists of the information that the state control of the AL has detected an error.

ID indication
This Boolean attribute consists of the information that the AL has loaded an ID value

State info
This optional attribute consists of application-specific information set by the slave application.

AL status code
This optional attribute is controlled by the application and reports the last error detected by the state control instance of the AL or an ID value.

PDI Control
One Object is composed of the following elements:

PDI type
This attribute specifies the process data interface type describing the hardware interface behaviour of the slave controller.

AL state control
This Boolean attribute is set to false if the AL Status operation is done by the application and is set to true if the AL Status will be emulated by the slave controller by copying the AL control into the AL status.

Specific settings
This optional attribute consists of application-specific information set by the master.

PDI Control
One Object is composed of the following elements:

Signal conditioning sync 0
This optional attribute specifies the process data interface type describing the hardware interface behaviour of the slave controller's sync signal.

Enable sync 0
This optional attribute enables the synchronisation signal. Two events sources can be selected. Allowed values are

- Interrupt
- Pulse
- Both
- None

Signal conditioning sync 1
This optional attribute specifies the process data interface type describing the hardware interface behaviour of the slave controller's sync signal.

Enable sync 1
This optional attribute enables the synchronisation signal. Two events sources can be selected. Allowed values are

- Interrupt
- Pulse
- Both
- None

Events
One Object is composed of the following elements:

AL control write event
This Boolean attribute is set if there is a write from the master to AL control. It is cleared if the AL control is read locally.

Latch event
This optional Boolean attribute is set if there is a new entry in the latch registers. It is cleared if the latch registers are read locally.

Sync 0 event
This optional Boolean attribute is set if there is a Sync 0 event. It is cleared if the event is read locally.

Sync 1 event
This optional Boolean attribute is set if there is a Sync 1 event. It is cleared if the event is read locally.

List of SM event
One Object is composed of the following elements:

SM event
This Boolean attribute is set if there is valid read or write to the SM area. It is cleared if the event is read locally.

AL control write event mask
This Boolean attribute is set if a write from the master to AL control should produce an event. It is cleared otherwise.

Latch event mask
This optional Boolean attribute is set if a new entry in the latch registers should result in an event. It is cleared otherwise.

Sync 0 event mask
This optional Boolean attribute is set if a Sync 0 event should result in an event. It is cleared otherwise.

Sync 1 event mask
This optional Boolean attribute is set if a Sync 1 event should result in an event. It is cleared otherwise.

List of SM event mask
One Object is composed of the following elements:

SM event mask
This Boolean attribute is set if valid read or write to the SM area should result in an event. It is cleared otherwise.

SM settings
One Object is composed of the following elements:

SM 0
This optional attribute consists of the SM settings as described in IEC 61158-4-12. SM0 is used as write mailbox.

SM 1
This optional attribute consists of the SM settings as described in IEC 61158-4-12. SM1 is used as read mailbox.

SM 2
This optional attribute consists of the SM settings as described in IEC 61158-4-12. SM2 is used as write buffer.

SM 3
This optional attribute consists of the SM settings as described in IEC 61158-4-12. SM3 is used as read buffer.

6.2.3 AR service specification

6.2.3.1 AL control

The AL Control service, specified in Table 39, is used by the master to request a state transition in the slave's application by writing the AL Control register. The slave confirms the state transition by writing the AL Status register, which will be read from the master to get the confirmation.

In case of an unsuccessful state transition, the master acknowledges this service with another AL Control service and Ack Flag set to one.

With an ID Request the master requests the ID value from the slave's application. The slave confirms the successful loading of the ID value to the AL Status Code register by setting the ID Flag to one.

Table 39 – AL control

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AL State	M	M (=)		
Ack Flag	M	M (=)		
ID Request	U	U (=)		
Result(+)			S	S (=)
AL Status Code			M	M (=)
Error Indication			M	M (=
ID Indication			U	U (=)
Result(–)			S	S (=)
AL Status Code			M	M (=)
Error Indication			M	M (=)
ID Indication			U	U (=)

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. See 1.2.

Argument
The argument conveys the service specific parameters of the service request.

AL state
This parameter indicates the state which is requested by the master in the slave's application. Possible values are

- Init
- Pre-operational
- Safe-operational
- Operational
- Bootstrap

Ack Flag
This Boolean parameter acknowledges a negative result of an AL Control service.

ID Request
This Boolean parameter indicates an ID request.

Result(+)
This selection type parameter indicates that the service request succeeded.

AL status code
This parameter indicates the actual state of the slave's application or the ID value.

Error Indication
This parameter indicates that the state control of the AL has detected an error.

ID Indication
This parameter indicates the AL has loaded an ID value.

Result(–)
This selection type parameter indicates that the service request failed.

AL status code
This parameter indicates the actual state of the slave's application.

Error Indication
This parameter indicates that the state control of the AL has detected an error.

ID Indication
This parameter indicates the AL has loaded an ID value.

6.2.3.2 AL state change

Additionally the slave's application can indicate a local state transition by writing the AL Status register as specified in Table 40. The master acknowledges the state transition with AL Control service.

Table 40 – AL state change

Parameter name	Req	Ind
Argument		
AL State	M	M (=)
AL Status Code	M	M (=)
Error Indication	M	M (=)
ID Indication	U	U (=)

Argument
The argument conveys the service specific parameters of the service request.

AL state
This parameter indicates the state which is requested by the slave's application. Possible values are

- Init
- Pre-operational
- Safe-operational
- Operational
- Bootstrap

AL status code
This parameter indicates the actual state of the slave's application.

Error Indication
This parameter indicates that the state control of the AL has detected an error.

ID Indication
This parameter indicates the AL has loaded an ID value.

Bibliography

IEC 61158-2, *Industrial communication networks – Fieldbus specifications – Part 2: Physical layer specification and service definition*

IEC 61158-4-12, *Industrial communication networks – Fieldbus specifications – Part 4-12: Data-link layer protocol specification – Type 12 elements*

IEC 61158-6-12, *Industrial communication networks – Fieldbus specifications – Part 6-12: Application layer protocol specification – Type 12 elements*

IEC 61784-1, *Industrial communication networks – Profiles – Part 1: Fieldbus profiles*

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

IEEE 802.1Q, *IEEE standard for Local and metropolitan area networks – Virtual bridged local area networks Bridges;* available at <http://www.ieee.org>

EN 50325-4, *Industrial communications subsystem based on ISO 11898 (CAN) for controller-device interfaces – Part 4: CANopen*

SOMMAIRE

AVANT-PROPOS 133
INTRODUCTION 135
1 Domaine d'application 136
1.1 Généralités 136
1.2 Spécifications 137
1.3 Conformité 137
2 Références normatives 137
3 Termes, définitions, symboles, abréviations et conventions 138
3.1 Termes et définitions du modèle de référence 138
3.2 Termes et définitions de convention pour les services 139
3.3 Termes et définitions pour les services de la couche application et de liaison de données 139
3.4 Symboles et abréviations communs 144
3.5 Conventions 145
4 Concepts 146
4.1 Concepts communs 146
4.2 Concepts spécifiques de type 146
5 ASE "Data type" 156
5.1 Généralités 156
5.2 Définition formelle des objets "types de données" 156
5.3 Types de données définis pour la FAL 156
5.4 Spécification de service de l'ASE "Data type" 169
6 Spécification de modèle de communication 169
6.1 Les ASE 169
6.2 AR 254
Bibliographie 267

Figure 1 – Modèle producteur-consommateur 148
Figure 2 – Modèle client-serveur 148
Figure 3 – Invocation déclenchée par le serveur 149
Figure 4 – Modèle de référence d'esclave 151
Figure 5 – Appareil esclave simple 152
Figure 6 – Appareil esclave complexe 153
Figure 7 – Vue d'ensemble fonctionnelle du maître 154
Figure 8 – Séquence de données de sortie de processus 170
Figure 9 – Séquence de données d'entrée de processus 171
Figure 10 – Modèle de serveur CoE 192
Figure 11 – Séquence d'un seul SDO-Download qui a réussi 198
Figure 12 – Séquence d'un seul SDO-Download qui a échoué 199
Figure 13 – Séquence d'un SDO-Download segmenté qui a réussi 201
Figure 14 – Séquence d'un seul SDO-Upload qui a réussi 201
Figure 15 – Séquence d'un seul SDO-Upload qui a échoué 202

Figure 16 – Séquence d'un SDO-Upload segmenté qui a réussi 204
Figure 17 – Séquence d'informations de SDO 204
Figure 18 – Service 'Emergency' (Urgence) 205
Figure 19 – Séquence de "Command" 206
Figure 20 – Mapping de PDO 207
Figure 21 – Sync manager PDO assignment (Assignation des PDO au gestionnaire de synchronisation) 208
Figure 22 – Service RxPDO 210
Figure 23 – Service TxPDO 211
Figure 24 – Séquence "RxPDO remote transmission" (émission distante RxPDO) 212
Figure 25 – Séquence "TxPDO remote transmission" (émission distante TxPDO) 213
Figure 26 – Séquence "EoE" 234
Figure 27 – Séquence "FoE read" avec succès 242
Figure 28 – Séquence "FoE read" avec erreir 243
Figure 29 – Séquence "FoE write" avec succès 243
Figure 30 – Séquence "FoE write" avec erreur 244
Figure 31 – Séquence "FoE write" avec occupation 245
Figure 32 – Séquence de commande d'AL qui a réussi 255
Figure 33 – Séquence de commande d'AL qui a échoué 256
Figure 34 – Séquence d'état changé d'AL 257

Tableau 1 – Process output data 174
Tableau 2 – Process input data 175
Tableau 3 – Update process input data 176
Tableau 4 – SII read 186
Tableau 5 – SII write 187
Tableau 6 – SII reload 188
Tableau 7 – Allocation des zones de SDO 193
Tableau 8 – SDO download expedited 217
Tableau 9 – SDO download normal 218
Tableau 10 – Download SDO segment 219
Tableau 11 – SDO upload expedited 220
Tableau 12 – SDO upload normal 221
Tableau 13 – Upload SDO segment 222
Tableau 14 – Abort SDO transfer 223
Tableau 15 – Get OD list 224
Tableau 16 – OD list segment 225
Tableau 17 – Get object description 226
Tableau 18 – Get entry description 227
Tableau 19 – Object entry segment 229
Tableau 20 – Emergency 230
Tableau 21 – RxPDO 230
Tableau 22 – TxPDO 231

Tableau 23 – RxPDO remote transmission 232
Tableau 24 – TxPDO remote transmission 232
Tableau 25 – Initiate EoE 237
Tableau 26 – EoE fragment 238
Tableau 27 – Set IP parameter 239
Tableau 28 – Set address filter 241
Tableau 29 – FoE read 246
Tableau 30 – FoE write 247
Tableau 31 – FoE data 247
Tableau 32 – FoEack 248
Tableau 33 – FoE busy 248
Tableau 34 – FoE error 249
Tableau 35 – MBX read 251
Tableau 36 – MBX write 252
Tableau 37 – MBX read upd 253
Tableau 38 – Primitives de de gestion d'AL et de services ESM 255
Tableau 39 – AL control 265
Tableau 40 – AL state change 266

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 5-12: Définition des services de la couche application – Éléments de type 12

AVANT-PROPOS

1) La Commission Electrotechnique Internationale (CEI) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, la CEI – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de la CEI"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec la CEI, participent également aux travaux. La CEI collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
2) Les décisions ou accords officiels de la CEI concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de la CEI intéressés sont représentés dans chaque comité d'études.
3) Les Publications de la CEI se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de la CEI. Tous les efforts raisonnables sont entrepris afin que la CEI s'assure de l'exactitude du contenu technique de ses publications; la CEI ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de la CEI s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de la CEI dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de la CEI et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
5) La CEI elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de la CEI. La CEI n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
7) Aucune responsabilité ne doit être imputée à la CEI, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de la CEI, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de la CEI ou de toute autre Publication de la CEI, ou au crédit qui lui est accordé.
8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
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 brevet. 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.

L'attention est attirée sur le fait que l'utilisation du type de protocole associé est restreinte par les détenteurs des droits de propriété intellectuelle. En tout état de cause, l'engagement de renonciation partielle aux droits de propriété intellectuelle pris par les détenteurs de ces droits autorise l'utilisation d'un type de protocole de couche avec les autres protocoles de couche du même type, ou dans des combinaisons avec d'autres types autorisées explicitement par les détenteurs des droits de propriété intellectuelle pour ce type.

NOTE Les combinaisons de types de protocoles sont spécifiées dans la CEI 61784-1 et la CEI 61784-2.

La Norme internationale CEI 61158-5-12 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 troisième édition annule et remplace la deuxième édition parue en 2010. Cette édition constitue une révision technique. Les principales modifications par rapport à l'édition précédente sont énumérées ci-dessous:

- réparations des erreurs;
- améliorations éditoriales;
- prise en charge de l'Identification Explicite d'Appareils ajoutée dans l'ESM (voir 6.2.2)

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
65C/763/FDIS	65C/773/RVD

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette norme.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 2.

Une liste de toutes les parties de la série CEI 61158, publiées sous le titre général *Réseaux de communication industriels – Spécifications de bus de terrain*, peut être consultée sur le site web de la CEI.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de la CEI sous "http://webstore.iec.ch" dans les données relatives à la publication recherchée. À cette date, la publication sera

- reconduite;
- supprimée;
- remplacée par une édition révisée, ou
- amendée.

INTRODUCTION

La présente partie de la CEI 61158 est l'une d'une série produite pour faciliter l'interconnexion des composants d'un système d'automation. Elle est liée à d'autres normes de la série telle que définie par le modèle de référence des bus de terrain "à trois couches" décrit dans la CEI 61158-1.

Le service application est fourni par le protocole d'application utilisant les services disponibles de la liaison de données ou autre couche immédiatement inférieure. La présente norme définit les caractéristiques de services d'application pouvant être exploitées par les applications de bus de terrain et/ou la gestion de système.

Dans toute la série de normes relatives aux bus de terrain, le terme “service” se réfère à la capacité abstraite fournie par une couche du Modèle de référence de base de l'Interconnexion des systèmes ouverts (OSI) à la couche immédiatement supérieure. Ainsi, le service de la couche application défini dans la présente norme est un service architectural conceptuel, indépendant des divisions administratives et de mise en œuvre.

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 5-12: Définition des services de la couche application – Éléments de type 12

1 Domaine d'application

1.1 Généralités

La couche application de bus de terrain (FAL «Fieldbus Application Layer») fournit aux programmes d'utilisateur un moyen d'accéder à l'environnement de communication du bus de terrain. À cet égard, la FAL peut être vue comme une «fenêtre entre des programmes d'application correspondants».

La présente norme fournit les éléments communs pour les communications de messagerie de base à temps critique et à temps non critique entre des programmes d'application dans un environnement d'automation et le matériau spécifique au bus de terrain de Type 12. Le terme "à temps critique" sert à représenter la présence d'une fenêtre temporelle, dans les limites de laquelle une ou plusieurs actions spécifiées sont tenues d'être parachevées avec un certain niveau défini de certitude. Le manquement à parachever les actions spécifiées dans les limites de la fenêtre temporelle risque d'entraîner la défaillance des applications qui demandent ces actions, avec le risque concomitant pour l'équipement, l'installation et éventuellement pour la vie humaine.

La présente norme définit de manière abstraite le service visible de l'extérieur fourni par les différents types de a couche application de bus de terrain en termes

a) d'un modèle abstrait pour définir des ressources (objets) d'application capables d'être manipulées par les utilisateurs par l'intermédiaire de l'utilisation du service FAL;
b) des actions et événements primitifs du service;
c) des paramètres associés à chaque action primitive et événement primitif, et la forme qu'ils prennent; et
d) l'interrelation entre ces actions et événements, et leurs séquences valides.

Le but de la présente norme est de définir les services fournis à

a) l'utilisateur de FAL à la frontière entre l'utilisateur et la Couche application du Modèle de référence de bus de terrain; et
b) la Gestion des systèmes au niveau de la frontière entre la Couche application et la Gestion des systèmes selon le Modèle de référence de bus de terrain.

La présente norme spécifie la structure et les services de la couche application des bus de terrain de la CEI, en conformité avec le Modèle de référence de base de l'OSI (ISO/CEI 7498) et la Structure de la couche application de l'OSI (ISO/CEI 9545).

Les services et protocoles de la FAL sont fournis par des entités d'application (application entity, AE) de la FAL contenues dans les processus d'application. L'AE de la FAL se compose d'un jeu d'Éléments de service application (ASE, Application Service Element) orientés objet et d'une Entité de gestion de couche (LME, Layer Management Entity) qui gère l'AE. Les ASE fournissent des services de communication qui fonctionnent sur un jeu de classes d'objets de processus d'application (APO, application process object) connexes. L'un des ASE de la FAL est un ASE de gestion qui fournit un jeu commun de services pour la gestion des instances de classes de la FAL.

Bien que ces services spécifient, du point de vue des applications, la manière dont la demande et les réponses sont émises et délivrées, ils n'incluent pas une spécification de ce que les applications qui demandent et qui répondent doivent en faire. À savoir, les aspects comportementaux des applications ne sont pas spécifiés; seule une définition des demandes et réponses qu'elles peuvent envoyer/recevoir est spécifiée. Cela permet une plus grande flexibilité aux utilisateurs de la FAL pour normaliser un tel comportement d'objet. En plus de ces services, certains services d'appui sont également définis dans la présente norme pour fournir l'accès à la FAL afin de maîtriser certains aspects de son fonctionnement.

1.2 Spécifications

L'objectif principal de la présente norme est de spécifier les caractéristiques des services conceptuels d'une couche application qui sont adaptées à des communications à temps critique et, donc, complètent le Modèle de référence de base de l'OSI en guidant le développement des protocoles de couche application pour les communications à temps critique.

Un objectif secondaire est de fournir des trajets de migration à partir de protocoles de communications industrielles préexistants. C'est ce dernier objectif qui donne naissance à la diversité des services normalisés comme les divers Types de la CEI 61158, et les protocoles correspondants normalisés dans les sous-parties de la CEI 61158-6.

La présente spécification peut être utilisée comme la base pour les interfaces de programmation d'applications (Application Programming-Interfaces) formelles. Néanmoins, elle n'est pas une interface de programmation formelle et il sera nécessaire pour toute interface de ce type de traiter de questions de mise en œuvre qui ne sont pas couvertes par la présente spécification, y compris

a) les tailles et l'ordonnancement des octets pour les divers paramètres de service à plusieurs octets, et

b) la corrélation de primitives appariées "request-confirm" (c'est-à-dire: demande et confirmation) ou "indication-response" (c'est-à-dire: indication et réponse).

1.3 Conformité

La présente norme ni ne spécifie de mises en œuvre individuelles ou de produits individuels ni ne contraint les mises en œuvre d'entités de couche application au sein des systèmes d'automatisation industriels.

Il n'y a pas de conformité d'équipement à la présente norme de définition des services de couche application. Au contraire, la conformité est obtenue par une mise en œuvre de protocoles conformes de couche application qui satisfont à tout type donné de services de couche application définis dans la présente norme.

2 Références normatives

Les documents suivants sont cités en référence de manière normative, en intégralité ou en partie, dans le présent document et sont indispensables pour son application. 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).

NOTE Toutes les parties de la série CEI 61158, ainsi que la CEI 61784-1 et la CEI 61784-2 font l'objet d'une maintenance simultanée. Les références croisées à ces documents dans le texte se rapportent par conséquent aux éditions datées dans la présente liste de références normatives.

CEI 61131-3, *Automates programmables – Partie 3: Langages de programmation*

CEI 61158-1:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 1: Présentation et lignes directrices des séries CEI 61158 et CEI 61784*

CEI 61158-3-12, *Réseaux de communication industriels – Spécifications des bus de terrain- Partie 3-12: Définition des services de la couche liaison de données – Eléments de type 12*

ISO/IEC 646:1991, *Information technology – ISO 7-bit coded character set for information interchange* (disponible en anglais seulement)

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/IEC 8802-3, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications* (disponible en anglais seulement)

ISO 9545, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Structure de la couche Application*

ISO/IEC 10646, *Information technology – Universal Coded Character Set (UCS)* (disponible en anglais seulement)

ISO/CEI 10731, *Technologies de l'information – Interconnexion de systèmes ouverts – Modèle de référence de base – Conventions pour la définition des services OSI*

ISO/IEC/IEEE 60559, *Information technology – Microprocessor Systems – Floating-Point arithmetic* (disponible en anglais seulement)

IEEE 802.1D, *IEEE standard for local and metropolitan area networks– Media access control (MAC) Bridges;* disponible à l'adresse <http://www.ieee.org>

IETF RFC 791, *Internet Protocol darpa internet program protocol specification;* Disponible à l'adresse <http://www.ietf.org>

3 Termes, définitions, symboles, abréviations et conventions

Pour les besoins du présent document, les termes, définitions, symboles, abréviations et conventions 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 l'ISO/CEI 7498-3 et utilise les termes suivants y définis:

3.1.1 entités (N) correspondantes [ISO/CEI 7498-1]
entités d'AL correspondantes (N=7)

3.1.2 entité (N) [ISO/CEI 7498-1]
entité d'AL (N=7)

3.1.3 couche (N) [ISO/CEI 7498-1]
couche AL (N=7)

3.1.4 gestion de couche [ISO/CEI 7498-1]

3.1.5 entités homologues [ISO/CEI 7498-1]

3.1.6 nom primitif [ISO/CEI 7498-3]

3.1.7 protocole d'AL [ISO/CEI 7498-1]

3.1.8 unité de données de protocole d'AL [ISO/CEI 7498-1]

3.1.9 réinitialisation [ISO/CEI 7498-1]

3.1.10 acheminement [ISO/CEI 7498-1]

3.1.11 segmentation [ISO/CEI 7498-1]

3.1.12 service (N)
service d'AL (N=7) [ISO/CEI 7498-1]

3.1.13 unité de données de service d'AL [ISO/CEI 7498-1]

3.1.14 transmission simplex d'AL [ISO/CEI 7498-1]

3.1.15 sous-système d'AL [ISO/CEI 7498-1]

3.1.16 gestion-systèmes [ISO/CEI 7498-1]

3.1.17 données d'utilisateur d'AL [ISO/CEI 7498-1]

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 liaison de données:

3.2.1 acceptant

3.2.2 service asymétrique

3.2.3 (primitive) "confirm";
(primitive) "requestor.deliver"

3.2.4 (primitive) "deliver"

3.2.5 primitive de service d'AL;primitive

3.2.6 fournisseur de service d'AL

3.2.7 utilisateur de service d'AL

3.2.8 (primitive) "indication";
(primitive) "acceptor.deliver"

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

3.2.10 demandeur

3.2.11 réponse (primitive);
(primitive) "acceptor.submit"

3.2.12 (primitive) "submit"

3.2.13 service symétrique

3.3 Termes et définitions pour les services de la couche application et de liaison de données

Pour les besoins du présent document, les termes et définitions suivants s'appliquent.

