IEC/PAS 61158-5-22

Edition 1.0 2009-08

PUBLICLY AVAILABLE SPECIFICATION

PRE-STANDARD

Industrial communication networks – Fieldbus specifications –
Part 5-22: Application layer service definition – Type SNpTYPE elements

IEC/PAS 61158-5-22:2009(E)

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

About the IEC

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

About IEC publications

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

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 It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub
Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

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

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

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

IEC/PAS 61158-5-22

Edition 1.0 2009-08

PUBLICLY AVAILABLE SPECIFICATION

PRE-STANDARD

Industrial communication networks – Fieldbus specifications –
Part 5-22: Application layer service definition – Type SNpTYPE elements

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE XC

ICS 25.040.40; 35.100.70

ISBN 978-2-88910-796-4

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 5-22: Application layer service definition – Type SNpTYPE elements

FOREWORD

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

A PAS is a technical specification not fulfilling the requirements for a standard, but made available to the public.

IEC-PAS 61158-5-22 has been processed by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

The text of this PAS is based on the following document:	This PAS was approved for publication by the P-members of the committee concerned as indicated in the following document
Draft PAS	**Report on voting**
65C/530/PAS	65C/534/RVD

Following publication of this PAS, which is a pre-standard publication, the technical committee or subcommittee concerned may transform it into an International Standard.

This PAS shall remain valid for an initial maximum period of 3 years starting from the publication date. The validity may be extended for a single 3-year period, following which it shall be revised to become another type of normative document, or shall be withdrawn.

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

INTRODUCTION

This PAS contains an additional profile – SNpTYPE – which may be integrated into a future new edition of the IEC 61158-5 series.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 5-22: Application layer service definition – Type SNpTYPE elements

1 Scope

1.1 Overview

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 part of IEC 61158-5 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 SNpTYPE 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 part of IEC 61158-5 defines in an abstract way the externally visible service provided by 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 part of IEC 61158-5 is to define the services provided to

1) the FAL user at the boundary between the user and the application layer of the fieldbus reference model, and

2) Systems Management at the boundary between the application layer and Systems Management of the fieldbus reference model.

This part of IEC 61158-5 specifies the structure and services of the 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 part of IEC 61158-5 to provide access to the FAL to control certain aspects of its operation.

1.2 Specifications

The principal objective of this part of IEC 61158-5 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 part of IEC 61158-5 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 the application layer services as defined in this part of IEC 61158-5.

2 Normative references

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

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

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

IEC 61158-4-22, *Industrial communication networks – Fieldbus specifications – Part 4-SNpTYPE: Data-link layer protocol specification – Type SNpTYPE elements*

IEC 61158-6-22, *Industrial communication networks - Fieldbus specifications - Part 6-SNpTYPE: Application layer protocol specification - Type SNpTYPE elements*

ISO/IEC 646, *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 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/IEC 8822, *Information Technology – Open Systems Interconnection – Presentation service definition*

ISO/IEC 8824-1, *Information Technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*

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

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

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

IETF RFC 791, *Internet Protocol*

3 Terms, definitions, abbreviations, symbols and conventions

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

3.1 ISO/IEC 7498-1 terms

a) application entity
b) application process
c) application protocol data unit
d) application service element
e) application entity invocation
f) application process invocation
g) application transaction
h) real open system
i) transfer syntax

3.2 ISO/IEC 8822 terms

For the purposes of this document, the following terms as defined in ISO/IEC 8822 apply:

a) abstract syntax
b) presentation context

3.3 ISO/IEC 9545 terms

For the purposes of this document, the following terms as defined in ISO/IEC 9545 apply:

a) application-association
b) application-context
c) application context name
d) application-entity-invocation
e) application-entity-type

f) application-process-invocation

g) application-process-type

h) application-service-element

i) application control service element

3.4 ISO/IEC 8824 terms

For the purposes of this document, the following terms as defined in ISO/IEC 8824 apply:

a) object identifier

b) type

3.5 Fieldbus application-layer specific definitions

3.5.1
application
function for which data is exchanged

3.5.2
application object
representation of a particular component within a device

3.5.3
acyclic data
data which is transferred from time to time for dedicated purposes

3.5.4
bit
unit of information consisting of a 1 or a 0. This is the smallest data unit that can be transmitted

3.5.5
cell
synonym for a single DL-segment which uses RTFL communication model

3.5.6
channel
path provided for conveying data

3.5.7
client
object which uses the services of a server by initiating a message to perform a task

3.5.8
communication cycle
fixed time period between which the root device issues empty frames for cyclic communication initiation in which data is transmitted utilizing CDC and MSC

3.5.9
connection
logical binding between two application objects

3.5.10
cycle time
duration of a communication cycle

3.5.11
cyclic
events which repeat in a regular and repetitive manner

3.5.12
cyclic communication
periodic exchange of telegrams

3.5.13
cyclic data
data which is transferred in a regular and repetitive manner for dedicated purposes

3.5.14
cyclic data channel (CDC)
part of one or more frames, which is reserved for cyclic data

3.5.15
data
generic term used to refer to any information carried over a fieldbus

3.5.16
device
physical entity connected to the fieldbus

3.5.17
error
discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition

3.5.18
error code
identification number of a specific type of error

3.5.19
gateway
device acting as a linking element between different protocols

3.5.20
index
position of an object within the object dictionary

3.5.21
inter-cell communication
communication between a RTFL device and a RTFN device or communication between a RTFL device and another RTFL device in different cells linked by RTFN

3.5.22
interface
shared boundary between two functional units, defined by functional characteristics, signal characteristic, or other characteristics as appropriate

3.5.23
intra-cell communication
communication between a RTFL device and another RTFL device in the same cell

3.5.24
mapping parameters
set of values defining the correspondence between application objects and process data objects

3.5.25
master clock
global time base for the PCS mechanism

3.5.26
message
ordered sequence of octets intended to convey data

3.5.27
message channel (MSC)
part of one or more frames, which is reserved for acyclic data

3.5.28
network
set of devices connected by some type of communication medium, including any intervening repeaters, bridges, routers and lower-layer gateways

3.5.29
ordinary device (OD)
slave in the communication system, which utilizes RTFL for cyclic and acyclic data interchange with other ODs in the same logical double line

3.5.30
precise clock synchronization (PCS)
mechanism to synchronize clocks of RTFL devices and maintain a global time base

3.5.31
process data
data designated to be transferred cyclically or acyclically for the purpose of processing

3.5.32
process data object
dedicated data object(s) designated to be transferred cyclically or acyclically for the purpose of processing

3.5.33
protocol
convention about the data formats, time sequences, and error correction in the data exchange of communication systems

3.5.34
root device (RD)
master in the communication system, which organises, initiates and controls the RTFL cyclic and acyclic data interchange for one logical double line

3.5.35
real time frame line (RTFL)
communication model for communication with high real time requirements

3.5.36
real time frame network (RTFN)
communication model for communication with low real time requirements

3.5.37
round trip time
transmission time needed by a DLPDU from the RD to the last OD in forward and backward direction

3.5.38
sub-index
sub-position of an individual element of an object within the object dictionary

3.5.39
timing signal
time-based indication of the occurrence of an event, commonly as an interrupt signal, used for DL-user synchronization

3.5.40
topology
physical network architecture with respect to the connection between the stations of the communication system

3.6 Abbreviations and symbols

AE	Application entity
AL	Application layer
ALME	Application layer management entity
AP	Application process
APDU	Application layer protocol data unit
APO	Application process object
AR	Application relationship
AREP	Application relationship end point
ASE	Application service element
CAN	Controller area network
CDC	Cyclic data channel
CDCL	CDC line
CDCN	CDC network
CeS	CANopen expands Type SNpTYPE
CL	Communication layer
Cnf	Confirmation
DA	Device address

DHCP	Dynamic Host Configuration Protocol
DL-	Data-link layer (as a prefix)
DLL	DL-layer
DLPDU	DL-protocol data unit
EDS	Electronic data sheet
EMCY	Emergency
FAL	Fieldbus application layer
GW	Gateway
ID	Identification
IETF	Internet Engineering Task Force
Ind	Indication
IP	Internet protocol
IPv4	IP version 4
IPv6	IP version 6
IRQ	Interrupt request
LME	Layer management entity
MAC	Medium access control
MC	Master clock
MII	Media independent interface
MSC	Message channel
MSCL	MSC line
MSCN	MSC network
OD	Ordinary device
OS	Operating system
OSI	Open systems interconnection
PCS	Precise clock synchronization

PDO	Process data object
PHY	Physical interface controller
PID	Packet ID
PTMNSU	Point-to-multipoint network-scheduled unconfirmed
PTPNSC	Point-to-point network-scheduled confirmed
PTPNSU	Point-to-point network-scheduled unconfirmed
PTPUTC	Point-to-point user-triggered confirmed
RD	Root device
Req	Request
RFC	Request for comments
Rsp	Response
RTF	Real time frame
RTFL	Real time frame line
RTFN	Real time frame network
RO	Read only
RW	Read and write access
Rx	Receive direction
RxPDO	Receive PDO
SDO	Service data object
SEF	Standard Ethernet frame
StdErr	Standard error output
StdIn	Standard input
StdOut	Standard output
SYNC	Synchronization
TCP	Transmission control protocol
TT	Transmission type

Tx	Transmit direction
TxPDO	Transmit PDO
UDP	User datagram protocol
WO	Write only

3.7 Conventions

3.7.1 Overview

The FAL is defined as a set of object-oriented ASEs. Each ASE is specified in a separate sub-clause. Each ASE specification is composed of two parts, its class specification, and its service specification.