3.3.1
application
fonction ou structure de données pour laquelle des données sont consommées ou produites

3.3.2
objets d'applications
classes contenant plusieurs objets permettant de gérer et d'assurer l'échange dynamique des messages sur le réseau et au sein de l'appareil de réseau

3.3.3
esclave de base
appareil esclave qui prend en charge uniquement l'adressage physique des données

3.3.4
bit
unité d'information consistant en un 1 ou un 0

Note 1 à l'article: il s'agit de la plus petite unité de données qui puisse être transmise.

3.3.5
client
1) objet qui utilise les services d'un autre objet (serveur) pour accomplir une tâche

2) initiateur d'un message auquel un serveur réagit

3.3.6
objet de communication
composant qui gère et assure l'échange de messages pendant le mode exécution à travers le réseau

3.3.7
connexion
liaison logique entre deux objets d'application au sein du même appareil ou dans des appareils différents

3.3.8
cyclique
relatif à des événements qui se répètent d'une manière régulière et répétitive

3.3.9
donnée
terme générique servant à se référer à toute information transportée sur un bus de terrain

3.3.10
cohérence de données
moyen pour une émission et un accès cohérents de l'objet de donnée d'entrée ou de sortie entre client et serveur et au sein du client et du serveur

3.3.11
type de données (data type)
relation entre les valeurs et le codage pour les données du type en question

Note 1 à l'article: Les définitions des types de données selon la CEI 61131-3 s'appliquent.

3.3.12
objet «data type» ("type de données")
entrée dans le dictionnaire d'objets indiquant un type de données

3.3.13
passerelle par défaut
appareil avec au moins deux interfaces dans deux sous-réseaux IP différents agissant comme routeur pour un sous-réseau

3.3.14
appareil
entité physique connectée au réseau de terrain composée d'au moins un élément de communication (l'élément de réseau) et qui peut avoir un élément de commande et/ou un élément final (transducteur, actionneur, etc.)

3.3.15
profil d'appareil
ensemble d'informations et de fonctionnalité dépendant de l'appareil qui assure la cohérence entre des appareils similaires relevant du même type d'appareil

3.3.16
information de diagnostic
toute donnée disponible au niveau du serveur à des fins de maintenance

3.3.17
horloges réparties
méthode permettant de synchroniser les esclaves et maintenir une base de temps globale

3.3.18
erreur
discordance entre une valeur ou un état calculé(e), observé(e) ou mesuré(e) et la valeur ou l'état spécifié(e) ou théoriquement correct(e)

3.3.19
classe d'erreurs
regroupement général pour des définitions d'erreurs connexes et des codes d'erreurs correspondants

3.3.20
code d'erreur
identification d'un type spécifique d'erreur dans une classe d'erreurs

3.3.21
événement
instance d'un changement de conditions

3.3.22
unité de gestion de mémoire de réseau de terrain
fonction qui établit une ou plusieurs correspondances entre les adresses logiques et la mémoire physique.

3.3.23
entité d'unité de gestion de mémoire de réseau de terrain
élément simple de l'unité de gestion de mémoire de réseau de terrain: une correspondance entre un espace d'adresses logiques cohérent et un emplacement cohérent de la mémoire physique

3.3.24
trame
synonyme déconseillé de DLPDU

3.3.25
esclave complet
appareil esclave qui prend en charge à la fois l'adressage physique et l'adressage logique des données

3.3.26
indice
adresse d'un objet au sein d'un processus d'application

3.3.27
interface
frontière partagée entre deux unités fonctionnelles, définies par les caractéristiques fonctionnelles, caractéristiques de signal ou autres caractéristiques selon le cas approprié

3.3.28
little endian (petit boutiste)
méthode de représentation des données pour les nombres de plus de 8 bits, pour laquelle l'octet de poids faible est émis le premier

3.3.29
maître
appareil qui commande le transfert de données sur le réseau et lance l'accès des esclaves aux supports en envoyant des messages et qui constitue l'interface au système de commande

3.3.30
mapping
correspondance entre deux objets de telle manière que l'un des objets soit partie intégrante de l'autre objet

3.3.31
paramètres de mapping
ensemble de valeurs définissant la correspondance entre des objets d'application et des objets de données de processus

3.3.32
support
câble, fibre optique ou autre moyen par lequel des signaux de communication sont émis entre deux ou plusieurs points

Note 1 à l'article: Le terme "media" est utilisé comme pluriel de "medium".

3.3.33
message
série ordonnée d'octets censés acheminer de l'information

Note 1 à l'article: Normalement utilisé pour acheminer de l'information entre homologues au niveau de la couche application.

3.3.34
réseau
ensemble de nœuds reliés par un certain type de support de communication, notamment des répétiteurs intermédiaires, ponts, routeurs et passerelles de couche inférieure

3.3.35
nœud
a) simple entité de DL telle qu'elle apparaît sur une liaison locale

b) point d'extrémité d'une liaison dans un réseau ou un point auquel deux ou plusieurs liaisons se rencontrent

[Dérivée de la CEI 61158-2]

3.3.36
objet
représentation abstraite d'un composant particulier au sein d'un appareil

Note 1 à l'article: Un objet peut être

a) une représentation abstraite des capacités d'un appareil. Les objets peuvent être constitués de tout ou partie des composants suivants:

   1) données (informations qui varient au fil du temps);

   2) configuration (paramètres pour le comportement);

   3) méthodes (choses qui peuvent être faites en utilisant les données et la configuration);

b) un ensemble de données connexes (sous la forme de variables) et de méthodes (procédures) pour opérer sur les données en question qui ont une interface et un comportement clairement définis

3.3.37
dictionnaire d'objets
structure de données adressée par Index (indice) et Sub-index (sous-indice) qui contient des descriptions d'objets types de données, d'objets de communication et d'objets d'application

3.3.38
donnée de processus
ensemble d'objets d'application destinés à être transférés de façon cyclique ou acyclique à des fins de mesure et de commande

3.3.39
objet de données de processus
structure décrite par des paramètres de mapping contenant une ou plusieurs entités de données de processus

3.3.40
segment
ensemble d'un maître réel et d'un ou plusieurs esclaves

3.3.41
serveur
objet qui fournit des services à un autre objet (client)

3.3.42
service
opération ou fonction qu'un objet et/ou une classe d'objets accomplit à la demande d'un autre objet et/ou d'une autre classe d'objets

3.3.43
esclave
entité de DL accédant au support uniquement après avoir été initiée par l'esclave précédent ou le maître

3.3.44
sous-indice
sous-adresse d'un objet au sein du dictionnaire d'objets

3.3.45
gestionnaire de synchronisation
ensemble d'éléments de commande servant à coordonner l'accès à des objets utilisés simultanément

3.3.46
voie (ou canal) de gestionnaire de synchronisation
simples éléments de commande servant à coordonner l'accès à des objets utilisés simultanément

3.3.47
commutateur
pont MAC tel que défini dans l'IEEE 802.1D

3.4 Symboles et abréviations communs

AL-	Préfixe relatif à la couche application
ALE	AL-entity (Entité d'AL, l'instance locale active de la couche application)
AL	AL-layer (Couche application)
APDU	AL-protocol-data-unit (Unité de données de protocole d'AL)
ALM	AL-management (Gestion d'AL)
ALME	AL-management entity (Entité de gestion AL, l'instance active locale de la gestion d'AL)
ALMS	AL-management service (Service de gestion d'AL)
ALS	AL-service (Service d'AL)
AR	Application Relationship (Relation d'applications)
ASE	Application Service Element (Élément de service application)
CAN	Controller Area Network (Gestionnaire de réseau de communication)
CiA	CAN in Automation (CAN en automation)
CoE	CAN application protocol over Type 12 services (Protocole d'application CAN sur services de Type 12)
CSMA/CD	Carrier sense multiple access with collision detection (Accès multiple par surveillance du signal et détection de collision)
DC	Distributed Clocks (Horloges distribuées)
DL	Préfixe relatif à la couche liaison de données)
DNS	Domain name system (Système de nom de domaine) (serveur pour la résolution des noms dans les réseaux IP)
E²PROM	Electrically erasable programmable read only memory (Mémoire morte à reprogrammation électrique)
EoE	Ethernet tunneled over Type 12 services (Services Ethernet en tunnel sur Type 12)
ESC	Type 12 slave controller (Appareil de commande d'esclave de Type 12)
FCS	Frame Check Sequence (Séquence de contrôle de trame)
FIFO	First-in first-out («Premier entré, premier sorti», méthode de mise en file d'attente)
FMMU	Fieldbus Memory Management Unit (Unité de gestion de mémoire de bus de terrain)
FoE	File access with Type 12 services (Accès fichier avec les services de Type 12)
HDR	Header (En-tête)
ID	Identifier (Identificateur)
IETF	Internet Engineering Task Force (Groupe de travail d'ingénierie Internet)
IP	Internet Protocol (Protocole Internet)
LAN	Local Area Network (Réseau local)
MAC	Medium Access Control (Commande d'accès au support)
OD	Object dictionary (Dictionnaire d'objets)
OSI	Open Systems Interconnection (Interconnexion des systèmes ouverts)
PDI	Physical device internal interface (Interface interne d'appareil physique, un jeu d'éléments qui permet l'accès à des services de DL depuis l'AL)
PDO	Process Data Object (Objet de données de processus)
Ph-layer	Ph-layer (couche Physique)
QoS	Quality of service (qualité de service)
RAM	Random access memory (Mémoire vive)
Rx	Receive (recevoir)
Objet SDO	Service Data Object (Objet de données de service)

SII	Slave information interface (Interface d'information esclave)
SM	Synchronization manager (Gestionnaire de synchronisation)
Sync Manager	Synchronization manager (Gestionnaire de synchronisation)
TCP	Transmission Control Protocol (Protocole de commande de transport)
Tx	Transmit (émettre)
UDP	User Datagram Protocol (Protocole datagramme d'utilisateur)
WKC	Working counter (Compteur en fonction)

3.5 Conventions

La présente norme utilise les conventions descriptives données dans l'ISO/CEI 10731.

Le modèle de service, les primitives de service et les diagrammes de temps-séquence utilisés sont des descriptions totalement abstraites; ils ne constituent pas une spécification pour une mise en œuvre.

Les primitives de service, utilisées pour représenter les interactions utilisateur de service/fournisseur de service (voir ISO/CEI 10731), acheminent des paramètres qui indiquent les informations disponibles dans l'interaction entre utilisateur et fournisseur.

La présente norme utilise un format de tableau pour décrire les paramètres de composants des primitives de service. Les paramètres qui s'appliquent à chaque groupe de primitives de service sont consignés en tableaux dans toute la suite de la présente norme. Chaque tableau comporte jusqu'à cinq colonnes, contenant le nom du paramètre de service, et une colonne chacune pour les primitives et les sens de transfert de paramètres utilisés par le service:

– les paramètres d'entrée de la primitive "request" (demande);
– les paramètres de sortie de la primitive "indication";
– les paramètres d'entrée de la primitive "response" (réponse); et
– les paramètres de sortie de la primitive "confirm" (confirmation).

NOTE Les primitives "request", "indication", "response" et "confirm" sont aussi appelées respectivement primitives "requestor.submit", "acceptor.deliver", "acceptor.submit" et "requestor.deliver" (voir ISO/CEI 10731).

Un paramètre (ou une partie de celui-ci) est énuméré dans chaque rangée de chaque tableau. Dans les colonnes appropriées de la primitive de service, un code est utilisé pour spécifier le type d'usage du paramètre sur la primitive et le sens de paramètres spécifiés dans la colonne:

M Le paramètre est obligatoire pour la primitive;

U Le paramètre est une option de l'utilisateur et peut ou peut ne pas être fourni, cela dépendant de l'usage dynamique de l'utilisateur du service. Lorsqu'il n'est pas fourni, une valeur par défaut est supposée pour le paramètre;

C Le paramètre est conditionné à d'autres paramètres ou à l'environnement de l'utilisateur du service;

(Blanc/Vide) Le paramètre n'est jamais présent;

Certaines entrées sont en plus qualifiées par des éléments entre parenthèses. Ceux-ci peuvent être une constante spécifique à un paramètre:

(=) indique que le paramètre équivaut du point de vue de la sémantique au paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau;

Dans n'importe quelle interface particulière, il n'est pas indispensable d'énoncer tous les paramètres de façon explicite. Certains peuvent être associés de façon implicite à la primitive.

Dans les diagrammes qui illustrent ces interfaces, des lignes tiretées indiquent des relations de cause à effet ou de temps-séquence tandis que les traits ondulés indiquent que des événements sont grosso modo contemporains.

4 Concepts

4.1 Concepts communs

L'ensemble de la CEI 61158-1, Article 9 est incorporé par référence, sauf neutralisation spécifique en 4.2.

4.2 Concepts spécifiques de type

4.2.1 Principe de fonctionnement

La présente Norme et ses normes d'accompagnement de Type 12 décrivent une technologie Ethernet temps réel qui vise à maximaliser l'utilisation de la largeur de bande Ethernet en duplex intégral. La commande d'accès au support utilise le principe de maître/esclave selon lequel le nœud maître (typiquement le système de commande) envoie les trames Ethernet aux nœuds esclaves, qui extraient des données de ces trames et insèrent des données dans ces trames.

Du point de vue de l'Ethernet, un segment de Type 12 est un appareil Ethernet pris séparément qui reçoit et envoie des trames Ethernet de la norme ISO/CEI 8802-3. Cependant, cet appareil Ethernet n'est pas limité à un seul appareil de commande Ethernet avec microprocesseur aval, mais peut consister en un grand nombre d'appareils esclaves de Type 12. Ceux-ci traitent directement les trames entrantes et extraient les données d'utilisateur pertinentes ou insèrent des données et transfèrent la trame vers le prochain appareil esclave. Le dernier appareil esclave au sein du segment renvoie la trame complètement traitée et, donc, elle est retournée par le premier appareil esclave vers le maître comme trame de réponse.

Cette procédure utilise la capacité "full duplex" (duplex intégral) de l'Ethernet: les deux sens de communication sont exploités de façon indépendante. La communication sans commutateur entre un appareil maître et un segment de Type 12 constitué d'un ou plusieurs appareils esclaves peut être établie.

Les systèmes de communication industriels doivent satisfaire à différentes exigences en termes des caractéristiques de transmission de données. Les données de paramètres sont transférées de manière acyclique et en grandes quantités. De ce fait, les exigences de temps sont relativement non critiques et la transmission est habituellement déclenchée par le système de commande. Les données de diagnostic sont également transférées de manière acyclique et sont événementielles, mais les exigences de temps sont plus drastiques et la transmission est habituellement déclenchée par un appareil périphérique.

D'autre part, les données de processus sont typiquement transférées de manière cyclique avec des temps de cycle différents. Les exigences de temps sont le plus drastiques pour la communication de données de processus. La couche application (AL) de Type 12 prend en charge une diversité de services et de protocoles pour satisfaire à ces exigences différentes.

4.2.2 Vue d'ensemble des modèles de communication

La couche application de Type 12 établit une distinction entre maître et esclave. La relation de communication est toujours lancée par le maître.

Un segment de Type 12 est constitué d'au moins un appareil maître et d'un ou plusieurs esclaves. Tous les appareils esclaves prennent en charge le diagramme d'états de Type 12 (Type 12 State Machine (ESM)) et prennent en charge l'émission de données de processus de Type 12.

La relation d'applications peut être modélisée indépendamment de la relation de communication. La relation maître-esclave est la relation d'applications normalisée.

4.2.3 Description de l'élément de couche application

4.2.3.1 Gestion

La gestion obligatoire consiste en un jeu d'objets pour commander l'état d'un esclave. Une interface à la DL fournit un accès lecture à tous les registres de DL.

4.2.3.2 Interface d'informations

L'interface d'informations esclave (Slave information interface (SII)), obligatoire, est constituée de tous les objets qui peuvent être stockés de façon persistante.

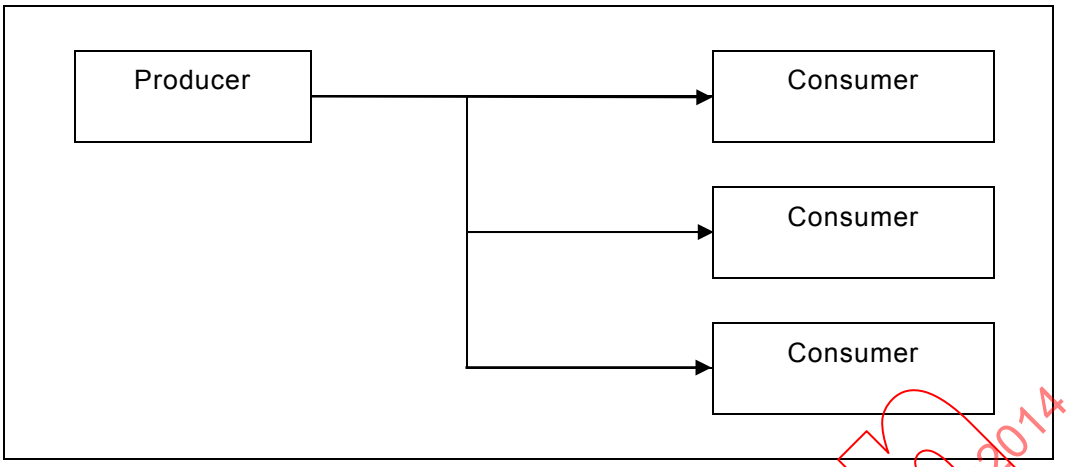
4.2.3.3 Prise en charge de la synchronisation

La prise en charge facultative du fonctionnement isochrone est constituée de plusieurs attributs pour la synchronisation et l'horodatage de signaux binaires.

4.2.3.4 Accès à l'esclave

L'entité temps réel est constituée d'une interface pour l'échange de données déclenché par le réseau et d'une interface pour l'accès déclenché par l'utilisateur aux objets d'esclave. Les objets utilisés principalement pour l'accès déclenché par le réseau sont appelés des PDO. Les SDO sont les objets qui sont utilisés pour l'accès déclenché par l'utilisateur.

Les méthodes d'accès pour les PDO sont la lecture et l'écriture dans un tampon. La structure de communication est une relation producteur-consommateur telle que montrée à la Figure 1 sans connaissance directe de la remise des données.La remise des données sera surveillée par des éléments supplémentaires dans l'esclave et dans le maître (à savoir, chien de garde et compteur en fonction). Le producteur peut être un maître comme il peut être un esclave.



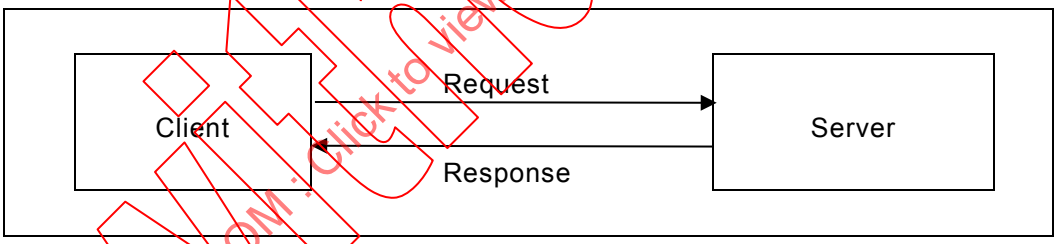
Légende

Anglais	Français
Producer	Producteur
Consumer	Consommateur

Figure 1 – Modèle producteur-consommateur

L'accès aux SDO suit le principe client-serveur. Le client émet une invocation de service vers le serveur. L'esclave démarre l'exécution du service et répond ensuite par le résultat. Il y a toujours une réponse nécessaire pour conclure ce type de service.

La Figure 2 montre le flux de production de cette interaction communicationnelle.

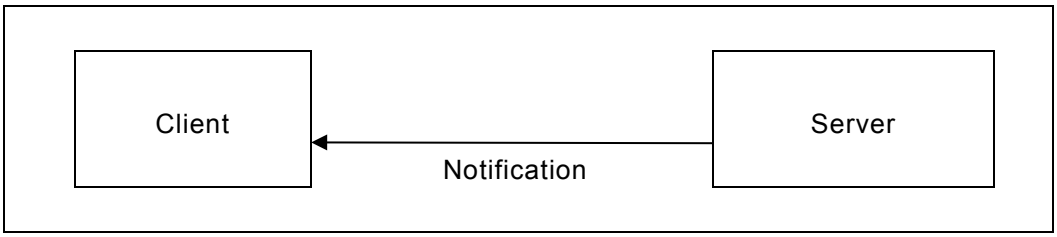


Légende

Anglais	Français
Client	Client
Request	Demande
Response	Réponse
Server	Serveur

Figure 2 – Modèle client-serveur

Il peut également y avoir un type non sollicité d'interaction serveur-client. Ce modèle est utilisé pour acheminer des données déclenchées par le serveur. Ce type de service appelé "notification" est montré à la Figure 3.



Légende

Anglais	Français
Client	Client
Notification	Notification
Server	Serveur

Figure 3 – Invocation déclenchée par le serveur

4.2.3.5 Suite TCP/UDP/IP

Le protocole facultatif est dédié pour esclaves qui utiliseront la suite normalisée de protocoles internet. Les protocoles eux-mêmes sont définis dans l'IETF. EoE décrit le mapping du Protocole IP (et genre similaire de communication) à la couche liaison de données de Type 12. IP est un type de communication sans connexion avec flot de données bidirectionnel.

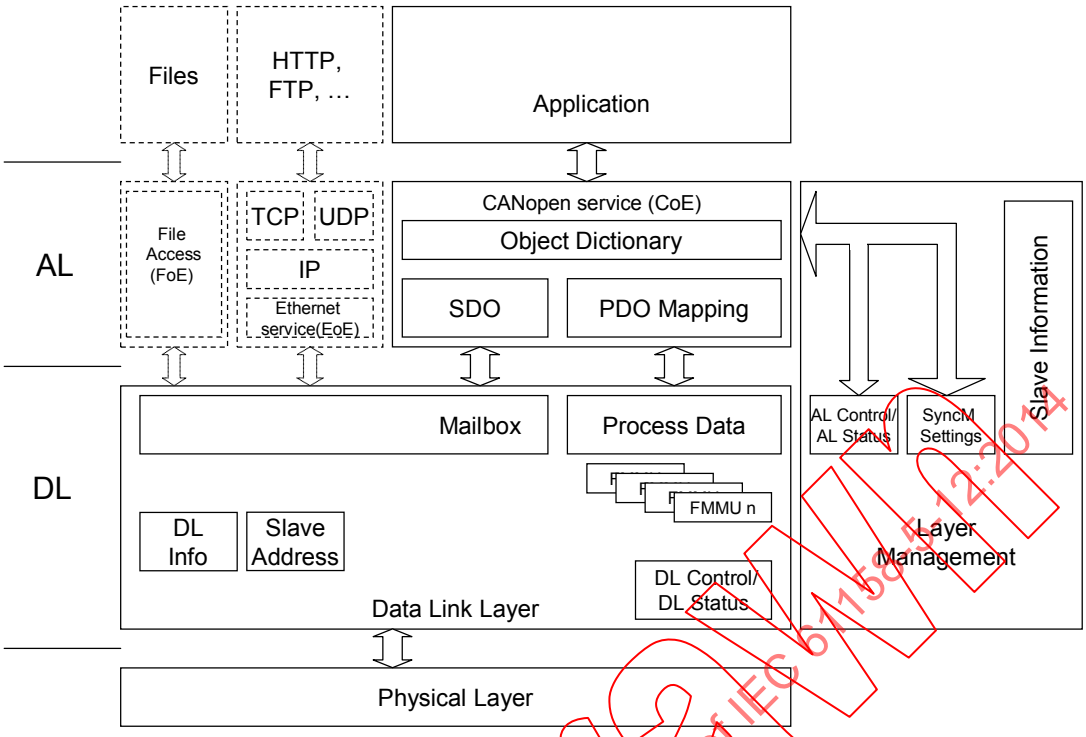
4.2.3.6 Accès à un fichier

L'utilisation principale du transfert de fichiers est constituée du téléchargement et du chargement de fichiers de programme et de données de configuration. L'accès à un fichier est effectué avec l'architecture du protocole client-serveur.

4.2.4 Modèle de référence d'esclave

4.2.4.1 Mapping avec le modèle de référence de base OSI

Le Type 12 est décrit en utilisant les principes, la méthodologie et le modèle de l'ISO/CEI 7498 Systèmes de traitement de l'information — Interconnexion des systèmes ouverts — Modèle de référence de base (OSI). Le modèle OSI fournit une approche stratifiée aux normes de communications et, par ce biais, des couches peuvent être mises au point et modifiées de manière indépendante. La spécification de Type 12 définit la fonctionnalité de haut en bas d'une pile OSI complète et un certain nombre de fonctions pour les utilisateurs de la pile. Les fonctions des couches OSI intermédiaires, les couches 3 à 6, sont consolidées soit en la couche liaison de données de Type 12, soit en la couche application de Type 12. De même, les fonctions communes aux utilisateurs de la couche d'application de bus de terrain peuvent être fournies par la couche d'application Type 12 afin de simplifier l'utilisation (voir Figure 4).



Légende

Anglais	Français
Files	Fichiers
HTTP, FTP, ...	HTTP, FTP, ...
Application	Application
AL	AL (couche application)
File Access (FoE)	Accès fichier (FoE)
TCP	TCP
UDP	UDP
CANopen service (CoE)	Service CANopen (CoE)
Object Dictionary	Dictionnaire d'objets
IP	IP
Ethernet service(EoE)	Service Ethernet (EoE)
SDO	SDO
PDO Mapping	Mapping d'objets de données de processus
Mailbox	Boîte à lettres
Process Data	Données de processus
AL Control/AL Status	Commande de couche application/Statut de couche application
SyncM Settings	Valeurs de réglage du gestionnaire de synchronisation
Slave Information	Informations d'esclave
DL	Liaison de données
DL Info	Informations de DL
Slave Address	Adresse d'esclave
FMMU	Unité de gestion de mémoire de réseau de terrain
Data Link Layer	Couche liaison de données
DL Control/DL Status	Commande de DL/Statut de DL

Anglais	Français
Layer Management	Gestionnaire de couche
Physical Layer	Couche physique

Figure 4 – Modèle de référence d'esclave

4.2.4.2 Caractéristiques de la couche liaison de données

La couche liaison de données fournit un support de base à temps critique pour les communications de données parmi des appareils connectés par l'intermédiaire d'un segment de Type 12. Le terme "à temps critique" sert à décrire des applications ayant une fenêtre temporelle, dans les limites de laquelle une ou plusieurs actions spécifiées sont tenues d'être parachevées avec un certain niveau défini de certitude. Le manquement à parachever les actions spécifiées dans les limites de la fenêtre temporelle risque d'entraîner la défaillance des applications qui demandent ces actions, avec le risque concomitant pour l'équipement, l'installation et éventuellement pour la vie humaine.

La couche liaison de données a la tâche de calculer, comparer et générer la séquence de contrôle de trame et fournir des communications en extrayant des données dans la trame Ethernet et/ou en y incluant des données. Cela se fait en fonction des paramètres de la couche liaison de données qui sont stockés en des emplacements mémoire prédéfinis. Les données d'application sont rendues disponibles à la couche application en mémoire physique, soit dans une configuration de boîte à lettres, soit au sein d'une section de données de processus.

De plus, certaines structures de données dans la couche liaison de données seront utilisées pour permettre une coordination de l'interaction entre maître et esclave, telles que par exemple AL Control (commande de couche application), AL Status (statut de couche application) et AL Event (événement de couche application) et Sync Manager settings (valeurs de réglage de Sync Manager).

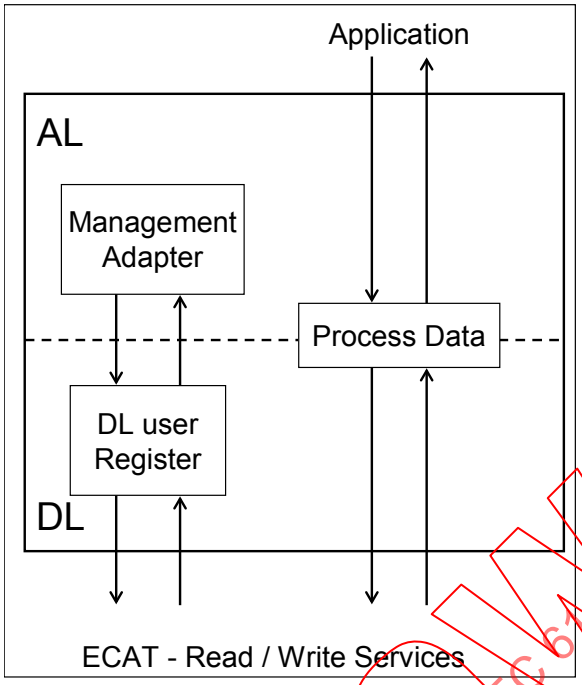
4.2.4.3 Classification AL d'esclaves

4.2.4.3.1 Appareil esclave simple

Du point de vue de la couche application, les appareils esclaves sont classés en appareils simples sans appareil de commande d'application et en appareils complexes avec appareil de commande d'application.

NOTE La classification des esclaves de la DL en esclaves de base et esclaves complets est indépendante de la vue AL, car les mécanismes d'adressages de la DL sont invisibles en l'interface AL.

Les appareils simples ont une disposition fixe des données de processus, qui est décrite dans le fichier de description de l'appareil. Les appareils simples peuvent confirmer les services de gestion d'AL (AL Management) sans une réaction au sein de l'application locale, comme noté à la Figure 5. Il n'y a pas de réaction spéciale nécessaire au fonctionnement à l'état de sécurité (par exemple, la valeur 0 sera traitée de la même manière, car aucune valeur valide ne sera envoyée).



Légende

Anglais	Français
Application	Application
AL	Couche application
Management Adapter	Adaptateur de gestion
Process Data	Données de processus
DL user Register	Registre utilisateur de DL
DL	Liaison de données
ECAT – Read / Write Services	ECAT – Services de lecture/écriture

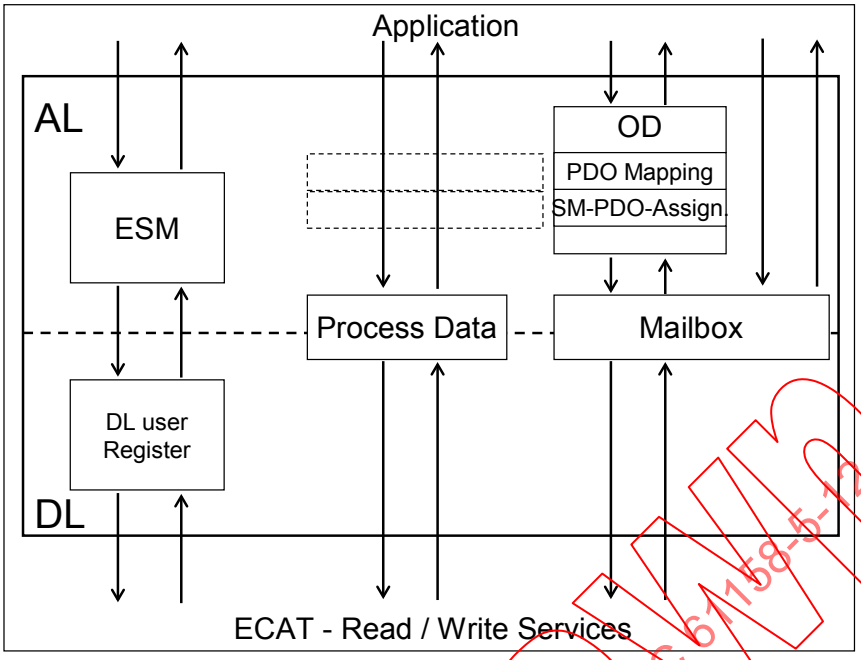
Figure 5 – Appareil esclave simple

4.2.4.3.2 Appareil esclave complexe

Comme noté à la Figure 6, les appareils complexes prennent en charge

- l'ESM,
- la Mailbox, c'est-à-dire boîte à lettres (facultatif),
- le dictionnaire d'objets CoE (recommandé si la boîte à lettres est prise en charge),
- les services SDO pour lire et/ou écrire les entrées de données du dictionnaire d'objets (recommandé si la boîte à lettres est prise en charge), et
- le service d'informations SDO pour lire les objets définis dans le dictionnaire d'objets et chaque description d'entrée au format compact (recommandé si la boîte à lettres est prise en charge).

Pour l'émission de données de processus, les objets "PDO mapping" (mapping d'objets de données de processus) et les objets "Sync Manager PDO" (assignation de PDO au gestionnaire de synchronisation), qui décrivent la disposition des données de processus, doivent être pris en charge pour lecture. Si un appareil complexe prend en charge des données de processus configurables, la configuration est réalisée en écrivant les objets "PDO mapping" et/ou les objets "Sync Manager PDO assign".



Légende

Anglais	Français
Application	Application
AL	Couche application
OD	Dictionnaire d'objets
PDO mapping	Mapping d'objets de données de processus
SM-PDO-Assign	Assignation d'objets de données de processus (PDO) au gestionnaire de synchronisation (SM)
ESM	Diagramme d'états
Process Data	Données de processus
Mailbox	Boîte à lettres
DL user Register	Registre utilisateur de DL
DL	Liaison de données
ECAT – Read / Write Services	ECAT – Services de lecture/écriture

Figure 6 – Appareil esclave complexe

Plusieurs types différents d'interaction sont définis dans la présente norme:

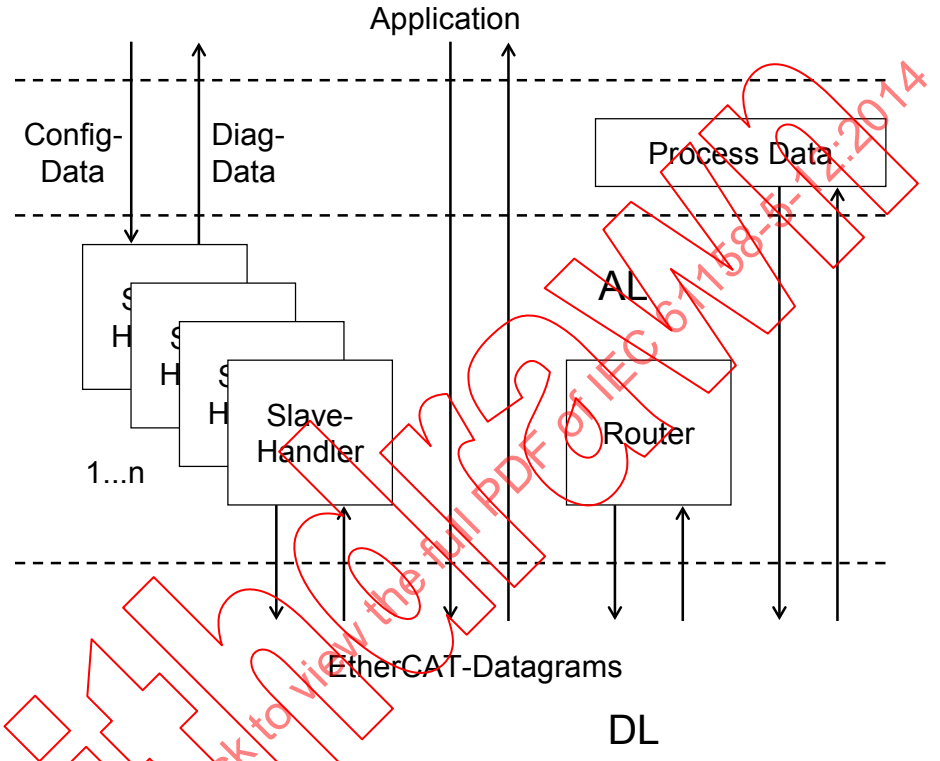
- services "CAN application protocol over Type 12" (CoE «Protocole d'application CAN sur Type 12»)
- services "Ethernet over Type 12" (EoE «Ethernet sur Type 12»)
- services "File Access over Type 12"(FoE «Accès fichier sur Type 12»)

Des types différents sont utilisés pour adresser des classes différentes d'objets. Ces types peuvent être mélangés au niveau d'une relation d'applications prise séparément.

4.2.5 Modèle de référence de maître

4.2.5.1 Vue d'ensemble

Le maître communique avec les esclaves en utilisant les services décrits dans le chapitre esclave. En outre, il y a un Handler (c'est-à-dire: descripteur) d'esclave pour chaque esclave défini dans le maître pour commander l'ESM de l'esclave et un Router (c'est-à-dire: routeur) qui permet la communication esclave-esclave par l'intermédiaire de la boîte aux lettres, comme noté à la Figure 7.



Légende

Anglais	Français
Application	Application
Config-Data	Données de configuration
Diag-Data	Données de diagnostic
Process Data	Données de processus
Handler Slave-	Descripteur d'esclave
1...n	1 à n
AL	Couche application
Router	Routeur
EtherCAT-Datagrams	Datagrammes EtherCAT
DL	Liaison de données

Figure 7 – Vue d'ensemble fonctionnelle du maître

4.2.5.2 Descripteur d'esclave

Il convient que le maître prenne en charge un descripteur d'esclave pour chaque esclave en utilisant les services d'état pour commander l'ESM de l'esclave. Le Handler d'esclave est l'image de l'ESM de l'esclave dans le maître. En outre, le Handler d'esclave peut envoyer des services SDO avant de changer l'état de l'ESM de l'esclave.

Paramètre

Position

Ce paramètre spécifie la position dans l'anneau logique qui est utilisée pour adresser l'esclave lors de la lecture de l'Identification et de l'écriture de la Station Address (adresse de poste). Il est obligatoire pour tous les esclaves.

Expected Identification

Ce paramètre spécifie l'identification prévue de l'esclave, qu'il convient de lire et comparer par le maître dans l'état "Init". Ce paramètre est obligatoire pour tous les esclaves.

Station Address

Ce paramètre spécifie l'adresse de poste qui est attribuée dans l'état "Init" à l'esclave. Tous les services ultérieurs utiliseront cette adresse de poste pour adresser l'esclave. Ce paramètre est obligatoire pour tous les esclaves.

Mailbox Configuration

Ce paramètre spécifie la configuration des voies 0 et 1 du Sync Manager pour la boîte à lettres qui est écrite dans l'état "Init" dans l'esclave. Ce paramètre est obligatoire pour les esclaves complexes.

FMMU Configuration

Ce paramètre spécifie la configuration des voies FMMU qui est écrite dans l'état "Pre-Operational" dans l'esclave.

Process Data Configuration

Ce paramètre spécifie la configuration des voies du Sync Manager qui est utilisée pour les données de processus et qui est écrite dans l'état "Pre-Operational" dans l'esclave.

PDO mapping

Ce paramètre spécifie les objets "PDO mapping" («mapping de PDO») qui peuvent être écrits dans l'état "Pre-Operational" dans l'esclave.

Sync Manager PDO assign

Ce paramètre spécifie les objets "Sync Manager PDO assign" (assignation de PDO au Sync Manager) qui peuvent être écrits dans l'état "Pre-Operational" dans l'esclave.

Start Up Objects

Ce paramètre spécifie les objets issus du dictionnaire d'objets de l'esclave qui peuvent être écrits dans l'esclave à partir du Handler d'esclave pendant le démarrage (start up).

4.2.5.3 Router

Le Router (c'est-à-dire: routeur) peut être utilisé pour plusieurs applications:

- acheminement de services de boîte à lettres de l'esclave client vers l'esclave serveur
- acheminement de réponses de services de l'esclave serveur vers l'esclave client
- transmission de services de boîte à lettres provenant d'appareils de tierces parties
- transmission de réponses de services de boîte à lettres vers des appareils de tierces parties

La tâche du routeur est d'écraser en écriture le champ "adresse" du service de boîte à lettres avec l'adresse de poste du client ou avec une adresse virtuelle avant d'acheminer le service de boîte à lettres vers le serveur adressé par le champ "adresse" d'origine. Le champ "adresse" de la réponse de service de boîte à lettres est écrasé par le routeur avec l'adresse de poste du serveur avant d'acheminer la réponse de service de boîte à lettres vers l'esclave client adressé par le champ "adresse" d'origine ou vers l'adresse IP/adresse MAC correspondante en cas d'adresse virtuelle.

5 ASE "Data type"

5.1 Généralités

L'ensemble de la CEI 61158-1, Paragraphe 5.1 est incorporé par référence.

5.2 Définition formelle des objets "types de données"

L'ensemble de la CEI 61158-1, Paragraphe 5.2 est incorporé par référence.

5.3 Types de données définis pour la FAL

5.3.1 Types Fixed length (longueur fixe)

5.3.1.1 Types Boolean (booléens)

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	1
2	Data type Name	=	Boolean
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Ce type de données exprime le type de données Boolean avec les valeurs TRUE (c'est-à-dire: vrai) et FALSE (c'est-à-dire: faux).

5.3.1.2 Types Bitstring (chaîne de bits)

5.3.1.2.1 BIT2

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	31
2	Data type Name	=	BIT2
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BIT2 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 2.

5.3.1.2.2 BIT3

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	32
2	Data type Name	=	BIT3
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BIT3 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 3.

5.3.1.2.3 BIT4

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	33
2	Data type Name	=	BIT3
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BIT4 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 4.

5.3.1.2.4 BIT5

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	34
2	Data type Name	=	BIT5
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BIT5 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 5.

5.3.1.2.5 BIT6

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	35
2	Data type Name	=	BIT6
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

n BIT6 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 6.

5.3.1.2.6 BIT7

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	36
2	Data type Name	=	BIT7
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BIT7 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 7.

5.3.1.2.7 BIT8

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	37
2	Data type Name	=	BIT8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BIT8 est une séquence ordonnée de types de données Boolean, numérotés de 1 à 8.

5.3.1.2.8 BITARR8

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	45
2	Data type Name	=	BITARR8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Un BITARR8 est une séquence ordonnée de bits, numérotés de 1 à 8.

5.3.1.2.9 BITARR16

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	46
2	Data type Name	=	BITARR16
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	2

Un BITARR16 est une séquence ordonnée de bits, numérotés de 1 à 15.

5.3.1.2.10 BITARR32

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	47
2	Data type Name	=	BITARR32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

Un BITARR32 est une séquence ordonnée de bits, numérotés de 1 à 31.

5.3.1.3 Types Currency (monnaies)

Il n'y a pas de types Currency définis pour le Type 12.

5.3.1.4 Types Date/Time (date et heure)

5.3.1.4.1 TimeOfDay

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	12
2	Data type Name	=	TimeOfDay
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	6

Ce type de données est constitué de deux éléments de valeurs unsigned (non signées) et exprime l'heure du jour et la date. Le premier élément est un type de données Unsigned32 et donne l'heure après minuit en millisecondes. Le second élément est un type de données Unsigned16 et donne la date en comptant les jours à partir du 1er janvier 1984.

5.3.1.4.2 TimeDifference

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	13
2	Data type Name	=	TimeDifference
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4 or 6

Ce type de données est constitué de deux éléments de valeurs unsigned (non signées) qui expriment la différence de temps. Le premier élément est un type de données Unsigned32 qui donne une partie fractionnaire d'un jour en millisecondes. Le second élément facultatif est un type de données Unsigned16 qui donne la différence en jours.

5.3.1.5 Types Enumerated (énumérés)

Il n'y a pas de types Enumerated définis pour le Type 12.

5.3.1.6 Types Handle (identification)

Il n'y a pas de types Handle définis pour le Type 12.

5.3.1.7 Types numériques

5.3.1.7.1.1 float

Ce type de données est le même que le type Float32.

5.3.1.7.1.2 Float32

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	8
2	Data type Name	=	Float32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

Ce type a une longueur de quatre octets. Le format de float32 est celui défini par l'ISO/CEI/IEEE 60559 comme étant en simple précision ("single precision").

5.3.1.7.1.3 double

Ce type de données est le même que le type Float64.

5.3.1.7.1.4 Float64

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	17
2	Data type Name	=	Float64
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	8

Ce type a une longueur de huit octets. Le format de float64 est celui défini par l'ISO/CEI/IEEE 60559 comme étant en double précision ("double precision").

5.3.1.7.2 Types Integer (entiers)

5.3.1.7.2.1 Integer8

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	2
2	Data type Name	=	Integer8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à un octet.

5.3.1.7.2.2 SINT

Ce type de la CEI 61131-3 est le même que le type Integer8.

5.3.1.7.2.3 char

Ce type de données est le même que le type Integer8.

5.3.1.7.2.4 Integer16

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	3
2	Data type Name	=	Integer16
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	2

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à deux octets.

5.3.1.7.2.5 INT

Ce type de la CEI 61131-3 est le même que le type Integer16.

5.3.1.7.2.6 short

Ce type de données est le même que le type Integer16.

5.3.1.7.2.7 Integer24

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	16
2	Data type Name	=	Integer24
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	3

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à trois octets.

5.3.1.7.2.8 Integer32

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	4
2	Data type Name	=	Integer32
3	Format	=	FIXED LENGTH

4.1	Octet Length	=	4

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à quatre octets.

5.3.1.7.2.9 DINT

Ce type de la CEI 61131-3 est le même que le type Integer32.

5.3.1.7.2.10 long

Ce type de données est le même que le type Integer32.

5.3.1.7.2.11 Integer40

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	18
2	Data type Name	=	Integer40
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	5

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à cinq octets.

5.3.1.7.2.12 Integer48

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	19
2	Data type Name	=	Integer48
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	6

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à six octets.

5.3.1.7.2.13 Integer56

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	20
2	Data type Name	=	Integer56
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	7

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à sept octets.

5.3.1.7.2.14 Integer64

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	21
2	Data type Name	=	Integer64
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	8

Ce type entier (Integer) est un nombre binaire en complément à deux avec une longueur égale à huit octets.

5.3.1.7.2.15 LINT

Ce type de la CEI 61131-3 est le même que le type Integer64.