The class specification defines the attributes of the class. The attributes are accessible from instances of the class using the Object Management ASE services specified in Clause 5 of this part of IEC 61158-5. The service specification defines the services that are provided by the ASE.

3.7.2 General conventions

This part of IEC 61158-5 uses the descriptive conventions given in ISO/IEC 10731.

3.7.3 Conventions for class definitions

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

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

3 (m) MgtService: service name

(1) The "FAL ASE:" entry is the name of the FAL ASE that provides the services for the class being specified.

(2) The "CLASS:" entry is the name of the class being specified. All objects defined using this template will be an instance of this class. The class may be specified by this part of IEC 61158-5, or by a user of this part of IEC 61158-5.

(3) The "CLASS ID:" entry is a number that identifies the class being specified. This number is unique within the FAL ASE that will provide the services for this class. When qualified by the identity of its FAL ASE, it unambiguously identifies the class within the scope of the FAL. The value "NULL" indicates that the class cannot be instantiated. Class IDs between 1 and 255 are reserved by this part of IEC 61158-5 to identify standardized classes. They have been assigned to maintain compatibility with existing national standards. CLASS IDs between 256 and 2048 are allocated for identifying user defined classes.

(4) The "PARENT CLASS:" entry is the name of the parent class for the class being specified. All attributes defined for the parent class and inherited by it are inherited for the class being defined, and therefore do not have to be redefined in the template for this class.

NOTE The parent-class "TOP" indicates that the class being defined is an initial class definition. The parent class TOP is used as a starting point from which all other classes are defined. The use of TOP is reserved for classes defined by this part of IEC 61158-5.

(5) The "ATTRIBUTES" label indicate that the following entries are attributes defined for the class.

a) Each of the attribute entries contains a line number in column 1, a mandatory (m) / optional (o) / conditional (c) / selector (s) indicator in column 2, an attribute type label in column 3, a name or a conditional expression in column 4, and optionally a list of enumerated values in column 5. In the column following the list of values, the default value for the attribute may be specified.

b) Objects are normally identified by a numeric identifier or by an object name, or by both. In the class templates, these key attributes are defined under the key attribute.

c) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting is used to specify

i) fields of a structured attribute (4.1, 4.2, 4.3),

ii) attributes conditional on a constraint statement (5). Attributes may be mandatory (5.1) or optional (5.2) if the constraint is true. Not all optional attributes require constraint statements as does the attribute defined in (5.2).

iii) the selection fields of a choice type attribute (6.1 and 6.2).

(6) The "SERVICES" label indicates that the following entries are services defined for the class.

a) An (m) in column 2 indicates that the service is mandatory for the class, while an (o) indicates that it is optional. A (c) in this column indicates that the service is conditional. When all services defined for a class are defined as optional, at least one has to be selected when an instance of the class is defined.

b) The label "OpsService" designates an operational service (1).

c) The label "MgtService" designates an management service (2).

d) The line number defines the sequence and the level of nesting of the line. Each

nesting level is identified by period. Nesting within the list of services is used to specify services conditional on a constraint statement.

3.7.4 Conventions for service definitions

3.7.4.1 Overview

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

3.7.4.2 Service parameters

Service primitives are used to represent service user/service provider interactions (ISO/IEC 10731). They convey parameters which indicate information available in the user/provider interaction. In any particular interface, not all parameters need be explicitly stated.

The service specifications of this part of IEC 61158-5 use a tabular format to describe the component parameters of the ASE service primitives. The parameters which apply to each group of service primitives are set out in tables. Each table consists of up to five columns for the:

1) parameter name;
2) request primitive;
3) indication primitive;
4) response primitive; and
5) confirm primitive.

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

M parameter is mandatory for the primitive.

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

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

(blank) parameter is never present.

S parameter is a selected item.

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

a) a parameter-specific constraint:

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

b) an indication that some note applies to the entry:

   "(n)" indicates that the following note "n" contains additional information pertaining to the parameter and its use.

3.7.4.3 Service procedures

The procedures are defined in terms of:

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

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

4 Concepts

4.1 Common concepts

All of IEC/TR 61158-1:2007, Clause 9 is incorporated by reference, except as specifically overridden in 4.2.

4.2 Type specific concepts

4.2.1 Operating principle

Type SNpTYPE consists of two types of communication models: RTFL and RTFN. RTFL is used to ensure synchronized cyclic real-time communication. RTFN is used in to network several RTFL cells to an overall system providing data interchange between several RTFL cells and between RTFL cells and RTFN devices.

In this context, a RTFL cell describes a DL-segment which uses RTFL for communication. An RTFL cell consists of a root device (RD) and one or several ordinary devices (OD). The central RTFL cell element is the root device which organizes and controls RTFL cell sequences such as cyclic real-time frame sending. A RTFL RD has at least one connection to RTFL, and can include a gateway (GW) which additionally has connection to RTFN. As each OD in the RTFL cell can only have a RTFL connection, the RD incorporating a GW therefore operates as a link between RTFL and RTFN. RTFN communication is not coordinated like communication in RTFL, but utilized by a switched fully duplex standard Ethernet. Thus, no determinism can be guaranteed for RTFN data transfer.

Communication of process and service data is accommodated by Type SNpTYPE networks using different mechanisms (channels) in RTFL and RTFN. Cyclic data can be transferred over the cyclic data channel (CDC). The message channel (MSC) allows additional acyclic data communication and is used for service data exchange.

Service data is typically transferred acyclic and is used for transfer of parameters, control commands, status and diagnostic data as well as for generally larger data segments. Service data are transferred either event driven or user driven (acyclic character). Parameter data used in particular in device configuration do not require strict time conditions whereas diagnostic data may have much greater time requirements.

In contrast, process data is typically transferred cyclically with different cycle times and higher real-time requirements.

Type SNpTYPE AL supports a variety of services and protocols to meet these differing requirements. Both communication models support the same fieldbus application layer. The services and protocols are mapped to the corresponding DL-services.

4.2.2 Communication model overview

4.2.2.1 Overview

Type SNpTYPE technology essentially specifies two communication models with corresponding protocols. RTFL communication is intended for fast machine communication

while RTFN provides for the networking of individual machines or cells. The corresponding protocols aim to offer an equal set of services for cyclic process data exchange as well as for acyclic message data communication.

The application relationship can be modeled independent of communication relationship.

4.2.2.2 Communication model RTFL

For RTFL communication model, communication follows a line topology. RTFL communication is based on cyclic data transfer in an ISO/IEC 8802-3 Ethernet frame. This basic cyclic data transfer is provided by a special device, the root device (RD). Root devices act as communication master to cyclically initiate communication. The Ethernet frames originated by the root device are passed to the Type SNpTYPE ordinary devices (OD). Each ordinary device receives the frame, writes its data and passes the frame on. A RTFL network requires exactly one root device. The last ordinary device of a RTFL network sends the processed frame back. The frame is transferred back in reverse device order to the root device so that it is returned by the first ordinary device to the root device as response frame. In backward direction, the ordinary devices read their relevant data from the frame.

4.2.2.3 Communication model RTFN

For RTFN communication model, communication is based on point to point connections between participating devices.

Networking of different RTFL parts or cells of an automation system into an overall automation system is supported by the usage of RTFN communication and corresponding gateways.

4.2.3 Application layer element description

4.2.3.1 CeS

The mandatory CeS ASE consists of several attributes and depicts the main application layer element to build up a distributed real-time application.

4.2.3.2 Standard Ethernet frame (SEF) communication

The optional SEF communication ASE depicts a possibility to utilize tunneled non Type SNpTYPE communication within the RTFL communication system.

4.2.3.3 Management

The mandatory management ASE consists of a set of services to control the state of a network and participating devices. Constraints in available services are specified for the different communication models RTFL and RTFN.

4.2.4 Producer-consumer interaction

The producer-consumer interaction model involves one producer and zero or more consumer(s). The model is characterized by an unconfirmed service requested by the producer and a correlated service indication in all consumers. Figure 1 illustrates the interaction for one producer and two consumers.