5.3.1.7.3 Types non signés (Unsigned)

5.3.1.7.3.1 Unsigned8

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	5
2	Data type Name	=	Unsigned8
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	1

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type a une longueur égale à un octet.

5.3.1.7.3.2 USINT

Ce type de la CEI 61131-3 est le même que le type Unsigned8.

5.3.1.7.3.3 unsigned char

Ce type de données est le même que le type Unsigned8.

5.3.1.7.3.4 BYTE

Ce type de données est le même que le type USINT.

5.3.1.7.3.5 Unsigned16

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	6
2	Data type Name	=	Unsigned16
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	2

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de deux octets.

5.3.1.7.3.6 UINT

Ce type de la CEI 61131-3 est le même que le type Unsigned16.

5.3.1.7.3.7 WORD

Ce type est utilisé de la même façon que le type UINT.

5.3.1.7.3.8 Unsigned24

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	22
2	Data type Name	=	Unsigned24
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	3

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de trois octets.

5.3.1.7.3.9 Unsigned32

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	7
2	Data type Name	=	Unsigned32
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	4

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de quatre octets.

5.3.1.7.3.10 UDINT

Ce type de la CEI 61131-3 est le même que le type Unsigned32.

5.3.1.7.3.11 Unsigned40

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	24
2	Data type Name	=	Unsigned40
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	5

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de cinq octets.

5.3.1.7.3.12 Unsigned48

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	25
2	Data type Name	=	Unsigned48
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	6

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de six octets.

5.3.1.7.3.13 Unsigned56

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	26
2	Data type Name	=	Unsigned56
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	7

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de sept octets.

5.3.1.7.3.14 Unsigned64

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	27
2	Data type Name	=	Unsigned64
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	8

Ce type est un nombre binaire. Le bit de poids fort de l'octet de poids fort est toujours utilisé comme le bit de poids fort du nombre binaire; aucun bit de signe n'est inclus. Ce type unsigned (non signé) a une longueur de huit octets.

5.3.1.7.3.15 ULINT

Ce type de la CEI 61131-3 est le même que le type Unsigned64.

5.3.1.8 Types Pointer (pointeurs)

Il n'y a pas de types Pointer définis pour le Type 12.

5.3.1.9 Types OctetString (chaîne d'octets)

Il n'y a pas de types OctetString de longueur fixe définis pour le Type 12.

5.3.1.10 Types de caractères VisibleString (chaîne visible)

Il n'y a pas de types VisibleString de longueur fixe définis pour le Type 12.

5.3.2 Types String (chaîne)

5.3.2.1 OctetString

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	10
2	Data type Name	=	OctetString
3	Format	=	STRING
4.1	Octet Length	=	1 à n

Un OctetString est une séquence ordonnée d'octets, numérotés de 1 à n. Pour les besoins du débat, l'octet 1 de la séquence est appelé le premier octet.

NOTE La CEI 61158-6-12 définit l'ordre d'émission.

5.3.2.2 VisibleString

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	9
2	Data type Name	=	VisibleString
3	Format	=	STRING
4.1	Octet Length	=	1 à n

Ce type est défini comme étant le type "string" (chaîne) de l'ISO/CEI 646.

5.3.2.3 UnicodeString

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	11
2	Data type Name	=	UnicodeString
3	Format	=	STRING
4.1	Octet Length	=	1 à n

Ce type est défini comme étant le type "string" (chaîne) de l'ISO/CEI 10646.

5.3.3 Types GUID

5.3.3.1 GUID

CLASSE: **Data type**

ATTRIBUTS:

1	Data type Numeric Identifier	=	29
2	Data type Name	=	GUID
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	16

Un GUID est un identificateur globalement unique ayant une longueur de 128 bits.

5.4 Spécification de service de l'ASE "Data type"

L'ensemble de la CEI 61158-1, Paragraphe 5.4 est incorporé par référence.

6 Spécification de modèle de communication

6.1 Les ASE

6.1.1 ASE "Process Data"

6.1.1.1 Vue d'ensemble

Dans l'environnement de couche application du Type 12, chaque processus d'application d'esclave peut contenir plusieurs objets pour chaque instance de processus d'application pour acheminer des données de processus. Il est structuré au moyen des PDO. Le contenu des données de processus peut être décrit par les objets "PDO Mapping" et "Sync Manager PDO assign" de l'ASE "CoE". Pour les appareils esclaves simples, les données de processus sont fixes et définies dans le fichier de description de l'appareil.

Pour la communication des données de processus, la mémoire d'application de type tampon est utilisée de sorte que le maître et l'esclave ont toujours accès aux données de processus.

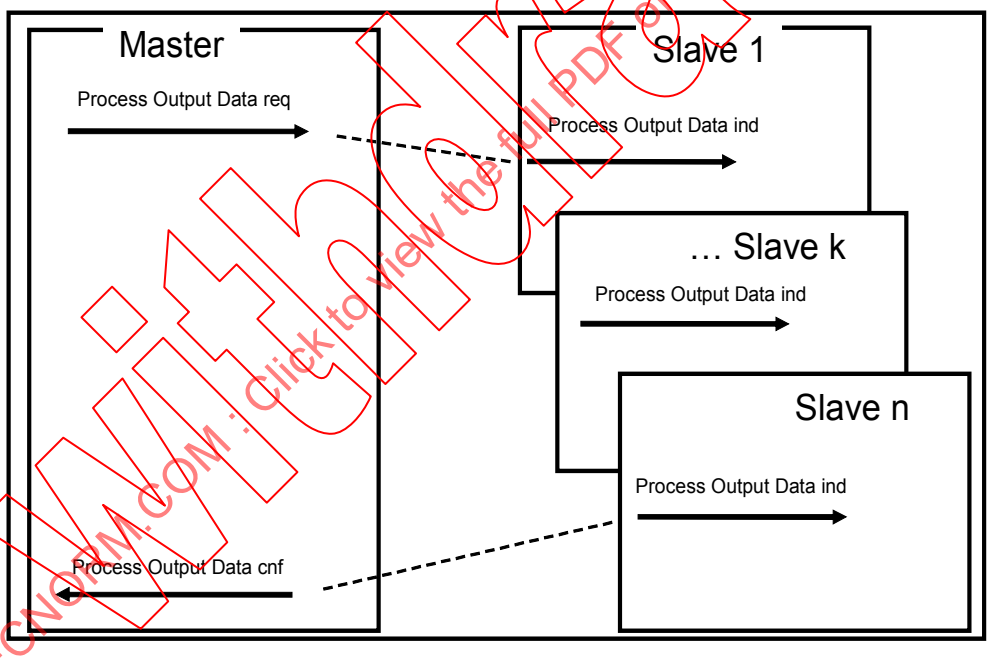
Des services supplémentaires sont fournis pour lire de façon acyclique les valeurs des objets de données de processus et pour indiquer les nouvelles valeurs pour l'objet de données d'entrée et l'objet de données de sortie.

Les objets de données de processus sont adressés de façon implicite par l'intermédiaire des services connexes. La granularité des données d'entrée ou de sorite dans un serveur/fournisseur est fonction des attributs de configuration correspondants.

L'ASE "Process data" (données de processus) utilise le modèle d'accès producteur/consommateur. Cela signifie que la mise à jour des données de processus avec les valeurs des entrées et la mise à jour des valeurs de sortie avec les données de processus sont découplées de l'acheminement des données. La réception d'une nouvelle donnée est indiquée par la primitive de service "indication" "Process Output Data".

Les primitives des services "Process Output Data" sont mappées avec les primitives de mémoire d'application du type tamponné décrites dans la DL. Il est recommandé, mais pas exigé, d'utiliser des entités FMMU. Configurée avec des FMMU, une seule demande de "Process Output Data" peut résulter en plusieurs indications de "Process Output Data". La confirmation de données de processus montrera au maître le succès ou l'échec de la procédure de mise à jour.

La Figure 8 montre les primitives entre le maître et les esclaves dans une séquence de "Process Output Data".



Légende

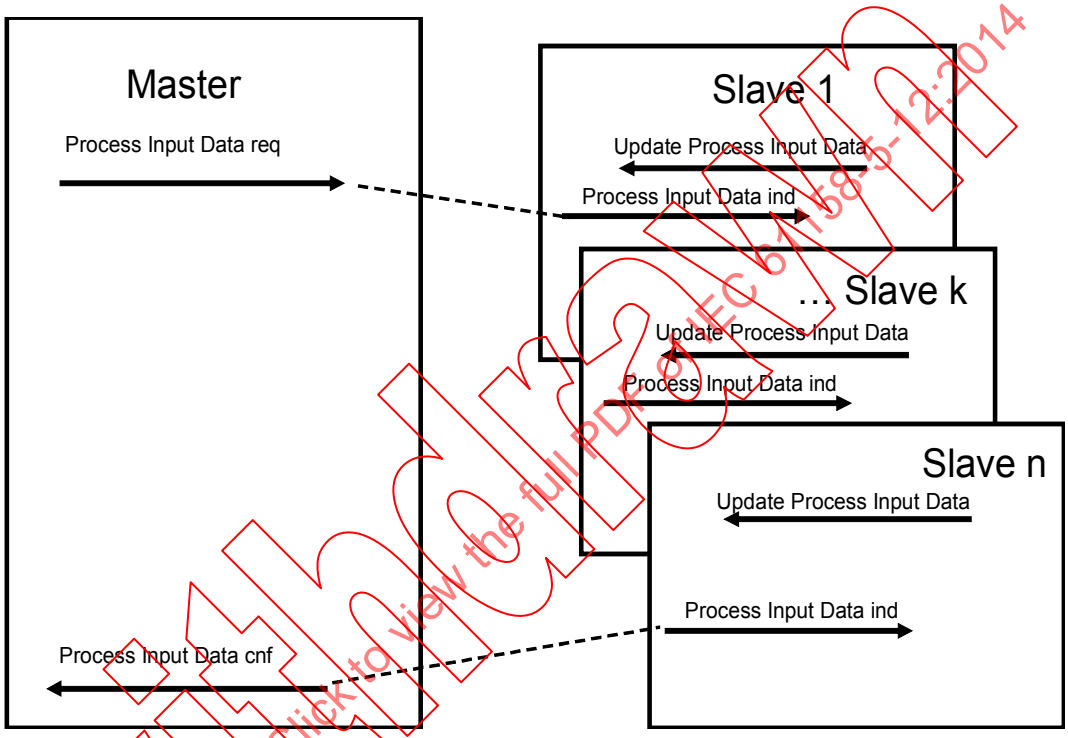
Anglais	Français
Master	Maître
Process Ouput Data req	Demande de données de sortie de processus
Slave 1	Esclave 1
Process Ouput Data ind	Indication de données de sortie de processus
Slave k	Esclave k
Slave k	Esclave n
Process Ouput Data cnf	Confirmation de données de sortie de processus

Figure 8 – Séquence de données de sortie de processus

Le maître envoie habituellement des données de sortie de processus à plusieurs esclaves en émettant un service d'écriture de DL ou de lecture-écriture (RW). Chaque esclave récupère l'AL Event (événement d'AL) du Sync Manager correspondant. L'appareil de commande d'AL de l'esclave peut lire les données de sortie de processus à tout moment dans la mémoire d'application correspondante.

Les primitives des services "Process Input Data" sont mappées avec les primitives de mémoire d'application du type tamponné décrites dans la DL.

La Figure 9 montre les primitives entre le maître et les esclaves dans une séquence de "Process Input Data".



Légende

Anglais	Français
Master	Maître
Process Input Data req	Demande de données d'entrée de processus
Slave 1	Esclave 1
Update Process Input Data	Données d'entrée de processus de mise à jour
Process Input Data ind	Indication de données d'entrée de processus
Slave k	Esclave k
Slave k	Esclave n
Process Input Data cnf	Confirmation de données d'entrée de processus

Figure 9 – Séquence de données d'entrée de processus

Le maître lit habituellement des données d'entrée de processus issues de plusieurs esclaves avec un service logique de lecture (Read) ou de lecture-écriture (RW). Le maître récupère les données issues du tampon écrit précédemment. Chaque esclave récupère l'AL Event du Sync Manager correspondant si les données d'entrées sont lues. L'appareil de commande d'AL de l'esclave peut écrire les données d'entrée de processus à tout moment dans la mémoire d'application correspondante.

Le modèle formel de l'ASE "process data" (données de processus) fait l'objet d'une description ci-après, suivie d'une description de ses services. En outre, l'ASE "process data" (données de processus) représente la structure réelle d'entrée et de sortie d'un appareil.

Pour toutes les primitives de services, il convient que les zones mémoire des paramètres qui sont écrites par des primitives de services actives simultanément ne se chevauchent pas.

6.1.1.2 Spécification de la classe de données de processus (Process data)

6.1.1.2.1 Modèle formel

L'objet "Process data" est décrit par le modèle suivant:

ASE: **ASE "Process Data"**

CLASSE: **Process Data**

CLASS ID: non utilisé

CLASSE PARENTE: TOP

ATTRIBUTS:

1.	(m)	Attribut-clé:	Implicit
2.	(m)	Attribut:	List of buffer
2.1	(m)	Attribut:	Buffertype
2.2	(m)	Attribut:	List of PDO
2.2.1	(m)	Attribut:	PDO index
2.2.2	(m)	Attribut:	List of Entry
2.2.2.1	(m)	Attribut:	Index (Indice)
2.2.2.2	(m)	Attribut:	Subindex
2.2.2.3	(m)	Attribut:	Bitlen

SERVICES:

1.	(o)	OpsService:	Process Output Data (Traiter les données de sortie)
2.	(o)	OpsService:	Process Input Data (Taiter les données d'entrée)
3.	(o)	OpsService:	Update Process Input Data (Mettre à jour les données d'entrée)

Les objets pour la structuration et l'accès supplémentaire peuvent être trouvés dans l'ASE "CoE"

6.1.1.2.2 Attributs

Implicit

L'attribut "Implicit" indique que l'objet de donnée de processus est adressé de façon implicite par les services.

List of buffer
Un Buffer (c'est-à-dire: tampon) se compose des éléments de liste suivants:

Buffertype
Cet attribut spécifie le type du tampon.

Valeurs permises: Input (entrée) ou Output (sortie).

List of PDO
Un élément Buffer (c'est-à-dire: tampon) se compose des éléments de liste suivants:

PDO index
Cet attribut spécifie l'indice de l'objet de données de processus. Sa plage admissible s'étend de 0x1600 à 0x17FF et de 0x1A00 à 0x1BFF.

List of entry
Cet attribut est constitué des attributs suivants.

Index
Cet attribut spécifie la référence à l'objet. Sa plage admissible s'étend de 1 à 0x7FFF.

Subindex
Cet attribut spécifie la référence à l'entrée d'objet. Sa plage admissible s'étend de 0 à 0xFF.

Bitlen
Cet attribut spécifie la longueur en bits de la valeur de l'entrée d'objet. Sa plage admissible s'étend de 1 à 255.

6.1.1.2.3 Services

Process output data
Ce service facultatif est utilisé par les maîtres pour acheminer des données de sortie vers l'esclave.

Process input data
Ce service facultatif est utilisé par les esclaves pour éditer des données d'entrée.

Update process Input data
Ce service facultatif est utilisé par les esclaves pour fournir des données d'entrée à éditer.

6.1.1.3 Spécification de service de l'ASE "Process data"

6.1.1.3.1 Services pris en charge

L'ASE "Process data" (données de processus) définit les services:

Process Output Data

Process Input Data

Update Process Input Data

6.1.1.3.2 Service "Process output data"

6.1.1.3.2.1 Vue d'ensemble du service

Le service "Process Output Data" est utilisé pour acheminer des données du maître vers l'esclave.

6.1.1.3.2.2 Primitives de service

Le Tableau 1 montre les paramètres du service.

Tableau 1 – Process output data

Nom de paramètre	Req	Ind	Cnf
Argument			
List of slave address	M	M (=)	
List of RxPDO	M	M (=)	
Output data	M	M (=)	
Result(+)			S
Result(–)			S
NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa primitive "request" précédente correspondante relève d'une initiative locale. Voir 1.2.			

Argument
L'argument achemine les paramètres spécifiques au service relatifs à la demande de service.

List of slave address
Ce paramètre spécifie l'identificateur pour les esclaves.

List of RxPDO
Ce paramètre spécifie un jeu fixe prédéfini d'objets RxPDO.

Output data
Ce paramètre spécifie la valeur des éléments de l'objet "Output Data".

L'allocation des entrées est effectuée de sorte que les entrées soient placées selon un ordre séquentiel. Si la longueur en bits est inférieure à 8 et le nombre de bits s'ajuste dans le bit non utilisé de l'octet effectif, l'objet sera mappé avec l'octet effectif, le premier bit étant positionné dans le premier bit non utilisé.

NOTE La structuration des données de sortie n'est pas fixe et peut varier avec l'utilisation des services CoE et des unités FMMU.

Result(+)
Ce paramètre de type sélection indique que la demande de service a réussi.
Result(–)
Ce paramètre de type sélection indique que la demande de service a échoué.

6.1.1.3.2.3 Procédure de service

Conformément aux caractéristiques du transport cyclique d'un tampon à l'autre, le comportement suivant est possible:

– Les demandes de "Process Output Data" sont émises plus vite que le contenu des tampons n'est transporté sur le réseau. Dans ce cas, les valeurs fournies avec le plus récent service "Process Output Data" sont acheminées sur le réseau.

– Les demandes de "Process Output Data" sont émises plus lentement que le contenu des tampons n'est transporté sur le réseau. Dans ce cas, les valeurs fournies avec chaque service "Process Output Data" sont acheminées plus d'une fois sur le réseau.

– Si les demandes de "Process Output Data" sont émises en synchronisme avec le transport du contenu des tampons, les valeurs fournies avec chaque service "Process Output Data" sont acheminées une seule fois sur le réseau.

6.1.1.3.3 Service "Process input data"

6.1.1.3.3.1 Vue d'ensemble du service

Le service "Process Input Data" est utilisé pour acheminer des données de l'esclave vers le maître.

6.1.1.3.3.2 Primitives de service

Le Tableau 2 montre les paramètres du service.

Tableau 2 – Process input data

Nom de paramètre	Req	Ind	Cnf
Argument			
List of slave address	M	M (=)	
List of TxPDO	M	M (=)	
Result(+)			S
Input data			M
Result(–)			S

NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa primitive "request" précédente correspondante relève d'une initiative locale. Voir 1.2.

Argument
L'argument achemine les paramètres spécifiques au service relatifs à la demande de service.

List of slave address
Ce paramètre spécifie l'identificateur pour les esclaves.

List of TxPDO
Ce paramètre spécifie un jeu fixe prédéfini d'objets TxPDO.

Result(+)
Ce paramètre de type sélection indique que la demande de service a réussi.

Input data
Ce paramètre spécifie la valeur des éléments de l'objet "Input Data".

L'allocation des entrées est effectuée de sorte que les entrées soient placées selon un ordre séquentiel. Si la longueur en bits est inférieure à 8 et le nombre de bits s'ajuste dans le bit non utilisé de l'octet effectif, l'objet sera mappé avec l'octet effectif, le premier bit étant positionné dans le premier bit non utilisé.

NOTE La structuration des données d'entrée n'est pas fixe et peut varier avec l'utilisation des services CoE et des unités FMMU.

Result(–)
Ce paramètre de type sélection indique que la demande de service a échoué.

6.1.1.3.3.3 Procédure de service

Conformément aux caractéristiques du transport cyclique d'un tampon à l'autre, le comportement suivant est possible:

– Les demandes de "Process Input Data" sont émises plus vite que le contenu des tampons n'est transporté sur le réseau. Dans ce cas, seules les valeurs fournies avec le plus récent service "Update Process Input Data" sont acheminées sur le réseau.

– Les demandes de "Process Input Data" sont émises plus lentement que le contenu des tampons n'est transporté sur le réseau. Dans ce cas, les valeurs fournies avec chaque service "Update Process Input Data " sont acheminées plus d'une fois sur le réseau.

– Si les demandes de "Process Input Data" sont émises en synchronisme avec le transport du contenu des tampons, les valeurs fournies avec chaque service "Update Process Input Data" sont acheminées une seule fois sur le réseau.

6.1.1.3.4 Service "Update process input data"

6.1.1.3.4.1 Vue d'ensemble du service

Le service "Update Process Input Data" est utilisé pour mettre à jour un tampon de données d'entrée dans l'esclave.

6.1.1.3.4.2 Primitives de service

Le Tableau 3 montre les paramètres du service.

Tableau 3 – Update process input data

Nom de paramètre	Req	Cnf
Argument		
List of TxPDO	M	
Input data	M	
Result(+)		S
Result(–)		S
NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa primitive "request" précédente correspondante relève d'une initiative locale. Voir 1.2.		

Argument
L'argument achemine les paramètres spécifiques au service relatifs à la demande de service.

List of TxPDO
Ce paramètre spécifie un jeu fixe prédéfini d'objets TxPDO.

Input data
Ce paramètre spécifie la valeur des éléments de l'objet "Input Data".

Result(+)
Ce paramètre de type sélection indique que la demande de service a réussi.

Result(–)
Ce paramètre de type sélection indique que la demande de service a échoué.

6.1.1.3.4.3 Procédure de service

– La procédure de service est décrite avec le service "Process Input Data".

6.1.2 SIIASE

6.1.2.1 Vue d'ensemble

Dans l'environnement de la couche application de Type 12, chaque processus d'application d'esclave a une SII (Interface d'information esclave) qui contient toutes les informations nécessaires pour l'identification d'un appareil. L'interface d'information esclave est stockée de manière persistante et spécifie les de configuration d'amorce (boot) et les données d'informations d'application. Les données de configuration d'amorce (boot) contiennent les

valeurs de réglage pour initialiser l'interface de l'appareil de commande d'esclave pendant la mise sous tension. Les données d'informations d'application contiennent le code de produit qui peut être utilisé par le maître pour trouver le fichier de configuration correspondant pour l'appareil. Les registres de l'interface d'information esclave définissent l'accès à l'interface d'information esclave qui est prise en charge par l'appareil de commande d'esclave (ESC).

Le modèle formel de l'ASE "SII"fait l'objet d'une description ci-après, suivie d'une description de ses services. En outre, l'ASE "SII" représente la structure réelle d'entrée et de sortie d'un appareil.