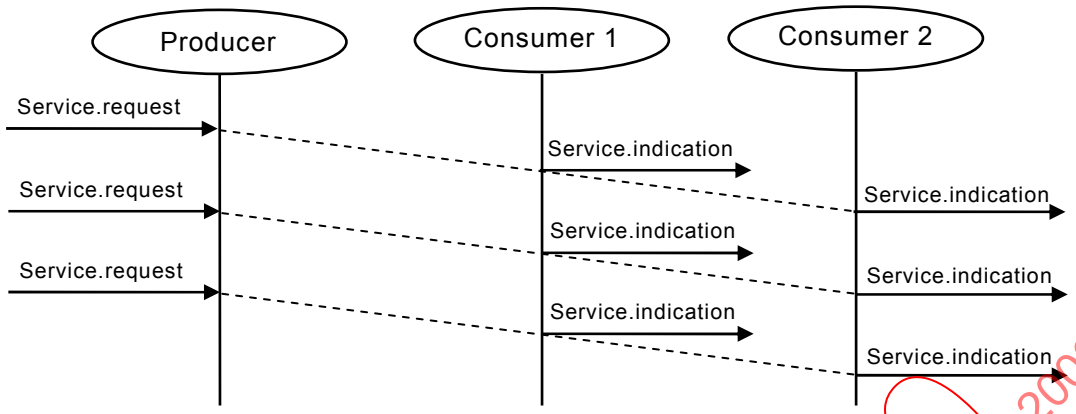

Figure 1 – Producer-consumer interaction model

The services supported by an interaction model are conveyed by application relationship endpoints (AREPs) associated with the communicating APs. The role that the AREP plays in the interaction (for example producer, consumer) is defined as an attribute of the AREP.

4.2.5 Device reference models

4.2.5.1 RTFL device reference model

Type SNpTYPE services are described using the principles, methodology and model of ISO/IEC 7498-1 (OSI). The OSI model provides a layered approach to communications standards, whereby the layers can be developed and modified independently. The Type SNpTYPE specification defines functionality from top to bottom of a full OSI model. Functions of the intermediate OSI layers, layers 3 to 6, are consolidated into either the Type SNpTYPE data-link layer or the Type SNpTYPE application layer. The device reference model for a Type SNpType RTFL device is shown in Figure 2.

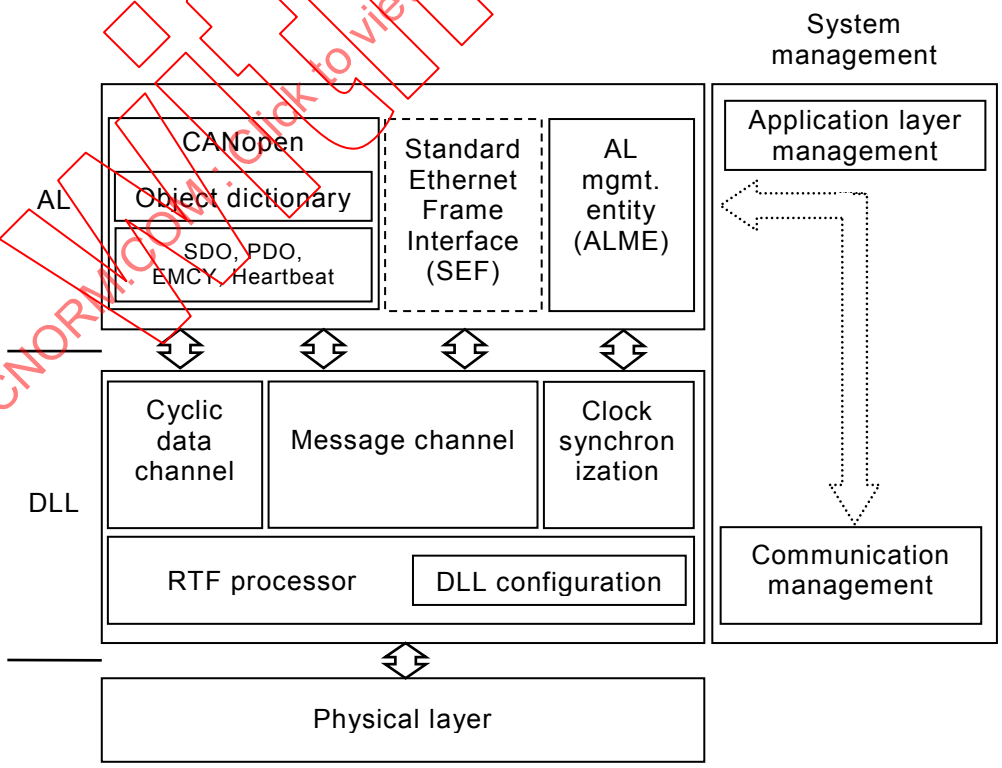

Figure 2 – RTFL device reference model

4.2.5.2 RTFN device reference model

Type SNpTYPE services are described using the principles, methodology and model of ISO/IEC 7498-1 (OSI). The OSI model provides a layered approach to communications standards, whereby the layers can be developed and modified independently. The Type SNpTYPE specification defines functionality from top to bottom of a full OSI model. Functions of the intermediate OSI layers, layers 3 to 6, are consolidated into either the Type SNpTYPE data-link layer or the Type SNpTYPE application layer. The device reference model for a Type SNpType RTFN device is shown in Figure 3.

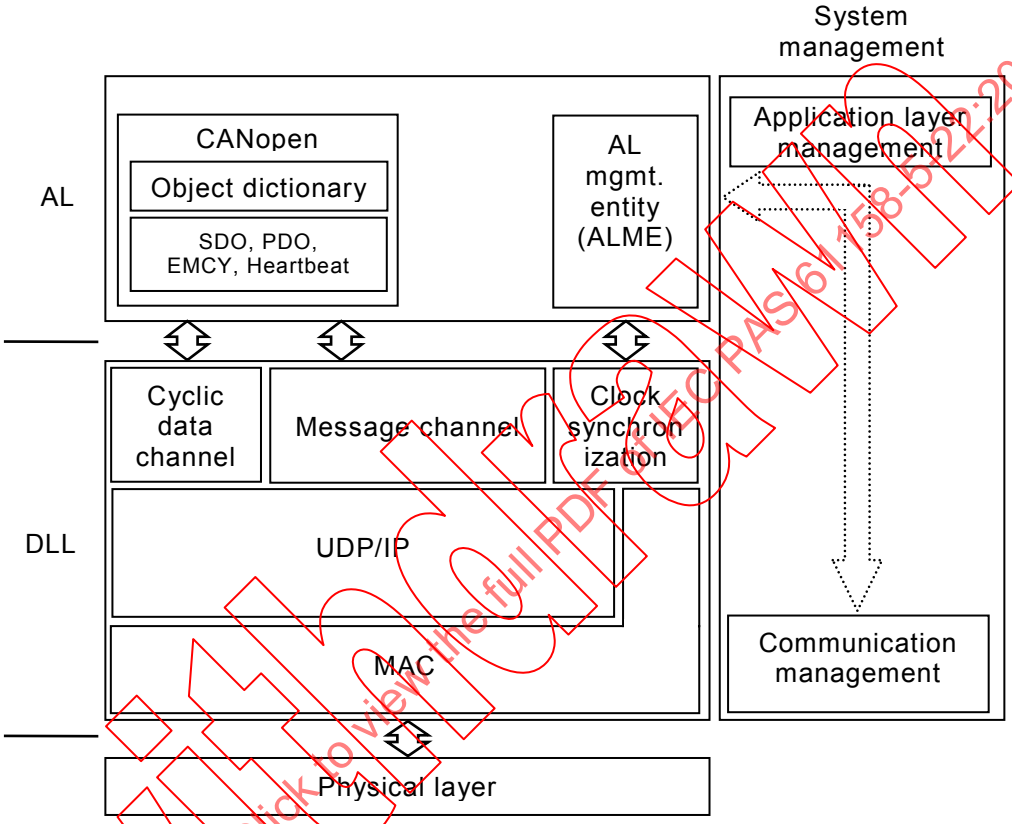

Figure 3 – RTFN device reference model

5 Data type ASE

5.1 Overview

All of IEC/TR 61158-1:2007, 10.1, is incorporated by reference.

5.2 Formal definition of data type objects

All of IEC/TR 61158-1:2007, 10.2, 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

There are no Bitstring types defined for Type SNpTYPE.

5.3.1.3 Currency types

There are no Currency types defined for Type SNpTYPE.

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 SNpTYPE.

5.3.1.6 Handle types

There are no Handle types defined for Type SNpTYPE.

5.3.1.7 Numeric types

5.3.1.7.1 float

This data type is the same as Float32.

5.3.1.7.2 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 IEC 60559 as single precision.

5.3.1.7.3 double

This data type is the same as Float64.

5.3.1.7.4 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 IEC 60559 as double precision.