6.1.2.2 Spécification de la classe "SII"

6.1.2.2.1 Modèle formel

L'objet "SII" est décrit par le modèle suivant:

ASE: **SIIASE**

CLASSE: **SII**

CLASS ID: non utilisé

CLASSE PARENTE: TOP

ATTRIBUTS:

1.	(m)	Attribut-clé:	Implicit
2.	(m)	Attribut:	PDI Control
2.1	(m)	Attribut:	PDI type
2.2	(m)	Attribut:	AL state control
2.3	(o)	Attribut:	Specific settings
2.4	(o)	Attribut:	Sync impulse length
2.4	(o)	Attribut:	ConfiguredStationAddress
3.	(m)	Attribut:	Device identification
3.1	(m)	Attribut:	Vendor ID
3.2	(m)	Attribut:	Product Code
3.3	(m)	Attribut:	Revision Number
3.4	(o)	Attribut:	Serial Number

4.	(o)	Attribut:	Mailbox Configuration
4.1	(o)	Attribut:	Bootstrap
4.1.1	(o)	Attribut:	Receive memory offset
4.1.2	(o)	Attribut:	Receive size
4.1.3	(o)	Attribut:	Send memory offset
4.1.4	(o)	Attribut:	SendSize
4.2	(o)	Attribut:	Standard
4.2.1	(o)	Attribut:	Receive memory offset
4.2.2	(o)	Attribut:	Receive size
4.2.3	(o)	Attribut:	Send memory offset
4.2.4	(o)	Attribut:	Send size
4.2.5	(o)	Attribut:	Mailbox protocols
5.	(m)	Attribut:	SII information
5.1	(m)	Attribut:	Size
5.2	(o)	Attribut:	Version
6.	(o)	Attribut:	Additional parameter
6.1	(o)	Attribut:	Delay Correcion 1
6.1.1	(o)	Attribut:	Port0 delay
6.1.1	(o)	Attribut:	Port1 delay
6.2	(o)	Attribut:	Delay Correcion 2
6.2.1	(o)	Attribut:	Port0 Tx delay
6.2.2	(o)	Attribut:	Port1 Tx delay
6.2.3	(o)	Attribut:	Port2 Tx delay
6.2.4	(o)	Attribut:	Port3 Tx delay
6.2.5	(o)	Attribut:	Port0 Rx delay
6.2.6	(o)	Attribut:	Port1 Rx delay
6.2.7	(o)	Attribut:	Port2 Rx delay
6.2.8	(o)	Attribut:	Port3 Rx delay

6.3	(o)	Attribut:	Delay Correcion 3
6.3.1	(o)	Attribut:	Forward port 0 to next port
6.3.2	(o)	Attribut:	Forward non-port 0 to next port
6.3.3	(o)	Attribut:	Additive forward time closed port
7.	(o)	Attribut:	Categories
7.1	(o)	Attribut:	List of strings
7.1.1	(o)	Attribut:	Strings
7.2	(o)	Attribut:	General information
7.2.1	(o)	Attribut:	Group name
7.2.2	(o)	Attribut:	Image
7.2.3	(o)	Attribut:	Order information
7.2.4	(o)	Attribut:	Nom
7.2.5	(o)	Attribut:	Couche physique
7.2.6	(o)	Attribut:	CoE details
7.2.7	(o)	Attribut:	FoE details
7.2.8	(o)	Attribut:	EoE Details
7.2.9	(o)	Attribut:	NoLWR
7.2.10	(o)	Attribut:	Power budget
7.3	(o)	Attribut:	List of FMMU
7.3.1	(o)	Attribut:	Entry
7.4	(o)	Attribut:	List of Sync Manager
7.4.1	(o)	Attribut:	Sync Manager
7.5	(o)	Attribut:	List of TxPDO
7.5.1	(o)	Attribut:	Index (Indice)
7.5.2	(o)	Attribut:	Related Sync Manager
7.5.3	(o)	Attribut:	Reference to DC
7.5.4	(o)	Attribut:	Nom
7.5.5	(o)	Attribut:	List of entries

7.5.5.1	(o)	Attribut:	Index (Indice)
7.5.5.2	(o)	Attribut:	Sous-indice
7.5.5.3	(o)	Attribut:	Nom
7.5.5.4	(o)	Attribut:	Data type (Type de données)
7.5.5.5	(o)	Attribut:	Data type (Type de données)
7.6	(o)	Attribut:	List of RxPDO
7.6.1	(o)	Attribut:	Index (Indice)
7.6.2	(o)	Attribut:	RelatedSync Manager
7.6.3	(o)	Attribut:	Reference to DC
7.6.4	(o)	Attribut:	Nom
7.6.5	(o)	Attribut:	List of entries
7.6.5.1	(o)	Attribut:	Index (Indice)
7.6.5.2	(o)	Attribut:	Sous-indice
7.6.5.3	(o)	Attribut:	Name
7.6.5.4	(o)	Attribut:	Data type (Type de données)
7.6.5.5	(o)	Attribut:	Data length
7.7	(o)	Attribut:	DC settings
7.7.1	(o)	Attribut:	DC area

SERVICES:

1.	(o)	OpsService:	Read SII
2.	(o)	OpsService:	Write SII
3.	(o)	OpsService:	Reload SII

Les objets pour la structuration et l'accès supplémentaire peuvent être trouvés dans l'ASE "CoE".

6.1.2.2.2 Attributs

Implicit

L'attribut "Implicit" indique que l'objet "SII" est adressé de façon implicite par les services.

PDI control
Cet objet se compose des éléments suivants:

PDI type
Cet attribut, défini en 6.2.2, spécifie la valeur d'initialisation pour le type de PDI de l'ASE "AR".

AL state control
Cet attribut, défini en 6.2.2, spécifie la valeur d'initialisation pour l'AL State Control de l'ASE "AR".

Specific settings
Cet attribut facultatif, défini en 6.2.2, spécifie la valeur d'initialisation pour les Specific Settings (réglages spécifiques) de l'ASE "AR".

Sync impulse length
Cet attribut facultatif spécifie la valeur pour la Sync Impulse (impulsion de synchronisation).

Configured station address
Cet attribut facultatif spécifie la valeur d'initialisation pour l'adresse de poste configurée, telle que spécifiée dans la CEI 61158-3-12.

Device identification
Cet objet se compose des éléments suivants:

Vendor ID
Cet attribut spécifie l'identificateur de vendeur d'initialisation tel que détaillé en 6.1.4.1.2.3.6. Sa longueur est de quatre octets.

Product code
Cet attribut spécifie le code de produit d'initialisation tel que détaillé en 6.1.4.1.2.3.6. Sa longueur est de quatre octets.

Revision number
Cet attribut spécifie le numéro de révision d'initialisation tel que détaillé en 6.1.4.1.2.3.6. Sa longueur est de quatre octets.

Serial number
Cet attribut spécifie le numéro de série d'initialisation tel que détaillé en 6.1.4.1.2.3.6. Sa longueur est de quatre octets.

Mailbox configuration
Cet objet se compose des éléments suivants:

Bootstrap
Cet objet facultatif se compose des éléments suivants:

Receive memory offset
Cet attribut spécifie le décalage par rapport à la zone mémoire. Sa plage admissible s'étend de 0x1000 à 0xFFDA.

Receive size
Cet attribut spécifie la taille de la boîte à lettres de réception. Sa plage admissible est de 38 octets à 1 486 octets.

Send memory offset
Cet attribut spécifie le décalage par rapport à la zone mémoire. Sa plage admissible s'étend de 0x1000 à 0xFFDA.

Send size
Cet attribut spécifie la taille de la boîte à lettres de réception. Sa plage admissible est de 38 octets à 1 486 octets.

Standard
Cet objet facultatif se compose des éléments suivants:

Receive memory offset
Cet attribut spécifie le décalage par rapport à la zone mémoire. Sa plage admissible s'étend de 0x1000 à 0xFFDA.

Receive size
Cet attribut spécifie la taille de la boîte à lettres de réception. Sa plage admissible est de 38 octets à 1 486 octets.

Send memory offset
Cet attribut spécifie le décalage par rapport à la zone mémoire. Sa plage admissible s'étend de 0x1000 à 0xFFDA.

Send size
Cet attribut spécifie la taille de la boîte à lettres de réception. Sa plage admissible est de 38 octets à 1 486 octets.

Mailbox protocols
Cet attribut spécifie les protocoles de boîte à lettres pris en charge. Ses valeurs autorisées sont EoE, CoE, FoE ainsi que des codages spécifiques au profil et à l'application.

SII information
Cet objet se compose des éléments suivants:

Size
Cet attribut spécifie la taille de la structure de SII en unités de 1 Kibit (1 024 bits). Sa plage s'étend de 1 unité à 65 536 unités de 1 Kibit chacune, représentée comme étant de 0 à 0xFFFF.

Version
Cet attribut spécifie la version de la structure de la SII; sa valeur est 1.

Additional parameter
Cet objet se compose des éléments suivants:

Delay correction 1
Cet attribut est constitué des éléments suivants, chacun de ceux-ci ayant une granularité de 100 ps.

Port0 delay
Cet attribut spécifie un facteur correctif pour le retard de ligne devant être ajouté si le maître est derrière le Port 0.

Port1 delay
Cet attribut spécifie un facteur correctif pour le retard de ligne devant être ajouté si le maître est derrière le Port 1.

Delay correction 2
Cet attribut est constitué des éléments suivants, chacun de ceux-ci ayant une granularité de 100 ps.

Port0 Tx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port0 en tant qu'expéditeur est impliqué dans un chemin. Il représente la différence temporelle entre le signal au support de PhL et l'entrée à la PhL.

Port1 Tx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port1 en tant qu'expéditeur est impliqué dans un chemin. Il représente la différence temporelle entre le signal au support de PhL et l'entrée à la PhL.

Port2 Tx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port2 en tant qu'expéditeur est impliqué dans un chemin. Il représente la différence temporelle entre le signal au support de PhL et l'entrée à la PhL.

Port3 Tx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port3 en tant qu'expéditeur est impliqué dans un chemin. Il représente la différence temporelle entre le signal au support de PhL et l'entrée à la PhL.

Port0 Rx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port0 en tant que destinataire est impliqué dans un chemin. Il représente la différence temporelle entre l'entrée à la PhL et le signal au support de PhL.

Port1 Rx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port1 en tant que destinataire est impliqué dans un chemin. Il représente la différence temporelle entre l'entrée à la PhL et le signal au support de PhL.

Port2 Rx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port2 en tant que destinataire est impliqué dans un chemin. Il représente la différence temporelle entre l'entrée à la PhL et le signal au support de PhL.

Port3 Rx delay
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port3 en tant que destinataire est impliqué dans un chemin. Il représente la différence temporelle entre l'entrée à la PhL et le signal au support de PhL.

Delay correction 3
Cet attribut est constitué des éléments suivants, chacun de ceux-ci ayant une granularité de 100 ps.

Forward port 0 to next port
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si le Port0 en tant que destinataire est impliqué dans un chemin. Il représente la différence temporelle entre la PhL vers la DL au Port 0 et celle de la DL vers la PhL au prochain port.

Forward non-port 0 to next port
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si un port autre que le Port0 en tant que destinataire est impliqué dans un chemin. Il représente la différence temporelle entre la PhL vers la DL au port autre que le Port 0 et celle de la DL vers la PhL au prochain port.

Additive forward time closed port
Cet attribut spécifie un facteur correctif pour le retard devant être ajouté si un port est dans l'état fermé (closed). Le temps de base est “Additive forward time closed Port” (temps additif direct en port fermé) si le Port 0 est impliqué comme destinataire virtuel.

Categories
Cet objet se compose des éléments suivants:

List of strings
Cet attribut est constitué des éléments de liste suivants:

Strings
Cet attribut spécifie des éléments textuels qui sont référencés par leur position dans la liste dans les éléments de catégorie suivants.

General information
Cet attribut est constitué des éléments suivants:

Group name
Cet attribut spécifie une référence à une chaîne dans l'ensemble de chaînes (Strings). Il est utilisé pour spécifier un nom de groupe pour l’appareil.

Image
Cet attribut spécifie une référence à une chaîne dans l'ensemble de chaînes (Strings). Il est utilisé pour spécifier un nom d'une représentation graphique pour l’appareil.

Order information
Cet attribut spécifie une référence à une chaîne dans l'ensemble de chaînes (Strings). Il est utilisé pour spécifier les informations d'ordre pour l'appareil.

Name
Cet attribut spécifie une référence à une chaîne dans l'ensemble de chaînes (Strings). Il est utilisé pour spécifier les informations de nom pour l'appareil.

Physical layer
Cet attribut spécifie la technologie principale de couche physique de l'appareil. Ses valeurs admissibles sont "non utilisé", E-Bus, 100BASE-TX, 100BASE-FX

CoE details
Cet attribut spécifie une structure avec les informations de CoE.

FoE details
Cet attribut spécifie une structure avec les informations de FoE.

NoLRW
Cet attribut Boolean est vrai si le service de DL "LogicalReadWrite" n'est pas pris en charge par cet appareil.

Power budget
Cet attribut spécifie le courant de fonctionnement requis par cet appareil. Les valeurs négatives indiquent que l'appareil fournit de la puissance. La plage de cet attribut s'étend de -128 mA à +127 mA.

List of FMMU
Cet attribut est constitué des éléments de liste suivants:

Entry
Cet attribut spécifie l'utilisation préférentielle d'une FMMU. Un maximum de trois utilisations peut être pris en charge.

List of Sync Manager
Cet attribut est constitué des éléments de liste suivants:

Sync Manager
Cet attribut spécifie l'utilisation préférentielle d'un Sync Manager, tel que défini dans la CEI 61158-3-12.

List of TxPDO
Cet attribut facultatif est présent s'il y a un nombre limité de sélections de PDO qui sont toujours disponibles pour ce type d'appareil. Il est constitué de la liste suivante d'éléments:

Index
Cet attribut spécifie l'indice d'un PDO. Sa plage admissible s'étend de 0x1A00 à 0x1BFF.

Related Sync Manager
Cet attribut spécifie l'allocation de la PDU à un Sync Manager. Sa plage admissible s'étend de 0 à 15.

Reference to DC sync
Cet attribut spécifie l'une d'un maximum de seize Sync Method (méthodes de synchronisation) telles que spécifiées dans le dictionnaire d'objets de CoE.

Name
Cet attribut spécifie le nom comme référence à une chaîne.

List of entries
Cet attribut est constitué des éléments de liste suivants:

Index
Cet attribut spécifie l'Index (indice) d'un objet.

Subindex
Cet attribut spécifie le Subindex (sous-indice) d'un objet.

Name
Cet attribut spécifie le nom comme référence à une chaîne.

Data type
Cet attribut spécifie le type de données comme indice à un répertoire d'objets de CoE.

Data length
Cet attribut spécifie la longueur de l'objet en bits.

List of RxPDO
Cet attribut facultatif est présent s'il y a un nombre limité de sélections de PDO qui sont toujours disponibles pour ce type d'appareil. Il est constitué de la liste suivante d'éléments:

Index
Cet attribut spécifie l'Index (indice) d'un PDO dans la plage de 0x1600 à 0x17FF

Related Sync Manager
Cet attribut spécifie l'allocation d'un Sync Manager; sa plage s'étend de 0 à 15.

Reference to DC sync
Cet attribut spécifie la Sync Method (méthode de synchronisation) telle que spécifiée dans le dictionnaire d'objets de CoE; sa plage s'étend de 0 à 15.

Name
Cet attribut spécifie le nom comme référence à une chaîne.

List of entries
Cet attribut est constitué des éléments de liste suivants:

Index
Cet attribut spécifie l'Index (indice) d'un objet.

Subindex
Cet attribut spécifie le Subindex (sous-indice) d'un objet.

Name
Cet attribut spécifie le nom comme référence à une chaîne.

Data type
Cet attribut spécifie le type de données comme indice à un répertoire d'objets de CoE.

Data length
Cet attribut spécifie la longueur de l'objet en bits.

DC settings
Cet attribut est constitué de l'élément suivant:

DC area
Cet attribut spécifie une chaîne d'octets dont le contenu sera défini dans les profils d'appareils.

6.1.2.2.3 Services

Write SII
Ce service facultatif est utilisé par les maîtres pour acheminer 2 octets de données vers la SII d'esclave.

Read SII
Ce service facultatif est utilisé par les maîtres pour acheminer 4 octets de données de la SII d'esclave vers le maître.

Reload SII
Ce service facultatif est utilisé par des maîtres pour acheminer 2 octets de données à partir de la SII d'esclave et mettre à jour le registre miroir dans la DL d'esclave.

6.1.2.3 Spécification des services de l'ASE "SII"

6.1.2.3.1 Services pris en charge

L'ASE "SII" définit les services

Read SII

Write SII

Reload SII

6.1.2.3.2 Service "SII read"

6.1.2.3.2.1 Vue d'ensemble du service

Le service "SII Read" est utilisé pour transférer un maximum de 4 octets de données du serveur vers le client. Le serveur répond avec le résultat de l'opération de lecture.

6.1.2.3.2.2 Primitives de service

Le Tableau 4 montre les primitives de service et le paramètre du service SII Read.

Tableau 4 – SII read

Nom de paramètre	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Word Index	M	M (=)		
Result(+)			S	S
Data			M	M (=)
Result(–)			S	S

NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa primitive "request" précédente correspondante relève d'une initiative locale. Voir 1.2.

Argument
L'argument achemine les paramètres spécifiques au service relatifs à la demande de service.

Address
Ce paramètre spécifie l'adresse de poste de l'esclave.

Word Index
Ce paramètre spécifie l'indice en mot de la zone de la SII qui doit être lue.

Result(+)
Ce paramètre de type sélection indique que la demande de service a réussi.

Data
Ce paramètre spécifie la valeur lue de l'objet.

Result(–)
Ce paramètre de type sélection indique que la demande de service a échoué.

6.1.2.3.3 Service "SII write"

6.1.2.3.3.1 Vue d'ensemble du service

Le service "SII Write" est utilisé pour transférer un maximum de 2 octets de données du client vers le serveur. Le serveur répond avec le résultat de l'opération d'écriture.

6.1.2.3.3.2 Primitives de service

Le Tableau 5 montre les primitives de service et le paramètre du service SII Write.

Tableau 5 – SII write

Nom de paramètre	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Word Index	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S

NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa primitive "request" précédente correspondante relève d'une initiative locale. Voir 1.2.

Argument
L'argument achemine les paramètres spécifiques au service relatifs à la demande de service.

Address
Ce paramètre spécifie l'adresse de poste de l'esclave.

Word Index
Ce paramètre spécifie l'indice en mot de la zone de la SII qui doit être écrite.

Data
Ce paramètre spécifie la valeur d'objet qui doit être écrite.

Result(+)
Ce paramètre de type sélection indique que la demande de service a réussi.

Result(–)
Ce paramètre de type sélection indique que la demande de service a échoué.

6.1.2.3.4 Service "SII reload"

6.1.2.3.4.1 Vue d'ensemble du service

Le service "SII Reload" est utilisé pour transférer un maximum de 2 octets de données du client vers le serveur et rafraîchir les registres de DL associés aux nouvelles valeurs de SII. Le serveur répond avec le résultat de l'opération de rechargement.

6.1.2.3.4.2 Primitives de service

Le Tableau 6 montre les primitives de service et le paramètre du service SII Reload.

Tableau 6 – SII reload

Nom de paramètre	Req	Ind	Rsp	Cnf
Argument				
Address	M	M (=)		
Word Index	M	M (=)		
Data	M	M (=)		
Result(+)			S	S
Result(–)			S	S
NOTE La méthode par laquelle une primitive "confirm" est corrélée à sa primitive "request" précédente correspondante relève d'une initiative locale. Voir 1.2.				

Argument
L'argument achemine les paramètres spécifiques au service relatifs à la demande de service.

Address
Ce paramètre spécifie l'adresse de poste de l'esclave.

Word index
Ce paramètre spécifie l'indice en mot de la zone de la SII qui doit être écrite.

Data
Ce paramètre spécifie la valeur d'objet qui doit être écrite.

Result(+)
Ce paramètre de type sélection indique que la demande de service a réussi.

Result(–)
Ce paramètre de type sélection indique que la demande de service a échoué.

6.1.3 ASE "Isochronous"

6.1.3.1 Vue d'ensemble

Dans l'environnement de la couche application de Type 12, un processus d'application d'un esclave peut avoir une PDI isochrone qui spécifie les informations nécessaires pour le fonctionnement synchrone de plusieurs appareils. La spécification isochrone suppose une interface qui consiste en deux signaux de synchronisation qui se rapportent à l'horloge système, telle que définie dans la CEI 61158-3 et la CEI 61158-4. De plus, un support pour signaux d'entrée de verrouillage peut être pris en charge.

Les registres de PDI isochrone définissent l'accès à l'interface d'information esclave si le fonctionnement isochrone est pris en charge par l'appareil de commande d'esclave (ESC).

Le modèle formel de l'ASE isochrone est décrit ci-après, les services sont des services locaux de lecture et d'écriture des événements de DL et locaux.

6.1.3.2 Spécification de la classe "Isochronous"

6.1.3.2.1 Modèle formel

L'objet isochrone est décrit par le modèle suivant:

ASE: **ASE "Isochronous"**

CLASSE: **Isochronous**

CLASS ID: non utilisé

CLASSE PARENTE: TOP

ATTRIBUTS:

1.	(m)	Attribut-clé:	Implicit
2.	(o)	Attribut:	Sync parameter
2.1	(o)	Attribut:	Cyclic operation enable
2.2	(o)	Attribut:	Sync0 activate
2.3	(o)	Attribut:	Sync1 activate
2.4	(o)	Attribut:	Sync impulse length
2.5	(o)	Attribut:	Interrupt 0 Status
2.6	(o)	Attribut:	Interrupt 1 Status
2.7	(o)	Attribut:	Cyclic Operation Start Time
2.8	(o)	Attribut:	Sync0 cycle time
2.9	(o)	Attribut:	Sync1 cycle time
3.	(m)	Attribut:	Latch Parameter
3.1	(m)	Attribut:	Latch 0 positive edge
3.2	(m)	Attribut:	Latch 0 negative edge
3.3	(m)	Attribut:	Latch 1 positive edge
3.4	(o)	Attribut:	Latch 1 negative edge
3.1	(m)	Attribut:	Latch 0 positive event
3.2	(m)	Attribut:	Latch 0 negative event
3.3	(m)	Attribut:	Latch 1 positive event
3.4	(o)	Attribut:	Latch 1 negative event

3.1	(m)	Attribut:	Latch 0 positive edge value
3.2	(m)	Attribut:	Latch 0 negative edge value
3.3	(m)	Attribut:	Latch 1 positive edge value
3.4	(o)	Attribut:	Latch 1 negative edge value

SERVICES:

1.	(o)	OpsService:	DL-Read Local (lecture locale de DL)
2.	(o)	OpsService:	DL-Write Local (écriture locale de DL)
1.	(o)	OpsServices:	DL-read
2.	(o)	OpsServices:	DL-write

La description des services peut être consultée dans la CEI 61158-3-12.

6.1.3.2.2 Attributs

Implicit

L'attribut "Implicit" indique que l'objet isochrone est adressé de façon implicite par les services.

Sync parameter
Cet objet se compose des éléments suivants:

Cyclic operation enable
Cet attribut Boolean permet la fonction cyclique Sync 0 ou 1.

Sync0 activate
Cet attribut Boolean permet le fonctionnement Sync 0.

Sync1 activate
Cet attribut Boolean permet le fonctionnement Sync 1.

Sync Impulse Length
Cet attribut spécifie la valeur de "Sync Impulse Length" (longueur d'impulsion de synchronisation) telle que spécifiée dans l'ASE SII.

Specific Settings
Cet attribut facultatif, défini en 6.2.2, spécifie la valeur d'initialisation pour les Specific Settings (réglages spécifiques) de l'ASE "AR".