5.3.1.8 Integer types

5.3.1.8.1 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.

5.3.1.8.2 SINT

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

5.3.1.8.3 char

This data type is the same as Integer8.

5.3.1.8.4 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.

5.3.1.8.5 INT

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

5.3.1.8.6 short

This data type is the same as Integer16.

5.3.1.8.7 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.

5.3.1.8.8 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.

5.3.1.8.9 DINT

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

5.3.1.8.10 long

This data type is the same as Integer32.

5.3.1.8.11 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.

5.3.1.8.12 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.

5.3.1.8.13 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.

5.3.1.8.14 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.

5.3.1.8.15 LINT

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

5.3.1.9 Unsigned types

5.3.1.9.1 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.

5.3.1.9.2 USINT

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

5.3.1.9.3 unsigned char

This data type is the same as Unsigned8.

5.3.1.9.4 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 type has a length of two octets.

5.3.1.9.5 UINT

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

5.3.1.9.6 WORD

This type is used in the same way as UINT.

5.3.1.9.7 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 type has a length of three octets.

5.3.1.9.8 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 type has a length of four octets.

5.3.1.9.9 UDINT

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

5.3.1.9.10 DWORD

This type is used in the same way as UDINT.

5.3.1.9.11 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 type has a length of five octets.

5.3.1.9.12 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 type has a length of six octets.

5.3.1.9.13 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 type has a length of seven octets.

5.3.1.9.14 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 type has a length of eight octets.

5.3.1.9.15 ULINT

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

5.3.1.9.16 Unsigned128

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	28
2	Data type Name	=	Unsigned128
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	16

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 sixteen octets.

5.3.1.9.17 Unsigned256

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	29
2	Data type Name	=	Unsigned256
3	Format	=	FIXED LENGTH
4.1	Octet Length	=	32

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 thirty-two octets.

5.3.1.10 Pointer types

There are no Pointer types defined for Type SNpTYPE.

5.3.1.11 OctetString types

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

5.3.1.12 VisibleString character types

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

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.

NOTE IEC/PAS 61158-6-22 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 Domain

CLASS:			**Data type**
ATTRIBUTES:			
1	Data type Numeric Identifier	=	15
2	Data type Name	=	Domain
3	Format	=	STRING
4.1	Octet Length	=	1 to n

Large variable amount of data, for example executable program.

6 Communication model specification

6.1 ASEs

6.1.1 CeS ASE

6.1.1.1 Overview

6.1.1.1.1 General information

CANopen is a non-proprietary open fieldbus standard specified and standardized by CiA (CAN in Automation) organization.

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

In conjunction with the CAN Bus based protocol, a uniform and standardized application layer is provided for industrial applications. This includes standardization of communication, including technical and functional features allowing networking of distributed field automating devices and standardization of application objects using device profiles.

The device profiles are one of the core elements of CANopen, specifying uniform functions and standardized parameters/objects for different application areas or for automated device groups. Based on these standardized profiles, a great degree of vendor compatibility can be

achieved due to interoperability and interchangeability of devices made by different manufacturers. All major device types used in automation engineering such as:

- digital and analogue I/O devices;
- drives;
- valves;
- programmable controls;
- encoders, etc.

are already standardized as device profiles and are reflected in the relevant CiA device profile specification.

The object dictionary contains parameters, application data and the mapping information between process data objects and application data (PDO mapping). Its entries can be accessed via service data objects (SDO), as shown in Figure 4.

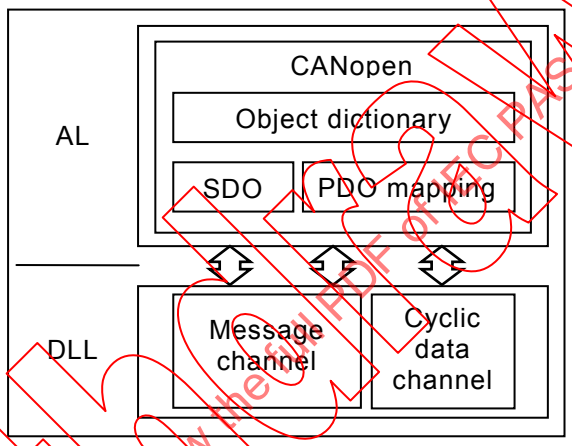

Figure 4 – Type SNpTYPE CeS device structure

6.1.1.1.2 Object dictionary structure

The object dictionary is the interface between the application and the communication sub-system. Essential as a central element, the object dictionary is a grouping of objects and specifies uniform communication and device parameters, data and functions which are stored and retrieved using objects. It is a collection of the device parameters data structures that can be accessed with the SDO Upload and SDO Download services.

The dictionary is organized in form of a table as indicated in Table 1 below. The structure corresponds to the CANopen specification 301 known from industrial automation.

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

Table 1 – Object dictionary structure

Section	Sub-section	Content
Data type	Basic data types	Definition of basic data types
—	Complex data types	Definition of complex data types
—	Manufacturer specific data types	Definition of manufacturer specific data types
—	Device profile specific basic data types	Definition of device profile specific basic data types
—	Device profile specific complex data types	Definition of device profile specific complex data types
Communication profile	—	Definition of the parameters which are used for communication configuration and dedicated communication purposes
Manufacturer defined profile	—	Definition of manufacturer specific parameters
Standardized device profile	—	Definition of the parameters defined in a standardized device profile
Standardized interface profile	—	Definition of the parameters defined in standardized interface profile
Type SNpTYPE RTFN interface profile	—	Definition of the parameters defined in Type SNpTYPE RTFN interface profile

6.1.1.1.3 Data type areas

The Data type area consists of the following parts:

Basic 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 basic data types

Definition of device profile specific simple data types

Device profile specific complex data types

Definition of device profile specific structured data types

6.1.1.1.4 Communication area

6.1.1.1.4.1 Device type

The device type object consists of the following parameter:

Parameter

Device profile number

This parameter specifies the device profile that is used of the device.

Additional information

This parameter specifies additional information defined by the device profile.

6.1.1.1.4.2 Error register

The error register object consists of the following parameter:

Parameter

Generic Error

This parameter indicates the presence of a generic error.

Current

This parameter indicates the presence of a current error.

Voltage

This parameter indicates the presence of a voltage error.

Temperature

This parameter indicates the presence of a temperature error.

Communication

This parameter indicates the presence of a communication error.

Device profile specific

This parameter indicates the presence of a device profile specific error.

Manufacturer specific

This parameter indicates the presence of a manufacturer specific error.

6.1.1.1.4.3 Manufacturer status register

The manufacturer status register object consists of the following parameter:

Parameter

Status register

This parameter specifies the status register of the device.

6.1.1.1.4.4 Event log

The event log object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries for this object.

Emergency error code

This parameter specifies the emergency error code for the occurrence of an event.

Manufacturer specific error field

This parameter specifies the manufacturer specific error field for the occurrence of an event.

Time stamp

This parameter specifies the occurrence in time of an event.

Length

This parameter specifies length of the extended manufacturer information.

Extended manufacturer information

This parameter specifies the extended manufacturer specific information.

6.1.1.1.4.5 Manufacturer device name

The manufacturer device name object consists of the following parameter:

Parameter

Device name

This parameter specifies the device name of the device.

6.1.1.1.4.6 Manufacturer hardware version

The manufacturer hardware version object consists of the following parameter:

Parameter

Hardware version

This parameter specifies the manufacturer hardware version of the device.

6.1.1.1.4.7 Manufacturer software version

The manufacturer software version object consists of the following parameter:

Parameter

Software version

This parameter specifies the manufacturer software version of the device.

6.1.1.1.4.8 Communication layer configuration

The communication layer configuration object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries for this object.

Symbolic device name

This parameter specifies the symbolic device name of the device.

Device role

This parameter specifies the role of the device within the communication system.

RTFN base cycle time

This parameter specifies the RTFN base cycle time of the device.

IP Address

This parameter specifies the IP address of the device.

Subnet mask

This parameter specifies the subnet mask of this device.

Default gateway

This parameter specifies the default gateway of the device.

DHCP enabled

This parameter specifies the usage of DHCP for the device.

Current IP configuration

This parameter specifies the activation for IP configuration.

6.1.1.1.4.9 Time sync IRQ configuration

The time sync IRQ configuration object consists of the following parameter:

Parameter

Sync ID number

This parameter specifies the device internal number of the time sync ID.

Number of entries

This parameter specifies the number of time sync IRQ configuration entries.

Time sync ID

This parameter specifies a network unique identifier for a group of synchronous devices.

Cycle time

This parameter specifies a value which triggers a specific timing signal (IRQ).

Time offset

This parameter specifies a value and describes the time offset of a device to the SYNC master.

Is master

This parameter specifies the role of the device for synchronization mechanism.

IPv4 address sync master

This parameter specifies the IP address for IPv4 of the sync master device.

IPv6 address sync master

This parameter specifies the IP address for IPv6 of the sync master device.

6.1.1.1.4.10 Time sync IRQ state

The time sync IRQ state object consists of the following parameter:

Parameter

Sync ID number

This parameter specifies the device internal number of the time sync ID.

Number of entries

This parameter specifies the number of time sync IRQ state entries.

Time sync IRQ state

This parameter specifies the state of synchronization for a particular time sync ID of the device.

6.1.1.1.4.11 Store parameters

The store parameters object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

All parameters

This parameter specifies a command to store all device parameters.

Communication profile

This parameter specifies a command to store all communication profile parameters.

Application parameters

This parameter specifies a command to store all application parameters.

Manufacturer specific parameters

This parameter specifies a command to store manufacturer specific parameters or parameter groups.

6.1.1.1.4.12 Restore default parameters

The restore default parameters object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

All parameters

This parameter specifies a command to restore all default device parameters.

Communication profile

This parameter specifies a command to restore all default communication profile parameters.

Application parameters

This parameter specifies a command to restore all default application parameters.

Manufacturer specific parameters

This parameter specifies a command to restore manufacturer specific default parameters or default parameters of parameter groups.

6.1.1.1.4.13 Diagnostic information

The diagnostic information object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

Application layer state

This parameter specifies information about the application layer state.

Application state

This parameter specifies information about the application state.

CL state RTFL

This parameter specifies information about the communication layer state for RTFL.

CL state RTFN

This parameter specifies information about the communication layer state for RTFN.

Number of delayed RTFL frames

This parameter specifies information about the number of delayed RTFL frames.

Number of corrupt frames

This parameter specifies information about the number of corrupt frames.

Number of received frames since startup

This parameter specifies information about the number of received frames since startup.

Number of MSC buffer overflows

This parameter specifies information about the number of MSC buffer overflows.

Number of received MSC messages since startup

This parameter specifies information about the number of received MSC messages since startup.

Cable attenuation port 1

This parameter specifies information about the cable attenuation for port 1.

Cable attenuation port 2

This parameter specifies information about the cable attenuation for port 2.

Cable length port 1

This parameter specifies information about the cable length for port 1 cabling.

Cable length port 2

This parameter specifies information about the cable length for port 2 cabling.

Distance to fault port 1

This parameter specifies information about the distance to a cabling fault for port 1.

Distance to fault port 2

This parameter specifies information about the distance to a cabling fault for port 2.

6.1.1.1.4.14 Diagnostic thresholds

The diagnostic thresholds object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

Expected RTFL round trip time

This parameter specifies information about the expected RTFL round trip time.

Delayed RTFL rate threshold

This parameter specifies information about the delayed RTFL rate threshold.

Corrupt frame rate threshold

This parameter specifies information about the corrupt frame rate threshold.

MSC buffer overflows rate threshold

This parameter specifies information about the MSC buffer overflows rate threshold.

Cable attenuation port 1 threshold

This parameter specifies information about the cable attenuation port 1 threshold.

Cable attenuation port 2 threshold

This parameter specifies information about the cable attenuation port 2 threshold.

6.1.1.1.4.15 IP address EMCY

The IP address EMCY object consists of the following parameter:

Parameter

IP address

This parameter specifies the IP address of the destination device for EMCY messages.

6.1.1.1.4.16 Inhibit time EMCY

The inhibit time EMCY object consists of the following parameter:

Parameter

Inhibit time

This parameter specifies a message inhibit time for emergency messages.

6.1.1.1.4.17 Consumer heartbeat list

The consumer heartbeat list object consists of the following parameter:

Parameter

Heartbeat producer number

This parameter specifies the device internal number of the heartbeat producer to be monitored.

Number of entries

This parameter specifies the number of consumer heartbeat list entries.

RTFL PID

This parameter specifies RTFL packet ID of the monitored heartbeat.

RTFN PID

This parameter specifies RTFN packet ID of the monitored heartbeat.

TT

This parameter specifies the transmission type of the monitored heartbeat.

Heartbeat time

This parameter specifies the heartbeat time as a multiple of the base cycle time.

Cycle multiplier

This parameter specifies the expected transmission cycle as a multiplier for the cycle time of the communication system in case of inter-cell communication.

Cycle offset

This parameter specifies an offset for the expected transmission cycle in relation to a communication system cycle in case of inter-cell communication.

Device address

This parameter specifies the device address of the heartbeat producer.

IPv4 address

This parameter specifies the IP address (IPv4) of the heartbeat producer.

IPv6 address

This parameter specifies the IP address (IPv6) of the heartbeat producer.

6.1.1.1.4.18 Producer heartbeat parameter

The producer heartbeat parameter object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of valid entries.

RTFL PID

This parameter specifies the RTFL packet ID.

RTFN PID

This parameter specifies the RTFN packet ID.

Transmission type

This parameter specifies the transmission type of the heartbeat.

Time sync ID

This parameter specifies a time sync ID assigned to the heartbeat in case of synchronized heartbeat processing.

Cycle multiplier

This parameter specifies the transmission cycle as a multiplier of the cycle time of the communication system.

Cycle offset

This parameter specifies the transmission cycle of the heartbeat.

Device address

This parameter specifies the device address of the recipient of the heartbeat for message channel based communication.

IPv4 address

This parameter specifies the IPv4 address of the recipient of the heartbeat for message channel based inter-cell communication.

IPv6 address

This parameter specifies the IPv6 address of the recipient of the heartbeat for message channel based inter-cell communication.

6.1.1.1.4.19 Identity object

The identity object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

Vendor ID

This parameter specifies the vendor ID of the device.

Product code

This parameter specifies the product code of the device.

Revision number

This parameter specifies the revision number of the device.

Serial number

This parameter specifies the serial number of the device.

Type SNpTYPE protocol version

This parameter specifies the Type SNpTYPE protocol version of the device.

6.1.1.1.4.20 SDO protocol timeout

The SDO protocol timeout object consists of the following parameter:

Parameter

Timeout time

This parameter specifies the SDO protocol timeout time for the device.

6.1.1.1.4.21 Enable client SDO parameter

The enable client SDO parameter object consists of the following parameter:

Parameter

Enable client SDO

This parameter specifies a command to enable client SDO for the device.

6.1.1.1.4.22 Enable EMCY

The enable EMCY object consists of the following parameter:

Parameter

Enable EMCY

This parameter specifies a command to enable EMCY for the device.

6.1.1.1.4.23 PDO timeout tolerance

The PDO timeout tolerance object consists of the following parameter:

Parameter

Timeout tolerance

This parameter specifies the PDO timeout tolerance for the device.

6.1.1.1.4.24 Store EDS

The store EDS object consists of the following parameter:

Parameter

Store EDS

This parameter specifies the EDS file of the device.

6.1.1.1.4.25 Storage format

The storage format object consists of the following parameter:

Parameter

Format

This parameter specifies the format of the file of the device.

6.1.1.1.4.26 OS command

The OS command object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

Command

This parameter specifies an OS command.

Status

This parameter specifies a status of the command execution.

Reply

This parameter specifies a reply to the command.

6.1.1.1.4.27 OS command mode

The OS command mode object consists of the following parameter:

Parameter

Command mode

This parameter specifies the OS command execution.

6.1.1.1.4.28 OS debugger interface

The OS debugger interface object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

Command

This parameter specifies an OS command.

Status

This parameter specifies a status for the command execution.

Reply

This parameter specifies a reply to the command executed.

6.1.1.1.4.29 OS prompt

The OS prompt object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

StdIn

This parameter specifies OS standard input channel.

StdOut

This parameter specifies OS standard output channel.

StdErr

This parameter specifies OS standard error channel.

6.1.1.1.4.30 Module list

The module list object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of connected modules.

Module

This parameter specifies the module type and characteristic of connected modules of a modular device.

6.1.1.1.4.31 Emergency subscriber

The module list object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of subscribed emergency messages.

Device address

This parameter specifies the device address of the emergency message producer.

IP address

This parameter specifies the IP address of the emergency message producer.

Additional information

This parameter specifies additional information for a producer.

6.1.1.1.4.32 Client SDO parameter

The client SDO parameter object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of entries.

Server address

This parameter specifies the server address.

6.1.1.1.4.33 Receive PDO communication parameter

The receive PDO communication parameter object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of valid entries.

RTFL PID

This parameter specifies the RTFL packet ID.

RTFN PID

This parameter specifies the RTFN packet ID.

Transmission type

This parameter specifies the transmission type of the PDO.

Time sync ID

This parameter specifies a time sync ID assigned to the RxPDO in case of synchronized RxPDO processing.

Timeout

This parameter specifies a timeout time for the PDO.

Cycle multiplier

This parameter specifies the expected transmission cycle as a multiplier for the cycle time of the communication system in case of inter-cell communication.