SyncImpulseLen
Cet attribut facultatif spécifie la valeur pour la "Sync Impulse" (impulsion de synchronisation) en multiples de 10 ns.

Interrupt 0 Status
Cet attribut Boolean facultatif indique une interruption Sync0 active.

Interrupt 1 Status
Cet attribut Boolean facultatif indique une interruption Sync1 active.

Cyclic Operation Start Time
Cet attribut facultatif fixe une heure de début relative au temps système pour le fonctionnement cyclique.

Sync0 cycle time
Cet attribut fixe le temps de cycle de Sync0 en multiples de 1 ns pour Sync0.

Sync1 cycle time
Cet attribut fixe le temps de cycle de Sync1 en multiples de 1 ns pour Sync0.

Latch Parameter
Cet objet se compose des éléments suivants:

Latch 0 positive edge
Cet attribut Boolean permet l'opération Latch 0 pour un événement simple (vrai) ou un verrouillage continu.

Latch 0 negative edge
Cet attribut Boolean permet l'opération Latch 0 pour un événement simple (vrai) ou un verrouillage continu.

Latch 1 positive edge
Cet attribut Boolean permet l'opération Latch 1 pour un événement simple (vrai) ou un verrouillage continu.

Latch 1 negative edge
Cet attribut Boolean permet l'opération Latch 1 pour un événement simple (vrai) ou un verrouillage continu.

Latch 0 positive event
Cet attribut Boolean indique un événement de flanc positif (montant) Latch 0.

Latch 0 negative event
Cet attribut Boolean indique un événement de flanc négatif (descendant) Latch 0.

Latch 1 positive event
Cet attribut Boolean indique un événement de flanc positif (montant) Latch 1.

Latch 1 negative event
Cet attribut Boolean indique un événement de flanc négatif (descendant) Latch 1.

Latch 0 positive edge value
Cet attribut contient la valeur du temps système dans le cas d'un événement de flanc positif (montant) Latch 0.

Latch 0 negative edge value
Cet attribut contient la valeur du temps système dans le cas d'un événement de flanc négatif (descendant) Latch 0.

Latch 0 positive edge value
Cet attribut contient la valeur du temps système dans le cas d'un événement de flanc positif (montant) Latch 0.

Latch 0 negative edge value
Cet attribut contient la valeur du temps système dans le cas d'un événement de flanc négatif (descendant) Latch 0.

6.1.3.2.3 Services

DL-read local
Ce service facultatif est utilisé par l'application esclave pour lire le paramètre isochrone. Il convient que la valeur de l'adresse mémoire du paramètre se situe dans la plage de 0x980 à 0x9FF.

DL-write local
Ce service facultatif est utilisé par l'application esclave pour écrire le paramètre isochrone. Il convient que la valeur de l'adresse mémoire du paramètre se situe dans la plage de 0x980 à 0x9FF. Seul un sous-ensemble des registres peut être écrit.

DL-read
Ce service facultatif est utilisé par le maître pour lire le paramètre isochrone. Il convient que la valeur de l'adresse mémoire du paramètre se situe dans la plage de 0x980 à 0x9FF.

DL-write
Ce service facultatif est utilisé par le maître pour écrire le paramètre isochrone. Il convient que la valeur de l'adresse mémoire du paramètre se situe dans la plage de 0x980 à 0x9FF. Seul un sous-ensemble des registres peut être écrit.

6.1.4 ASE CoE

6.1.4.1 Vue d'ensemble

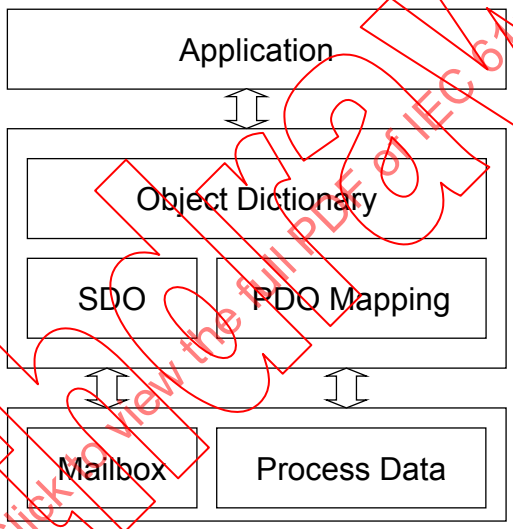
6.1.4.1.1 Généralités

CANopen est un protocole de communication mis au point à l'origine pour des systèmes basés sur CAN. Il s'agit d'un réseau intégré normalisé ayant des capacités de configuration très flexibles.

NOTE CANopen (la spécification particulière CiA DS 301) est normalisée comme EN 50325-4.

Le Type 12 utilise la définition de services de protocole d'application CAN.

Le dictionnaire d'objets contient des paramètres, des données d'application et les informations de mapping entre interface de données de processus et données d'application (PDO mapping). Ses entrées sont accessibles par l'intermédiaire d'objets de données de service (SDO), comme montré à la Figure 10.



Légende

Anglais	Français
Application	Application
Object Dictionary	Dictionnaire d'objets
SDO	Objet de données de service
PDO mapping	Mapping d'objets de données de processus
Mailbox	Boîte à lettres
Process Data	Données de processus

Figure 10 – Modèle de serveur CoE

6.1.4.1.2 Object dictionary (Dictionnaire d'objets)

6.1.4.1.2.1 Structure du dictionnaire d'objets (Object Dictionary)

Le dictionnaire d'objets contient d'une manière normalisée tous les objets de données CoE de l'appareil. Il est une collection des structures de données de paramètres d'appareils qui sont accessibles par l'intermédiaire des services "SDO Upload" (chargement d'objets de données de services) et "SDO Download" (téléchargement d'objets de données de services).

Les services d'informations de SDO permettent de lire les entrées disponibles du dictionnaire d'objets et la description d'entrée relative à une entrée du dictionnaire d'objets.

Le dictionnaire d'objets est constitué des zones telles que décrites dans le Tableau 7.

Tableau 7 – Allocation des zones de SDO

Zone	Contenu
Zone Data type:	Définition des types de données
Zone CoE Communication:	Définition des variables qui peuvent être utilisées pour tous les serveurs et à des fins de communications spécialisées
Zone Manufacturer Specific:	Définition des variables spécifiques au fabricant
Zone Device Profile:	Définition des variables définies dans un profil d'appareils
Zone Reserved:	réservée pour un usage futur

6.1.4.1.2.2 Zone Data type

La zone Data type est constituée des parties suivantes:

Types "Static Data"

Définition de types de données simples généraux

Types "Complex Data"

Définition de types de données structurés généraux

Types "Manufacturer Specific Complex Data"

Définition de types de données structurés spécifiques à un fabricant

Types "Profile Specific Static Data"

Définition de types de données simples spécifiques à un profil d'appareils

Types "Device Profile Specific Complex Data"

Définition de types de données structurés spécifiques à un profil d'appareils

Types "Enumeration Data"

Définition de types de données d'énumération spécifiques à un appareil

6.1.4.1.2.3 Zone "CoE Communication"

6.1.4.1.2.3.1 Device type (Type de l'appareil)

L'objet "Device type" (type d'appareils) a le paramètre suivant:

Paramètre

Device profile

Ce paramètre spécifie le profil d'appareils qui est utilisé pour l'appareil.

Additional information

Ce paramètre spécifie des informations complémentaires définies par le profil d'appareils.

6.1.4.1.2.3.2 Error register

L'objet "Error Register" (registre d'erreurs) a le paramètre suivant:

Paramètre

Generic error

Ce paramètre est mis à «true» en cas de présence d'une erreur générique.

Current error

Ce paramètre est mis à «true» en cas de présence d'une erreur liée au courant.

Voltage error

Ce paramètre est mis à «true» en cas de présence d'une erreur liée à la tension.

Temperature error

Ce paramètre est mis à «true» en cas de présence d'une erreur liée à la température.

Communication error

Ce paramètre est mis à «true» en cas de présence d'une erreur liée à la communication.

Device profile specific error

Ce paramètre est mis à «true» en cas de présence d'une erreur spécifique à un profil d'appareils.

Manufacturer specific error

Ce paramètre est mis à «true» en cas de présence d'une erreur spécifique à un fabricant.

6.1.4.1.2.3.3 Nom d'appareil attribué par le fabricant

L'objet "Manufacturer Device Name" (nom d'appareil de fabricant) a le paramètre suivant:

Paramètre

Device name

Ce paramètre spécifie le nom d'appareil relatif à l'appareil.

6.1.4.1.2.3.4 Hardware version

L'objet "Hardware Version" (version du matériel) a le paramètre suivant:

Paramètre

Hardware version

Ce paramètre spécifie la version de matériel pour l'appareil.

6.1.4.1.2.3.5 Software version

L'objet "Software Version" (version du logiciel) a le paramètre suivant:

Paramètre

Software version

Ce paramètre spécifie la version de logiciel pour l'appareil.

6.1.4.1.2.3.6 Identity

L'objet "Identity " (identité) a le paramètre suivant:

Paramètre

ID du fournisseur

Ce paramètre spécifie l'ID de vendeur de l'appareil.

Product code

Ce paramètre spécifie le code produit de l'appareil.

Major Revision Number

Ce paramètre spécifie le numéro de révision majeure de l'appareil qui identifie la fonctionnalité de l'appareil.

Minor Revision Number

Ce paramètre spécifie le numéro de révision mineure de l'appareil qui identifie une version différente avec la même fonctionnalité.

Numéro de série

Ce paramètre spécifie le numéro de série de l'appareil.

6.1.4.1.2.3.7 Mapping PDO en réception

L'objet "Receive PDO Mapping " («mapping d'objet de données processus en réception») a le paramètre suivant:

Paramètre

PDO Number

Ce paramètre spécifie le numéro du PDO.

Number of mapping entries

Ce paramètre spécifie le nombre d'entrées de mapping.

List of mapping entries

Ce paramètre spécifie une liste d'entrées de mapping.

6.1.4.1.2.3.8 Mapping de PDO en émission

L'objet "Transmit PDO Mapping " («mapping d'objets de données processus en émission») a le paramètre suivant:

Paramètre

PDO Number

Ce paramètre spécifie le numéro du PDO.

Number of mapping entries

Ce paramètre spécifie le nombre d'entrées de mapping.

List of mapping entries

Ce paramètre spécifie une liste d'entrées de mapping.

6.1.4.1.2.3.9 Type de communication du gestionnaire de synchronisation

L'objet "Sync Manager Communication Type" (type de communication du gestionnaire de synchronisation) a le paramètre suivant:

Paramètre

Number of Used Sync Manager entities

Ce paramètre spécifie le nombre d'entités de Sync Manager (gestionnaire de synchronisation) utilisées.

List of Sync Manager Communication Types

Ce paramètre spécifie le type de communication pour chaque entité de gestionnaire de synchronisation utilisée.

6.1.4.1.2.3.10 Attribution d'objet PDO du gestionnaire de synchronisation

L'objet "Sync Manager PDO assignment " (attribution d'objet de données de processus du gestionnaire de synchronisation) a le paramètre suivant:

Paramètre

Channel number

Ce paramètre spécifie le numéro de la voie du gestionnaire de synchronisation.

Number of assigned PDOs

Ce paramètre spécifie le nombre des PDO attribués.

List of assigned PDOs

Ce paramètre spécifie la liste des PDO attribués.

6.1.4.1.2.3.11 Synchronization timing («temporisation de la synchronisation»)

L'objet "Sync Manager Communication" (communication du gestionnaire de synchronisation) a le paramètre suivant:

Paramètre

Channel number

Ce paramètre spécifie le numéro de la voie du gestionnaire de synchronisation.

Type de synchronisation

Ce paramètre spécifie la méthode de synchronisation.

Durée de cycle

Ce paramètre spécifie la longueur cible d'un cycle.

Délai de changement

Ce paramètre spécifie un paramètre de décalage pour action synchronisée.

6.1.4.1.3 Interactions SDO

6.1.4.1.3.1 Services SDO

Les services SDO permettent de lire ou écrire les entrées du dictionnaire d'objets. Le protocole de transport de SDO permet d'émettre des objets de toute taille. Le protocole SDO est l'équivalent du protocole CANopen SDO pour prendre en charge la réutilisation de piles protocolaires existantes.

Le premier octet du premier segment spécifie les nécessaires informations de contrôle de flot. Les trois octets suivants du premier segment spécifient l'indice et le sous-indice de l'entrée du dictionnaire d'objets devant être lue ou écrite. Les octets suivants du premier segment sont disponibles pour les données d'utilisateur. Le deuxième segment et les suivants contiennent l'octet de contrôle et les données d'utilisateur. Le destinataire confirme chaque segment ou bloc de segments et, de fait, une communication d'homologue à homologue (client/serveur) a lieu.

En mode patrimonial, le protocole SDO consiste en 8 octets seulement pour concorder avec la taille de données CAN. En mode évolué, les données de charge utile sont simplement agrandies sans changer les en-têtes de protocole. Ainsi, la taille plus large de données de la boîte à lettres est appliquée au protocole SDO et l'émission de données volumineuses est accélérée en conséquence.

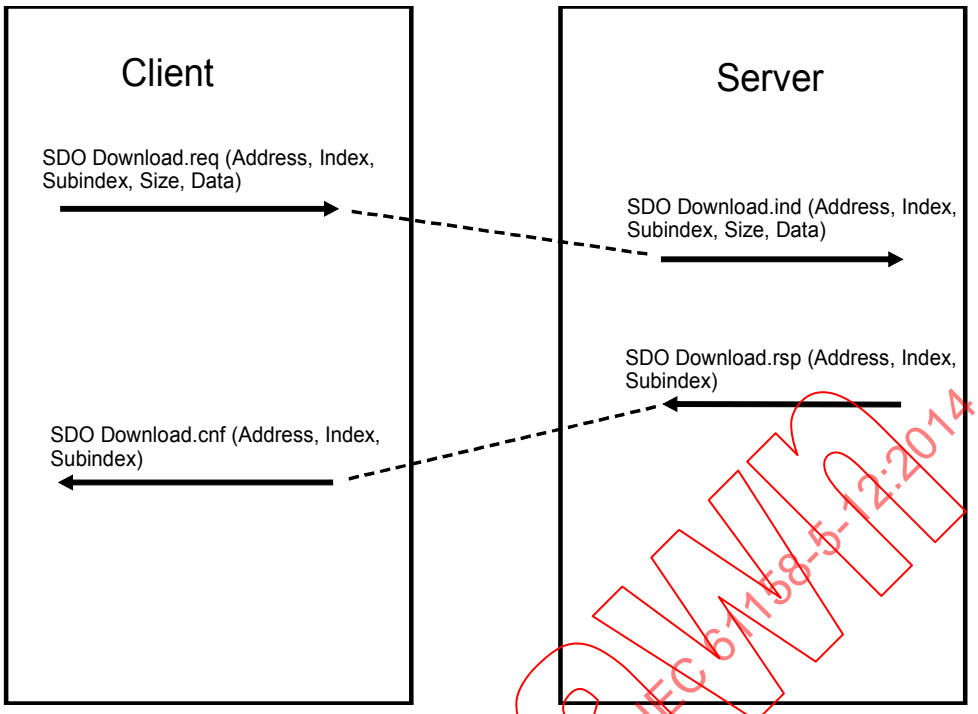
En outre, il est ajouté un mode qui permet de transférer en une seule fois la totalité des données situées à un indice du dictionnaire d'objets. Les données de tous les sous-indices sont transférées par la suite.

Les services confirmés (SDO Upload, SDO Download, Download SDO Segment et Upload SDO Segment) et le service non confirmé (Abort SDO Transfer) sont utilisés pour les objets de données de service accomplissant le transfert segmenté/accéléré. La classe de service SDO Download est divisée en services "SDO Download Expedited" et 'Initiate SDO Download Normal". La classe de service SDO Upload est divisée en services "SDO Upload Expedited" et 'Initiate SDO Upload Normal".

Les primitives des services de SDO sont mappées avec les primitives des services d'émission de boîte à lettres.

6.1.4.1.3.2 Séquence de SDO download (téléchargement de SDO)

La Figure 11 montre les primitives entre client et serveur dans le cas d'une séquence d'un seul SDO Download qui a réussi.

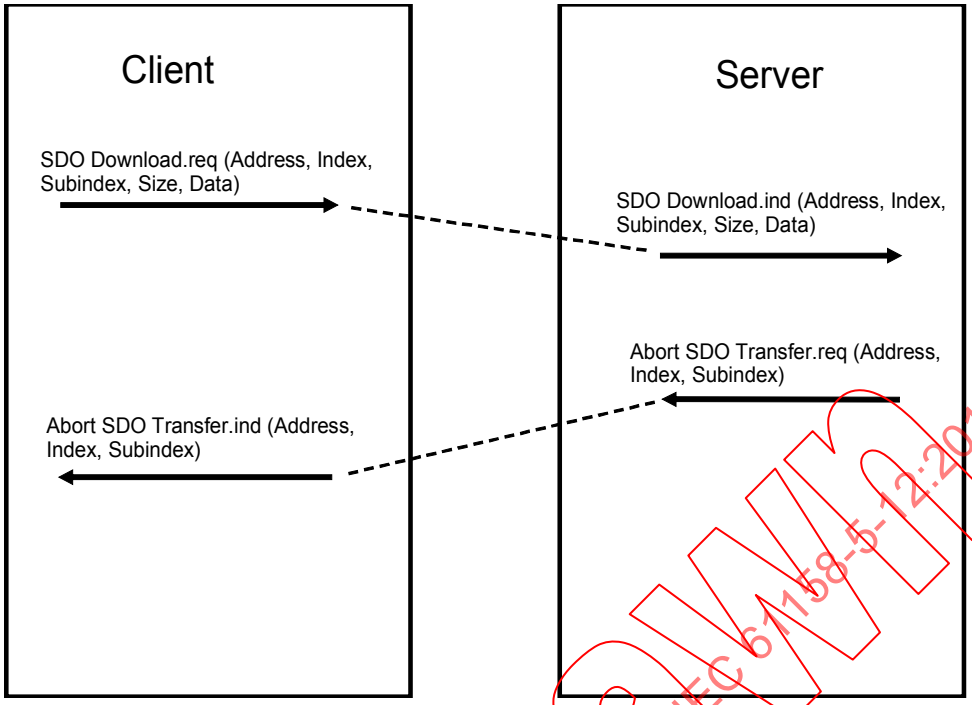


Légende

Anglais	Français
Client	Client
Server	Serveur
SDO Download.req (Address, Index, Subindex, Size, Data)	Demande de téléchargement de SDO (adresse, indice, sous-indice, taille, données)
SDO Download.ind (Address, Index, Subindex, Size, Data)	Indication de téléchargement de SDO (adresse, indice, sous-indice, taille, données)
SDO Download.rsp (Address, Index, Subindex)	Réponse de téléchargement de SDO (adresse, indice, sous-indice)
SDO Download.cnf (Address, Index, Subindex)	Confirmation de téléchargement de SDO (adresse, indice, sous-indice)

Figure 11 – Séquence d'un seul SDO-Download qui a réussi

La Figure 12 montre les primitives entre client et serveur dans le cas d'une séquence d'un SDO Download qui a échoué. Il convient que les paramètres de service Address, Index et Subindex soient les mêmes dans la demande Download (téléchargement) et dans la demande Abort (Abandon).

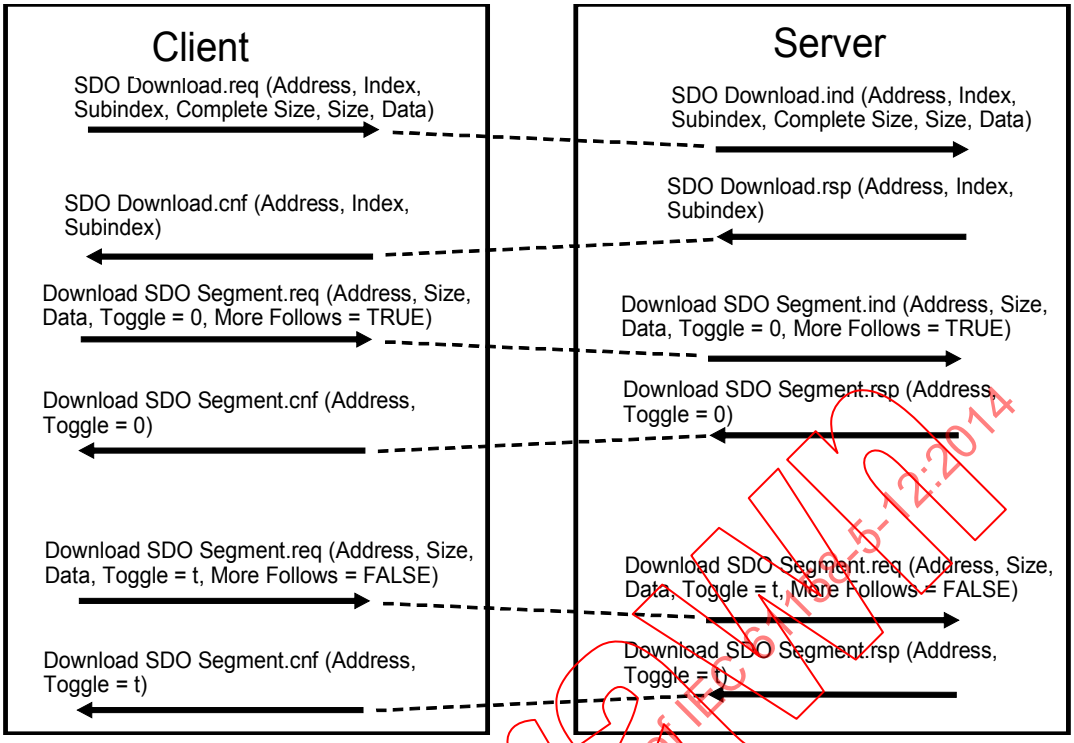


Légende

Anglais	Français
Client	Client
SDO Download.req (Address, Index, Subindex, Size, Data)	Demande de téléchargement de SDO (adresse, indice, sous-indice, taille, données)
Server	Serveur
SDO Download.ind (Address, Index, Subindex, Size, Data)	Indication de téléchargement de SDO (adresse, indice, sous-indice, taille, données)
Abort SDO Transfer.req (Address, Index, Subindex)	Demande d'abandon de transfert de SDO (adresse, indice, sous-indice)
Abort SDO Transfer.ind (Address, Index, Subindex)	Indication d'abandon de transfert de SDO (adresse, indice, sous-indice)

Figure 12 – Séquence d'un seul SDO-Download qui a échoué

La Figure 13 montre les primitives entre client et serveur dans le cas d'une séquence d'un SDO Download segmenté qui a réussi.