Cycle offset

This parameter specifies an offset for the expected transmission cycle in relation to a communication system cycle in case of inter-cell communication.

Device address

This parameter specifies the device address of the producer of the PDO for message channel based communication.

IPv4 address

This parameter specifies the IPv4 address of the producer of the PDO for message channel based inter-cell communication.

IPv6 address

This parameter specifies the IPv6 address of the producer of the PDO for message channel based inter-cell communication.

6.1.1.1.4.34 Receive PDO mapping parameter

The receive PDO mapping parameter object consists of 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.

6.1.1.1.4.35 Transmit PDO communication parameter

The transmit PDO communication parameter object consists of the following parameter:

Parameter

Number of entries

This parameter specifies the number of valid entries.

RTFL PID

This parameter specifies the RTFL packet ID.

RTFN PID

This parameter specifies the RTFN packet ID.

Transmission type

This parameter specifies the transmission type of the PDO.

Time sync ID

This parameter specifies a time sync ID assigned to the TxPDO in case of synchronized TxPDO processing.

Cycle multiplier

This parameter specifies the transmission cycle as a multiplier of the cycle time of the communication system.

Cycle offset

This parameter specifies the transmission cycle of the PDO.

Device address

This parameter specifies the device address of the consumer of the PDO for message channel based communication.

IPv4 address

This parameter specifies the IPv4 address of the consumer of the PDO for message channel based inter-cell communication.

IPv6 address

This parameter specifies the IPv6 address of the consumer of the PDO for message channel based inter-cell communication.

6.1.1.1.4.36 Transmit PDO mapping parameter

The transmit PDO mapping parameter object consists of 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.

6.1.1.1.5 SDO interactions

6.1.1.1.5.1 SDO services

Object dictionary access is handled by service data object (SDO) services. Object entries in remote object dictionaries can be read or written (upload and download) using the appropriate SDO services. SDOs are data formats (structures) of any size which are addressable. The addressing follows the CANopen object dictionary addressing format and consists of a 16 bit index and 8 bit sub-index, used to address the appropriate object dictionary entries. Statement of index and sub-index is also referred to as multiplexor.

In order to provide concurrent SDO service processing in a device, Type SNpTYPE introduces the client-specified JobID parameter. This JobID is assigned to each single SDO communication. The server uses this JobID in responses. One JobID is valid for the entire protocol sequence.

SDO communication is based on the client-server model and is implemented using a point-to-point link between two devices. Type SNpTYPE CeS does not require a segmented SDO transport protocol for standard SDO services for the transfer of larger data volumes. The segmenting function is provided by the Type SNpTYPE DLL which transfers large data volumes of any length across the network.

SDO communication is confirmed, receiving an SDO message is acknowledged by an appropriate SDO message. An unconfirmed abort service records any SDO communication error. SDO communication starts with an initialization phase preparing client and server for the transmission. In case of little data volume, the data is transmitted directly in the initialization phase and the communication terminates with the subsequent acknowledgement

of the server. This mechanism is called expedited transfer. If large SDOs are transferred, no data is transferred during the initialization phase. This mechanism is called normal transfer. Following the acknowledgement in the connection setup, which ensures that data can be received, the actual data transfer phase with the appropriate SDO services commences.

The primitives of the SDO services are mapped to the primitives of the message channel services.

6.1.1.1.5.2 SDO download sequence

The SDO download service is used to write data from a client to the server. The services are acknowledged by the server and return a request success status. In case of an error they return an additional error message containing the reason for error. The reason for the error is notified to the client using SDO abort service.

Figure 5 shows a successful SDO expedited download sequence between client and server and the primitives between client and server.

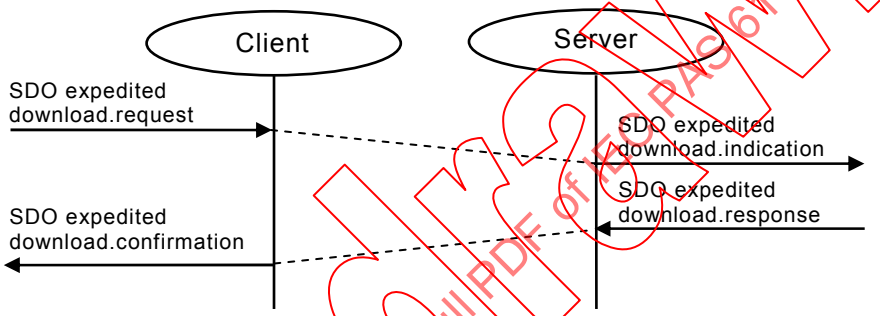

Figure 5 – Successful SDO expedited download sequence

Figure 6 shows a successful SDO normal download initialization sequence and the primitives between client and server. This transfer includes no data and has to be continued with SDO download service.

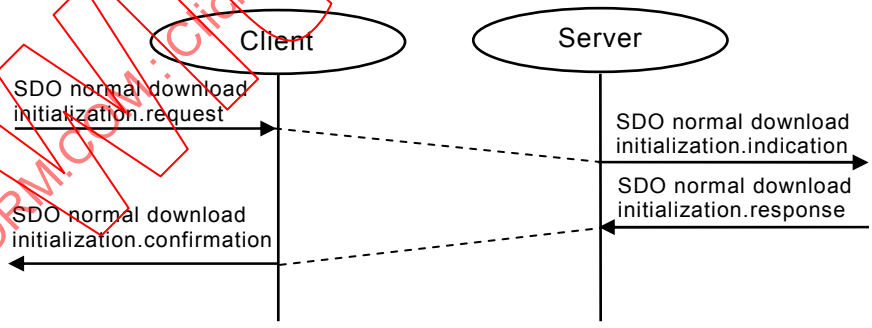

Figure 6 – Successful SDO normal download initialization sequence

Figure 7 shows a successful SDO download sequence and the primitives between client and server. This sequence follows after the successful SDO normal download initialization sequence.

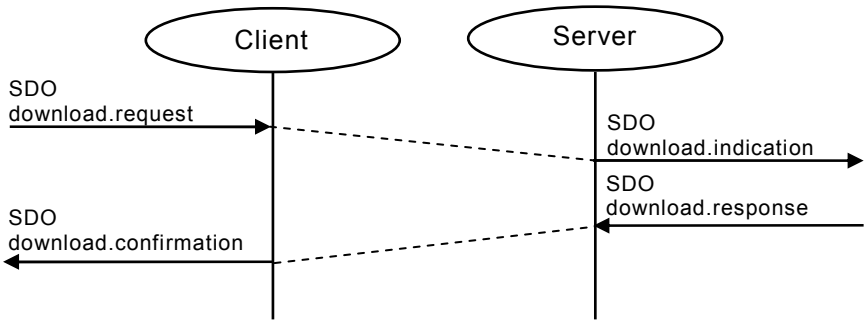

Figure 7 – Successful SDO download sequence

6.1.1.1.5.3 SDO upload sequence

The client uses the SDO upload service to read data from the server. The services are acknowledged and return a request success status as well as the requested data. In case of an error the reason is returned. This reason is notified to the client using SDO abort service.

Figure 8 shows a successful SDO expedited upload sequence and the primitives between client and server.

Figure 8 – Successful SDO expedited upload sequence

Figure 9 shows a successful SDO normal upload initialization sequence and the primitives between client and server. This transfer includes no data and has to be continued with SDO upload service.

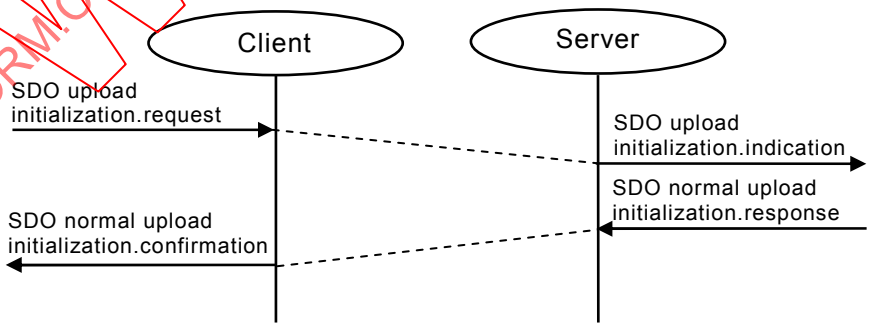

Figure 9 – Successful SDO normal upload initialization sequence

Figure 10 shows a successful SDO upload sequence and the primitives between client and server. This sequence follows a successful initialization sequence.