Légende

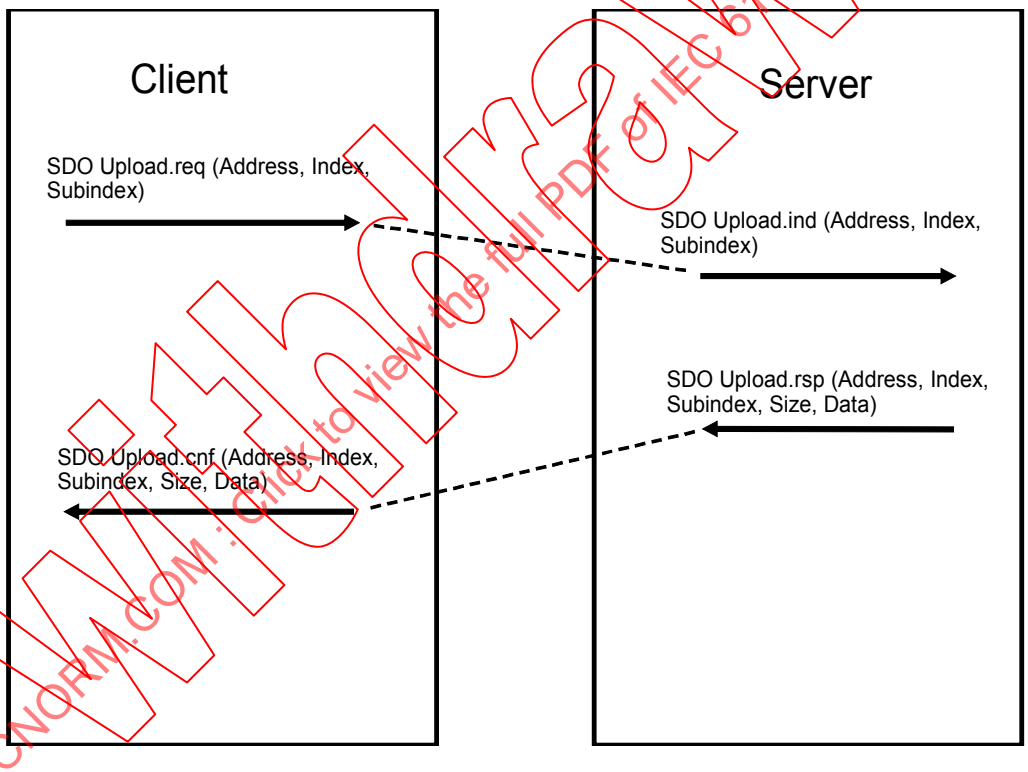
Anglais	Français
Client	Client
SDO Download.req (Address, Index, Subindex, Complete Size, Size, Data)	Demande de téléchargement de SDO (adresse, indice, sous-indice, taille complète, taille, données)
Server	Serveur
SDO Download.ind (Address, Index, Subindex, Complete Size, Size, Data)	Indication de téléchargement de SDO (adresse, indice, sous-indice, taille complète, taille, données)
SDO Download.rsp (Address, Index, Subindex)	Réponse de téléchargement de SDO (adresse, indice, sous-indice)
SDO Download.cnf (Address, Index, Subindex)	Confirmation de téléchargement de SDO (adresse, indice, sous-indice)
Download SDO Segment.req (Address, Size, Data, Toggle=0, More follows =TRUE)	Demande de téléchargement de segment de SDO (adresse, taille, données, bascule=0, à suivre= VRAI)
Download SDO Segment.ind (Address, Size, Data, Toggle=0, More follows =TRUE)	Indication de téléchargement de segment de SDO (adresse, taille, données, bascule=0, à suivre= VRAI)
Download SDO Segment.rep (Address, Toggle=0)	Réponse de téléchargement de segment de SDO (adresse, bascule=0,)
Download SDO Segment.cnf (Address, Toggle=0,)	Confirmation de téléchargement de segment de SDO (adresse, bascule=0)

Download SDO Segment.req (Address, Size, Data, Toggle=t, More follows =FALSE)	Demande de téléchargement de segment de SDO (adresse, taille, données, bascule=t, à suivre= FAUX)
Download SDO Segment.req (Address, Size, Data, Toggle=t, More follows =FALSE)	Demande de téléchargement de segment de SDO (adresse, taille, données, bascule=t, à suivre= FAUX)
Download SDO Segment .cnf (Address, Toggle=t)	Confirmation de téléchargement de SDO (adresse, bascule=t)
Download SDO Segment.rsp (Address, Toggle=t)	Réponse de téléchargement de SDO (adresse, bascule=t)

Figure 13 – Séquence d'un SDO-Download segmenté qui a réussi

6.1.4.1.3.3 SDO Upload Sequence

La Figure 14 montre les primitives entre client et serveur dans le cas d'une séquence d'un seul SDO Upload qui a réussi.

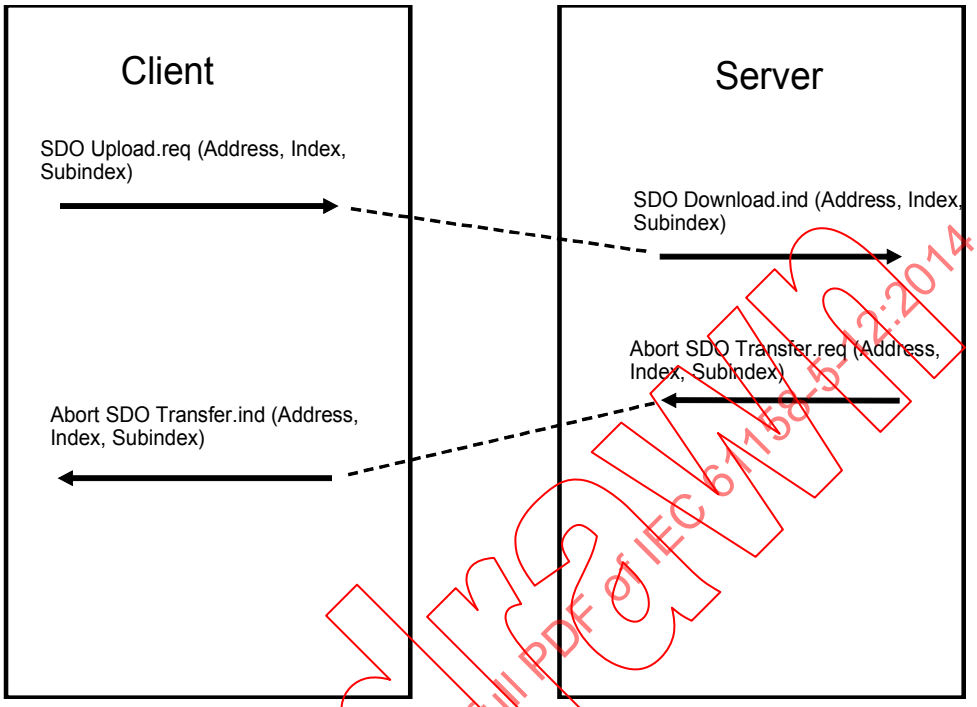


Légende

Anglais	Français
Client	Client
SDO Upload.req (Address, Index, Subindex)	Demande de chargement de SDO (adresse, indice, sous-indice)
Server	Serveur
SDO Upload.ind (Address, Index, Subindex)	Indication de chargement de SDO (adresse, indice, sous-indice)
SDO Upload.rsp (Address, Index, Subindex, Size, Data)	Réponse de chargement de SDO (adresse, indice, sous-indice, taille, données)
SDO Upload.cnf (Address, Index, Subindex, Size, Data)	Confirmation de chargement de SDO (adresse, indice, sous-indice, taille, données)

Figure 14 – Séquence d'un seul SDO-Upload qui a réussi

La Figure 15 montre les primitives entre client et serveur dans le cas d'une séquence d'un SDO Upload qui a échoué. Il convient que les paramètres de service Address, Index et Subindex soient les mêmes dans la demande Upload (chargement) et dans la demande Abort (Abandon).

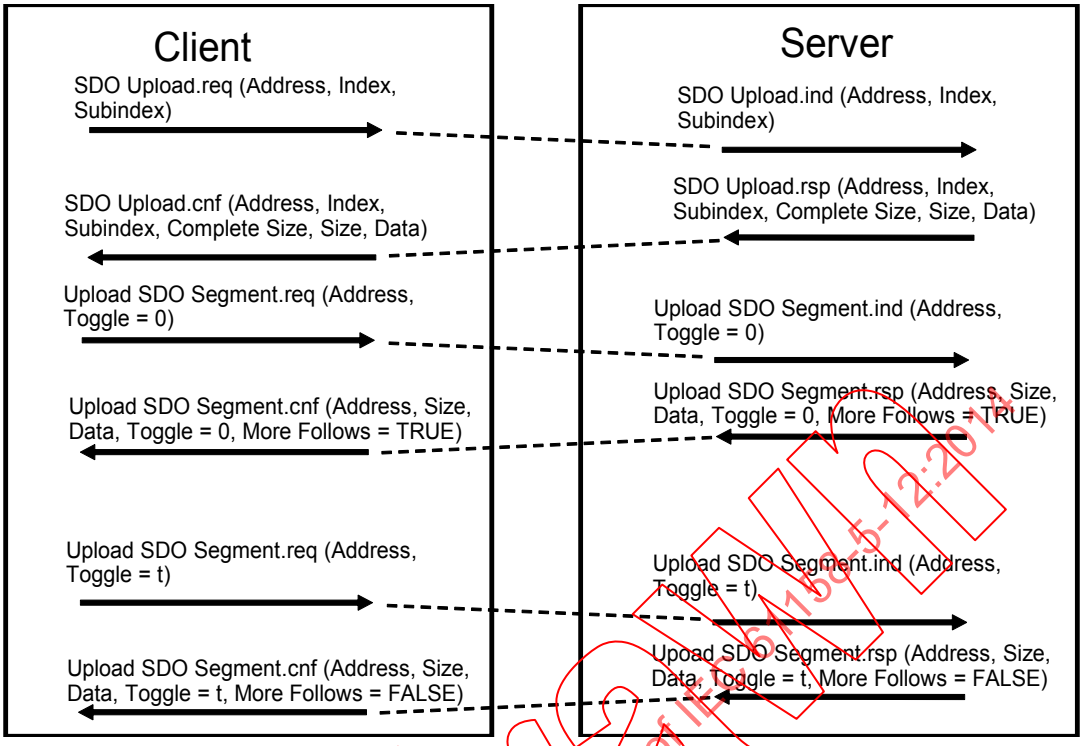


Légende

Anglais	Français
Client	Client
SDO Upload.req (Address, Index, Subindex)	Demande de chargement de SDO (adresse, indice, sous-indice)
Server	Serveur
SDO Download.ind (Address, Index, Subindex)	Indication de téléchargement de SDO (adresse, indice, sous-indice)
Abort SDO Transfer.req (Address, Index, Subindex)	Demande d'abandon de transfert de SDO (adresse, indice, sous-indice)
Abort SDO Transfer.ind (Address, Index, Subindex)	Indication d'abandon de transfert de SDO (adresse, indice, sous-indice)

Figure 15 – Séquence d'un seul SDO-Upload qui a échoué

La Figure 16 montre les primitives entre client et serveur dans le cas d'une séquence d'un SDO Upload segmenté qui a réussi.



Légende

Anglais	Français
Client	Client
SDO Upload.req (Address, Index, Subindex)	Demande de chargement de SDO (adresse, indice, sous-indice)
Server	Serveur
SDO Upload.ind (Address, Index, Subindex)	Indication de chargement de SDO (adresse, indice, sous-indice)
SDO Upload.rsp (Address, Index, Subindex, Complete Size, Size, Data)	Réponse de chargement de SDO (adresse, indice, sous-indice, taille complète, taille, données)
SDO Upload.cnf (Address, Index, Subindex, Complete Size, Size, Data)	Confirmation de chargement de SDO (adresse, indice, sous-indice, taille complète, taille, données)
Upload SDO Segment.req (Address, Toggle=0)	Demande de chargement de segment de SDO (adresse, bascule=0,)
Upload SDO Segment.ind (Address, Toggle=0)	Indication de chargement de segment de SDO (adresse, bascule=0,)
Upload SDO Segment.rsp (Address, Size, Data, Toggle=0, More follows =TRUE)	Réponse de chargement de segment de SDO (adresse, taille, données, bascule=0, à suivre= VRAI)
Upload SDO Segment.cnf (Address, Size, Data, Toggle=0, More follows =TRUE)	Confirmation de chargement de segment de SDO (adresse, taille, données, bascule=0, à suivre= VRAI)
Upload SDO Segment .req (Address, Toggle=t)	Demande de chargement de SDO (adresse, bascule=t)

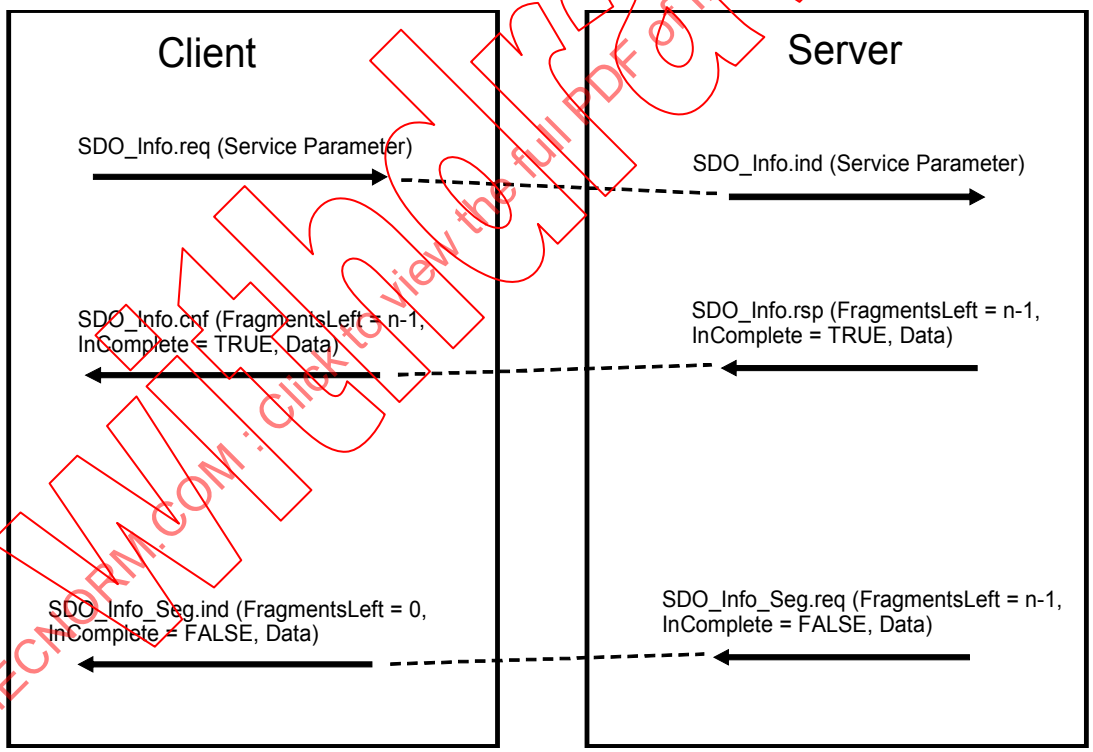
Upload SDO Segment .ind (Address, Toggle=t)	Indication de chargement de SDO (adresse, bascule=t)
Upload SDO Segment.rsp (Address, Size, Data, Toggle=t, More follows =FALSE)	Réponse de chargement de segment de SDO (adresse, taille, données, bascule=t, à suivre= FAUX)
Upload SDO Segment.cnf (Address, Size, Data, Toggle=t, More follows =FALSE)	Confirmation de chargement de segment de SDO (adresse, taille, données, bascule=t, à suivre= FAUX)

Figure 16 – Séquence d'un SDO-Upload segmenté qui a réussi

6.1.4.1.3.4 Services informations

Les services informations de SDO permettent au client de lire le dictionnaire d'objets d'un serveur. Les primitives des services informations de SDO sont mappées avec les primitives des services d'émission de boîte à lettres.

La Figure 17 montre les primitives entre client et serveur qui sont les mêmes pour tous les services informations de SDO désignés SDO_Info. La fragmentation est accomplie par un service SDO_Info_Seg.



Légende

Anglais	Français
Client	Client
Server	Serveur

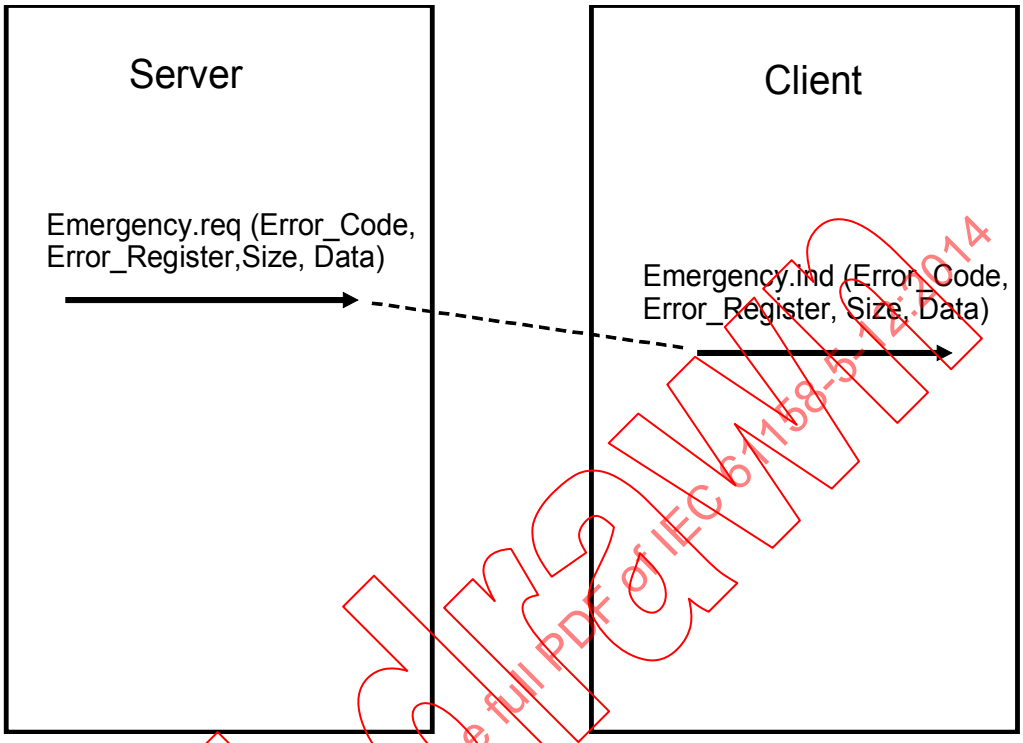
Figure 17 – Séquence d'informations de SDO

6.1.4.1.4 Interaction d'urgence

Les messages d'urgence sont déclenchés par l'occurrence d'une situation d'erreur interne à l'appareil. L'émission est exécutée par l'interface de la boîte à lettres.

Les primitives du service Emergency sont mappées avec les primitives des services d'émission de boîte à lettres.

La Figure 18 montre les primitives entre serveur et client dans le cas d'un service Emergency.



Légende

Anglais	Français
Client	Client
Server	Serveur

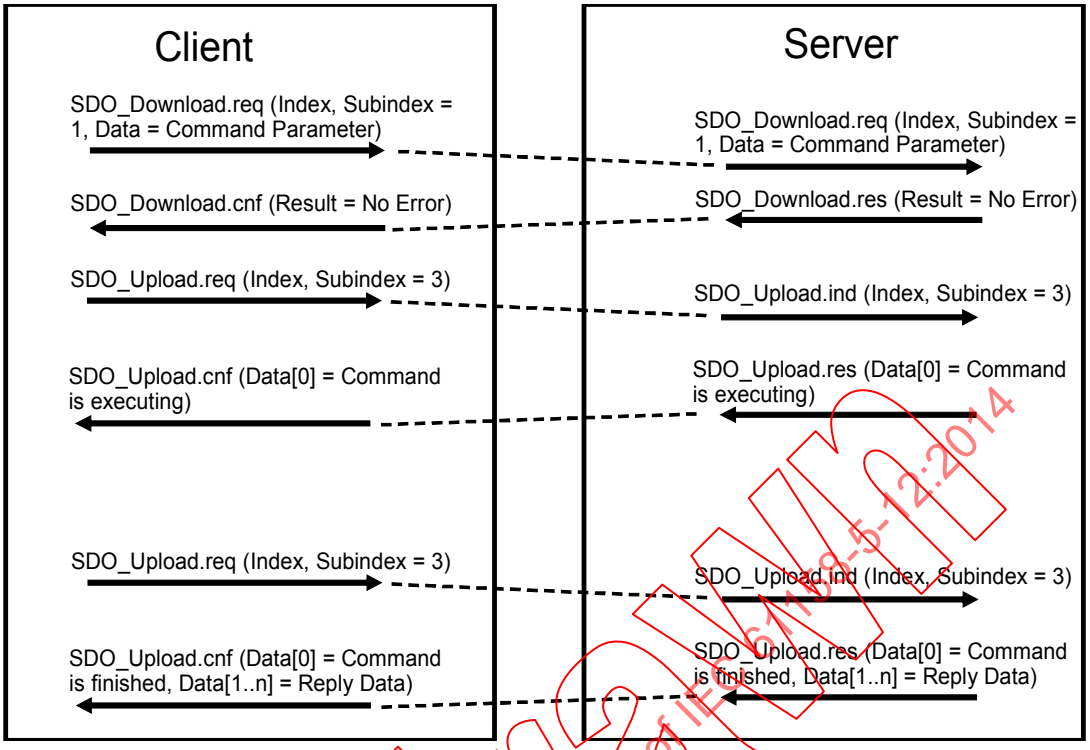
Figure 18 – Service 'Emergency' (Urgence)

6.1.4.1.5 Command

Les objets Command peuvent être utilisés pour des opérations où des données doivent être envoyées dans la demande et la réponse.

Une opération sera lancée dans le serveur par l'écriture d'un sous-indice 1 de l'objet Command avec les données de demande de commande par un service SDO Download. Le statut et les données de réponse de l'opération seront lus avec un SDO Upload au sous-indice 3 de l'objet Command.

La Figure 19 montre les primitives entre serveur et client dans le cas d'un service Command.