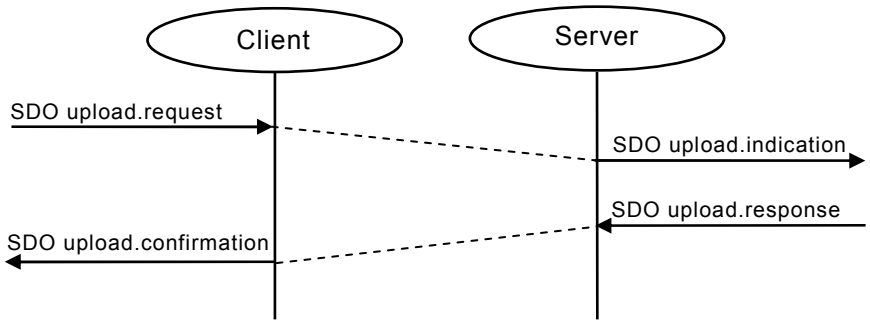

Figure 10 – Successful SDO upload sequence

6.1.1.1.5.4 SDO abort

This service aborts the SDO upload or download attempt whereby either client or server can request this service. The SDO abort service is unconfirmed and additionally contains a coded error description.

Figure 11 shows a failed SDO expedited download initialization sequence and the primitives between client and server. The server requests SDO abort service. The abort sequences for failed SDO normal download initialization or SDO upload initialization sequences are identical.

Figure 11 – Failed SDO expedited download initialization sequence

Figure 12 shows a download access error message initiated by the client to the server after a successful SDO normal download initialization sequence. The abort service sequence following an upload access is identical.

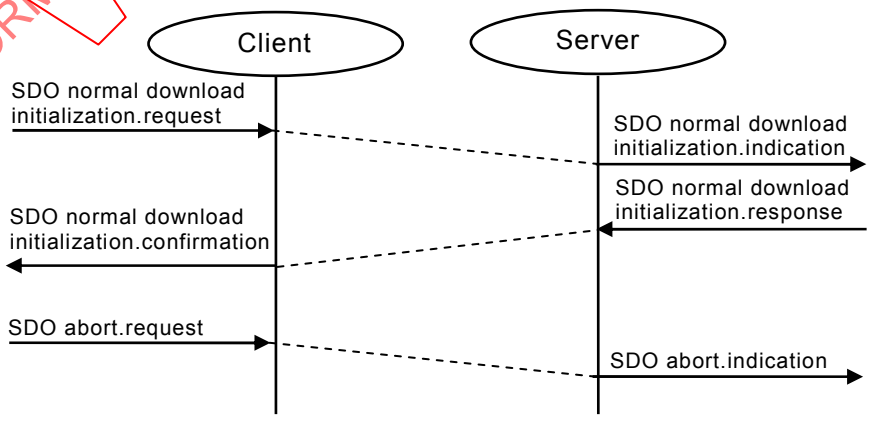

Figure 12 – Failed SDO download after initialization sequence

Figure 13 shows a failed SDO download sequence which can follow a successful SDO normal download initialization sequence. Due to an error, the server requests the SDO abort service after data reception.

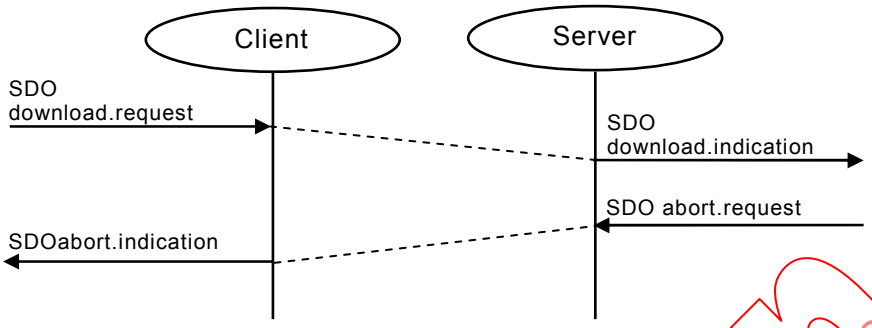

Figure 13 – Failed SDO download sequence

6.1.1.1.6 Emergency

Emergency (EMCY) messages are triggered by the occurrence of a device internal error situation. The primitives of the emergency service are mapped to the primitives of the message channel services.

Figure 14 shows the primitives between producer and consumer.

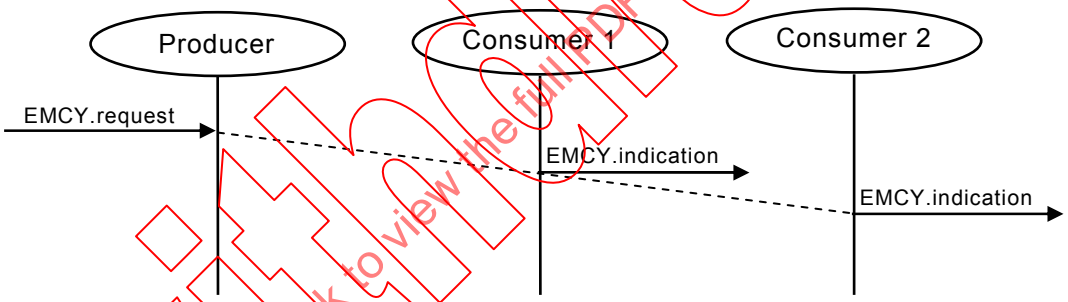

Figure 14 – Emergency sequence

6.1.1.1.7 Heartbeat

Devices can monitor each other by means of the heartbeat mechanism. Heartbeat producers generate life-signs which are received by heartbeat consumers. A heartbeat consumer can configure in the object dictionary which heartbeat producers it wants to monitor. If a heartbeat fails to arrive, a heartbeat event is generated at the heartbeat consumer.

The primitives of the heartbeat service can be mapped to CDC or to MSC primitives. This is configuration dependent. Figure 15 illustrates the functional principle of the heartbeat service.

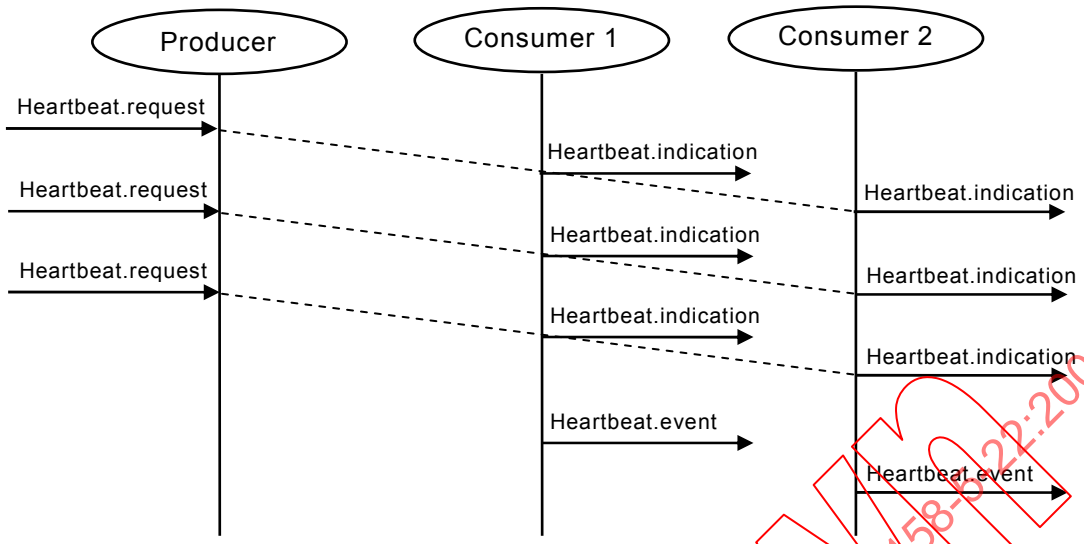

Figure 15 – Heartbeat sequence

6.1.1.1.8 Process data service

6.1.1.1.8.1 Process data interaction

The process data write service is used by a PDO producer to send the data of the mapped application objects to the PDO consumers. The process data write service indicates the PDO consumer that a valid PDO has been received.

The primitives of the process data service can be mapped to the to CDC or MSC service primitives depending on the communication parameters of a particular PDO. Figure 16 shows the process data write service sequence.

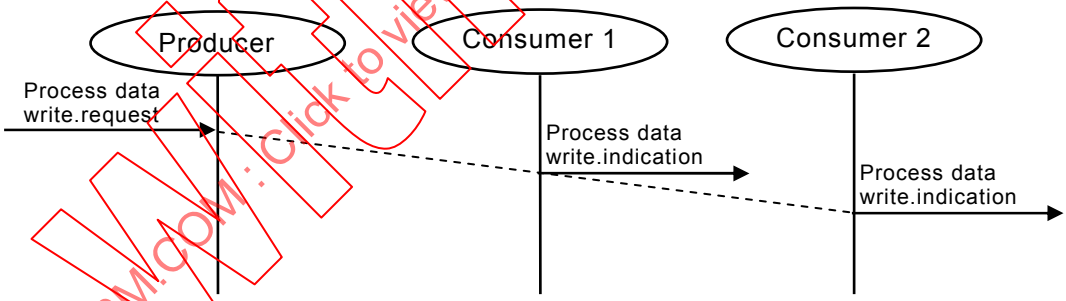

Figure 16 – Process data write sequence

6.1.1.1.8.2 Process data object communication parameters

The PDO communication parameters describe a PDO with regard to its communication characteristics. The transmission type parameter identifies the PDO transfer type. The communication channel (CDCL / CDCN / MSCL / MSCN) over which the PDO is to be transmitted or received and the communication mode, either cyclic, synchronized or event driven (acyclic) can be specified for a PDO. Figure 17 shows an example of a PDO configuration including communication and mapping parameters.