Légende

Anglais	Français
Client	Client
Server	Serveur

Figure 19 – Séquence de "Command"

6.1.4.1.6 Interaction de données de processus

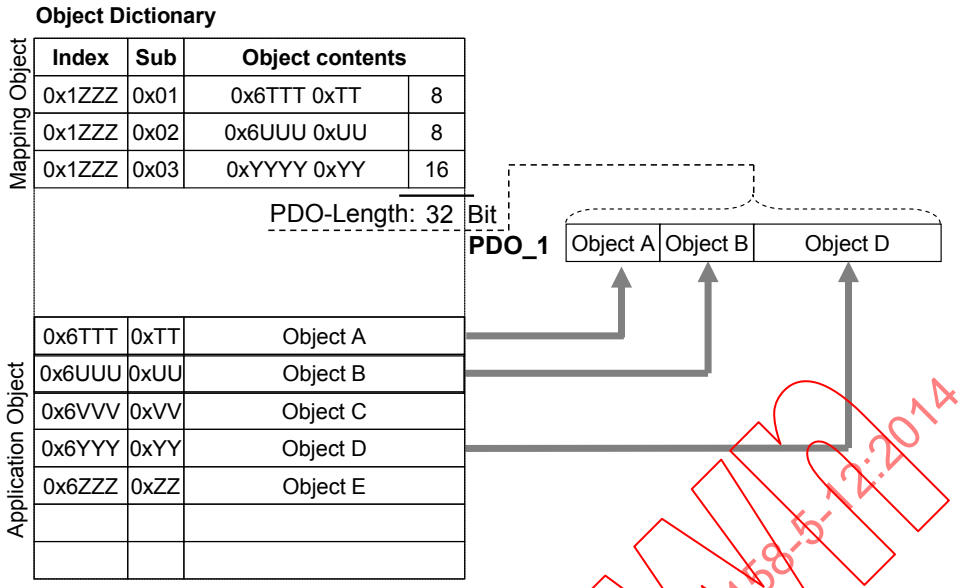
6.1.4.1.6.1 Mapping d'objets avec des données de processus

Les données de processus d'un esclave sont spécifiées dans l'ASE "Process data". Chaque objet "Sync Manager channel PDO assignment" décrit une zone cohérente à l'intérieur de l'ASE "Process data" et consiste en plusieurs (1 à 255) objets de données de processus.

Les objets de données de processus (PDO) consistent en des objets contenus dans le dictionnaire d'objets qui peuvent être mappés avec des PDO. Les objets "PDO mapping" («mapping de PDO») contiennent une liste de références d'objets ainsi que la longueur en bits.

Le mapping de PDO se rapporte au mapping d'objets d'application (données de processus temps réel) issus du dictionnaire d'objets avec les PDO. Les profils d'appareils fournissent un mapping par défaut pour chaque type d'appareils qui est approprié pour la plupart des applications. Par exemple, le mapping par défaut pour E/S numériques représente simplement les entrées et sorties selon leur ordre topologique dans les PDO d'émission et de réception.

Le mapping courant peut être lu au moyen d'entrées correspondantes contenues dans le répertoire d'objets. Celles-ci sont appelées les "tables de mapping". Le premier emplacement dans la table de mapping (sous-indice 0) spécifie le nombre d'objets mappés qui sont énumérés après lui. Les tables sont situées dans le répertoire d'objets à l'indice 0x1600 à 0x17FF pour les RxPDO et à l'indice 0x1A00 à 0x1BFF pour les TxPDO. La Figure 20 montre un exemple de mapping de PDO.



Légende

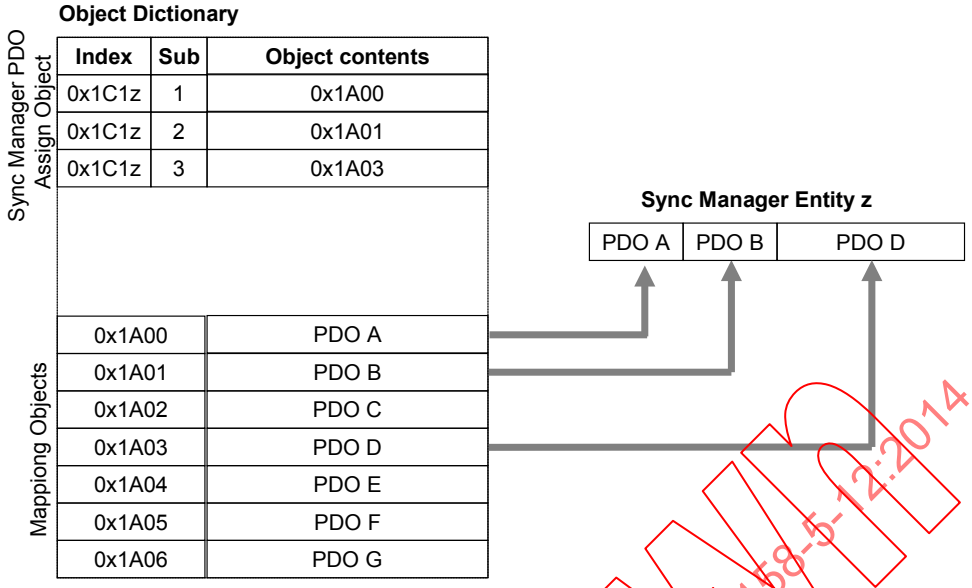
Anglais	Français
Object Dictionary	Dictionnaire d'objets
Index	Indice
Sub	Sous(-indice)
Object contents	Contenu d'objet
Mapping Object	Objet de mapping
PDO-Length: 32 Bit	Longueur de PDO : 32 bits
PDO_1	PDO_1
Object A	Objet A
Object B	Objet B
Object C	Objet C
Object D	Objet D
Object E	Objet E
Application objet	Objet application

Figure 20 – Mapping de PDO

La voie de gestionnaire de synchronisation (Sync Manager channel) est constituée de plusieurs PDO. Les objets "Sync Manager PDO assign" décrivent comment ces PDO sont reliés à un Sync Manager.

Pour les appareils avec un mapping configurable ou les appareils permettant un accès aux données de processus par différentes tâches de maître, il convient que les objets "Sync Manager PDO assign" soient inscriptibles.

L'assignation courante peut être lue au moyen d'entrées correspondantes contenues dans le répertoire d'objets. Le premier emplacement dans la table "Sync Manager PDO assignment" d'assignation de PDO de gestionnaire de synchronisation (sous-indice 0) spécifie le nombre des PDO assignés qui sont énumérés après lui. Les tables sont situées dans le répertoire d'objets à l'indice 0x1C10 à 0x1C2F. La Figure 21 montre un exemple d'assignation de PDO.



Légende

Anglais	Français
Object Dictionary	Dictionnaire d'objets
Index	Indice
Sub	Sous(-indice)
Object contents	Contenu d'objet
Sync Manager PDO Assign Object	Objet "Sync Manager PDO Assign" («Assignation des PDO au gestionnaire de synchronisation»)
Sync Manager Entity z	Entité gestionnaire de synchronisation z
Mapping Objects	Objets de mapping
PDO-Length: 32 Bit	Longueur de PDO : 32 bits
PDO A	PDO A (Objet de données de processus A)
PDO B	PDO B (Objet de données de processus B)
PDO C	PDO C (Objet de données de processus C)
PDO D	PDO D (Objet de données de processus D)
PDO E	PDO E (Objet de données de processus E)
PDO F	PDO F (Objet de données de processus F)
PDO G	PDO G (Objet de données de processus G)

Figure 21 – Sync manager PDO assignment (Assignation des PDO au gestionnaire de synchronisation)

Les esclaves avec communication de boîte à lettres prennent en charge la lecture des objets "PDO mapping" et "Sync Manager PDO assign". Pour les esclaves avec un mapping configurable, il y a deux possibilités pour la manière dont il convient que ce mapping soit pris en charge. Dans la première possibilité, l'esclave prend en charge différents mappings décrits dans les objets "PDO mapping" qui sont en lecture seule. Le maître configure le mapping réel en sélectionnant les objets "PDO mapping" souhaités lors de l'écriture des objets "Sync Manager PDO assign". La seconde possibilité pour des appareils plus évolués permet au maître d'écrire les objets "PDO mapping" pour obtenir une possibilité de configuration plus souple des mappings.

6.1.4.1.6.2 Objets de données de processus (PDO)

Chaque objet "Process data" a les paramètres suivants.

Paramètre

Number

Ce paramètre spécifie le numéro pour identifier le PDO.

Direction

Ce paramètre spécifie les sens Rx (maître vers esclave, client vers serveur) or Tx (esclave vers maître, serveur vers client) du PDO.

Number of mapped Objects

Ce paramètre spécifie le nombre d'objets mappés qui sont relatifs au PDO.

List of mapped Objects

Ce paramètre spécifie les objets connexes qui définissent le mapping du PDO.

6.1.4.1.6.3 Sync managerPDO assignment (Assignation d'objet PDO au gestionnaire de synchronisation)

Chaque objet "Sync Manager channel" a les paramètres suivants.

Paramètre

Number

Ce paramètre spécifie le numéro pour identifier la voie du gestionnaire de synchronisation (Sync Manager channel).

Direction

Ce paramètre spécifie le sens Output (maître vers esclave) ou Input (esclave vers maître) de la voie de Sync Manager.

Number of assigned PDOs

Ce paramètre spécifie le nombre des PDO attribués qui sont relatifs à la voie de Sync Manager.

List of assigned PDOs

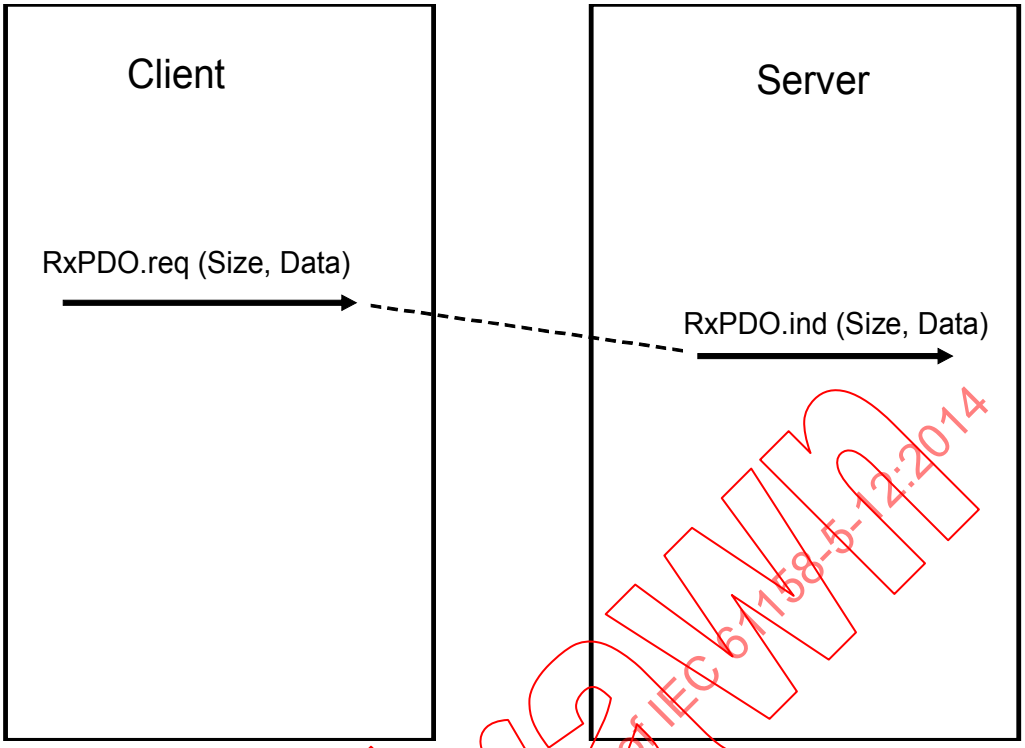
Ce paramètre spécifie les PDO attribués à la voie du gestionnaire de synchronisation (Sync Manager channel).

6.1.4.1.6.4 PDO transmission with CoE (Émission de PDO avec CoE)

Les services PDO permettent d'émettre de manière cyclique par l'interface de boîte à lettres les objets "Process data" (données de processus), et ce, du client vers le serveur (RxPDO) ou du serveur vers le client (TxPDO).

Les primitives des services de PDO sont mappées avec les primitives des services d'émission de boîte à lettres.

La Figure 22 montre les primitives entre client et serveur dans le cas d'un service RxPDO.

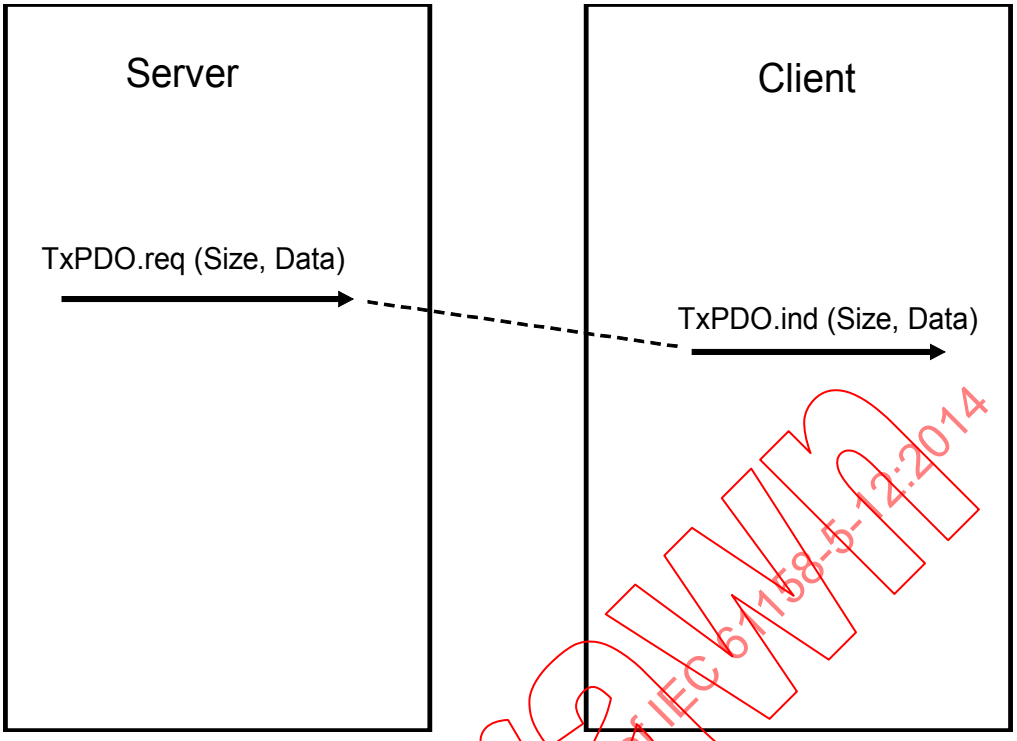


Légende

Anglais	Français
Client	Client
Server	Serveur

Figure 22 – Service RxPDO

La Figure 23 montre les primitives entre serveur et client dans le cas d'un service TxPDO.

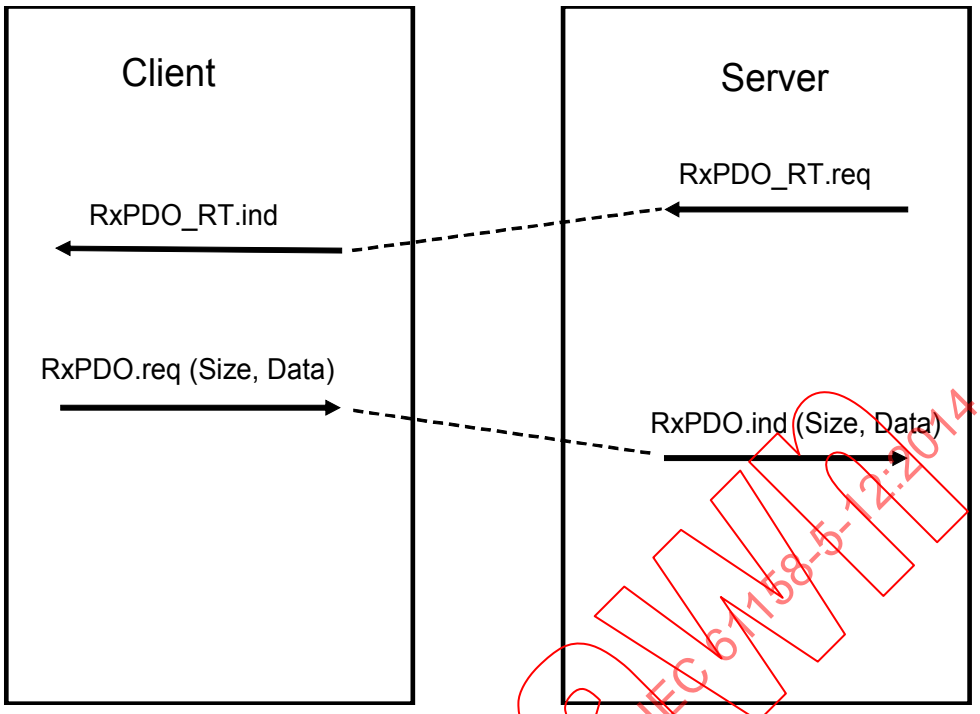


Légende

Anglais	Français
Client	Client
Server	Serveur

Figure 23 – Service TxPDO

La Figure 24 montre les primitives entre client et serveur dans le cas d'un service RxPDO Remote Transmission (émission distante d'objets de données de processus d'émission du client vers le serveur).

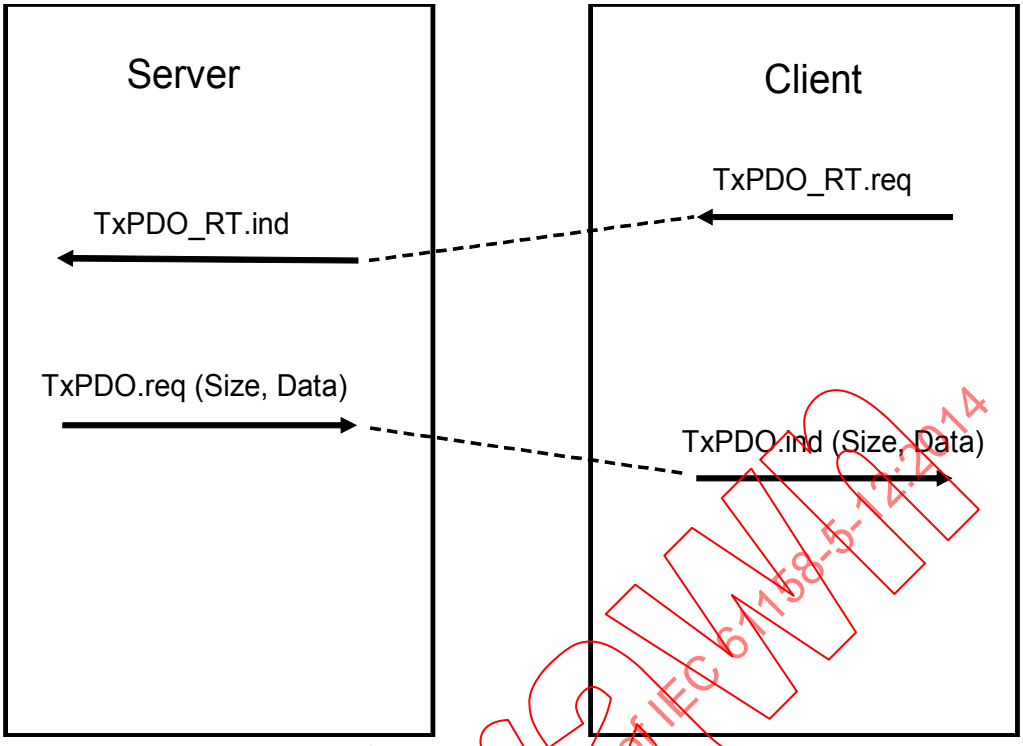


Légende

Anglais	Français
Client	Client
Server	Serveur

Figure 24 – Séquence "RxPDO remote transmission" (émission distante RxPDO)

La Figure 25 montre les primitives entre client et serveur dans le cas d'un service TxPDO Remote Transmission (émission distante d'objets de données de processus d'émission du serveur vers le client).



Légende

Anglais	Français
Client	Client
Server	Serveur

Figure 25 – Séquence "TxPDO remote transmission" (émission distante TxPDO)

6.1.4.2 Spécification de la classe CoE

6.1.4.2.1 Modèle formel

L'objet "CoE Object Dictionary" (Dictionnaire d'objets CoE) est décrit par le modèle suivant:

ASE: **CoE**

CLASSE: **CoE Object Dictionary**

CLASS ID: non utilisé

CLASSE PARENTE: TOP

ATTRIBUTS:

1.	(m)	Attribut-clé:	Implicit
2.	(m)	Attribut:	List of objects
2.1	(m)	Attribut:	Index (Indice)
2.2	(m)	Attribut:	List of Entry
2.2.1	(m)	Attribut:	Sous-indice
2.2.2	(m)	Attribut:	Bitlen
2.2.3	(o)	Attribut:	Acces
2.2.4	(o)	Attribut:	DataType
2.2.5	(o)	Attribut:	DefaultValue
2.2.6	(o)	Attribut:	MinValue
2.2.7	(o)	Attribut:	MaxValue
2.2.8	(o)	Attribut:	Description

SERVICES:

1.	(o)	OpsService:	SDO Download Expedited (Téléchargement express de l'objet SDO)
2.	(o)	OpsService:	SDO Download Normal (Téléchargement normal de l'objet SDO)
3.	(o)	OpsService:	Download SDO Segment (Téléchargement de segment SDO)
4.	(m)	OpsService:	SDO Upload Expedited (Chargement express de l'objet SDO)
5.	(m)	OpsService:	SDO Upload Normal (Chargement normal de l'objet SDO)
6.	(o)	OpsService:	Upload SDO Segment (Chargement de segment SDO)
7.	(m)	OpsService:	Abort SDO Transfer (Abandon du transfert d'objet SDO)

8.	(m)	OpsService:	Get OD List (Obtention de la liste de dictionnaires OD)
9.	(o)	OpsService:	Get Object Description (Obtention de la description d'objet)
10.	(o)	OpsService:	Get Entry Description (Obtention de la description d'entrée)
11.	(o)	OpsService:	OD List Segment (Segment de liste de dictionnaires OD)
12.	(o)	OpsService:	Object entry segment
13.	(o)	OpsService:	Emergency (Urgence)
14.	(o)	OpsService:	RxPDO
15.	(o)	OpsService:	TxPDO
16.	(o)	OpsService:	RxPDO remote transmission
17.	(o)	OpsService:	TxPDO remote transmission

6.1.4.2.2 Attributs

Implicit

L'attribut "Implicit" indique que l'objet "CoE Object Dictionary" est adressé de façon implicite par les services.

List of objects
Un objet se compose des éléments de liste suivants:

Index
Cet attribut spécifie un indice numérique de l'objet.

List of Entry
Cet attribut est constitué des attributs suivants.

Subindex
Cet attribut spécifie la référence à l'entrée d'objet.

Bitlen
Cet attribut spécifie la longueur en bits de la valeur de l'entrée d'objet.

Pour les PDO, il convient que la longueur soit ≤ 11 888.

Access
Cet attribut spécifie les droits d'accès à l'entrée en question. Ses valeurs permises sont

- RP (Read in Pre-Operational, lecture dans l'état pré-opérationnel)
- RS (Read in Safe-Operational, lecture dans l'état opérationnel en toute sécurité)
- RO (Read in Operational, lecture dans l'état opérationnel)
- WP (Write in Pre-Operational, écriture dans l'état pré-opérationnel)
- WS (Write in Safe-Operational, écriture dans l'état opérationnel en toute sécurité)
- WO (Write in Operational, écriture dans l'état opérationnel)
- RX (utilisable pour les RxPDO)
- TX (utilisable pour les TXPDO)

DataType
Cet attribut facultatif spécifie le type de données de la valeur de l'entrée d'objet.