6.1.1.1.8.3 Mapping of objects to process data

The content of process data objects (PDOs) is described in the object dictionary and consists of application objects out of the object dictionary which can be mapped to PDOs. The PDO mapping objects depict lists of object references together with the length information. A valid PDO shall contain at least one application object and at most 254 application objects.

The current PDO mapping can be read by means of corresponding objects in the object dictionary. These are known as mapping objects. The first entry in a mapping object (sub-index 0) specifies the number of mapped application objects that are listed after it. The tables are located in the object dictionary at index 0x1600 to 0x17FF for the RxPDOs and at 0x1A00 to 0x1BFF for the TxPDOs. Figure 17 shows an example of a PDO configuration including communication and mapping parameters.

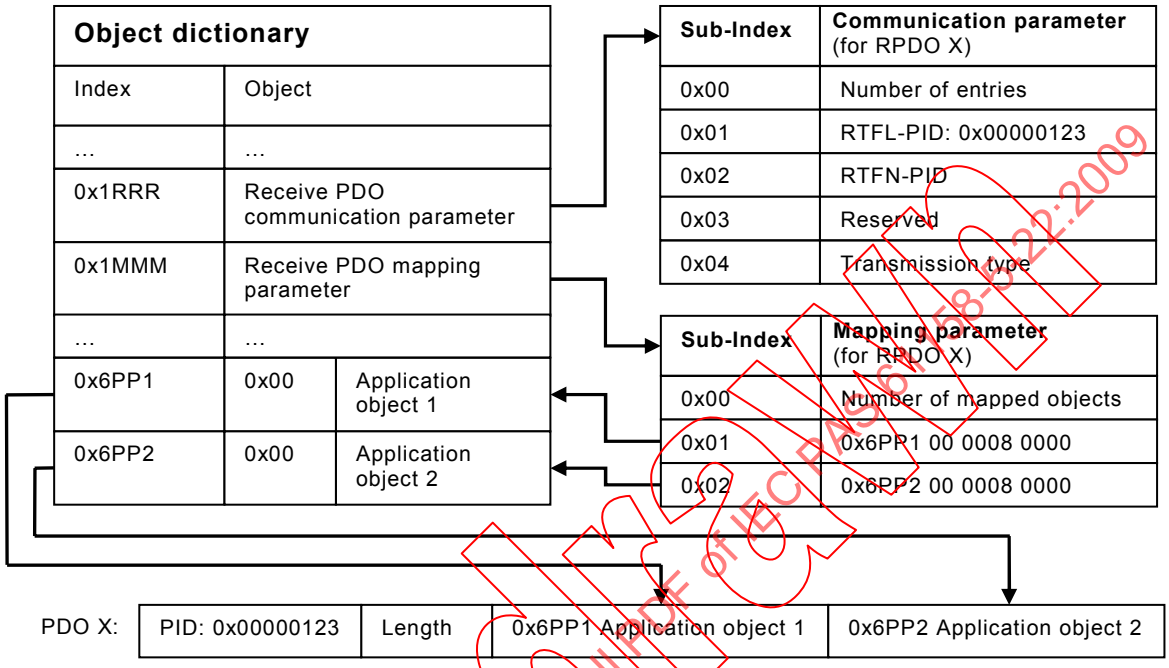

Figure 17 – PDO mapping principle

6.1.1.1.8.4 Process data objects

The general structure of a process data object is depicted in Figure 18 and corresponds to the CDC DLPDU data as specified in IEC/PAS 61158-4-22:2009, 5.7.

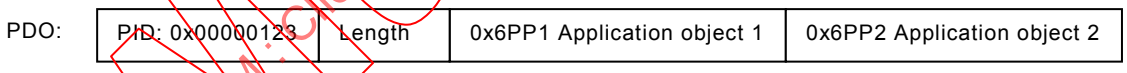

Figure 18 – Process data object

6.1.1.2 CeS class specification

6.1.1.2.1 Formal model

FAL ASE:	**CeS**
CLASS:	**CeS object dictionary**
CLASS ID:	**not used**
PARENT CLASS:	TOP

ATTRIBUTES:

1	(m)	Key Attribute:	Implicit
1	(m)	Attribute:	Entry list
1.1	(m)	Attribute:	Index
1.2	(m)	Attribute:	Object type
1.3	(o)	Attribute:	Name and description
1.4	(m)	Attribute:	List of entry
1.4.1	(m)	Attribute:	Sub-index
1.4.2	(o)	Attribute:	Description
1.4.3	(m)	Attribute:	Data type
1.4.4	(o)	Attribute:	Access attribute
1.4.5	(o)	Attribute:	PDO mapping
1.4.6	(o)	Attribute:	Value range
1.4.7	(o)	Attribute:	Default value

SERVICES:

1	(o)	OpsService:	Initiate SDO expedited download
2	(o)	OpsService:	Initiate SDO normal download
3	(o)	OpsService:	SDO download
4	(o)	OpsService:	Initiate SDO expedited upload
5	(o)	OpsService:	Initiate SDO normal upload
6	(o)	OpsService:	SDO upload
7	(c)	Constraint:	Any SDO services supported
7.1	(m)	OpsService:	SDO abort
8	(m)	OpsService:	Process data write
9	(o)	OpsService:	Emergency
10	(m)	OpsService:	Heartbeat

6.1.1.2.2 Attributes

Implicit

The attribute indicates that the object is implicitly addressed by the service.

Entry list

One object is composed of the following list elements:

Index

This attribute specifies a numerical index of the object.

Object type

This attribute specifies a characterising object type of an object.

Name and description

This attribute specifies a name uniquely describing the object.

List of entry

This attribute consists of the following attributes.

Sub-index

This attribute specifies a numerical index to identify individual elements of an object.

Description

This attribute specifies a name of an element of the object.

Data type

This attribute specifies a data type of an element of the object.

Access attribute

This attribute specifies the access right of an element of the object. Its allowed values are:

– RW (Read and write access)

– WO (Write only)

– RO (Read only)

PDO mapping

This attribute specifies if an element of the object can be mapped into a PDO.

Value range

This attribute specifies value range or data type of the entire range of an element of the object.

Default value

This attribute specifies the default value on device initialization of an element of the object.

6.1.1.2.3 Services

Initiate SDO expedited download

This service writes small data volumes (configuration dependent) to the server.

Initiate SDO normal download

This service is used for write access initialization and requests to prepare for SDO transfer.

SDO download

This service is used for data transfer to the server.

Initiate SDO expedited upload

This service reads small data volumes (configuration dependent) from the server.

Initiate SDO normal upload

This service is used for read access initialization and requests to prepare for SDO transfer.

SDO upload

This service is used for data transfer from the server.

SDO abort

Abort of service by client or server in case of an error.

Process data write

This service is used for transfer of process data.

Emergency

Service provides for diagnostic messages on occurrence of errors or unexpected conditions.

Heartbeat

This service provides for device monitoring.

6.1.1.3 CeS service specification

6.1.1.3.1 Supported services

The CeS ASE defines the services

Initiate SDO expedited download
Initiate SDO normal download
SDO download
Initiate SDO expedited upload
Initiate SDO normal upload
SDO upload
SDO abort
Process data write
Emergency
Heartbeat

6.1.1.3.2 Initiate SDO expedited download service

This service depicts the expedited transfer initialization and is only used by the client if the data volume is small enough to be transferred directly. The maximum data volume is configuration dependent. The service requires the data write location in the server object dictionary with its multiplexor consisting of index and sub-index and the data. The server shall respond to the client's request or acknowledge the arrival of the SDO. SDO transfer is completed with receipt of the positive confirmation by the client. The service return parameter returns the communication success status to the client. In case of server error, the response must use the SDO abort service. Table 2 shows the service primitives and parameter of the service.

Table 2 – Initiate SDO expedited download service

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP	M	M		
Address	M	M (=)		
JobID	M	M (=)		
Index	M	M (=)		
Sub-index	M	M (=)		
Data	M	M (=)		
Result (+)			S	S (=)
Address			M	M
JobID			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 shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Index

This parameter specifies the index in the server object dictionary which is to be downloaded.

Sub-index

This parameter specifies the sub-index in the server object dictionary which is to be downloaded.

Data

This parameter specifies the object value to be downloaded.

Result(+)

This selection type parameter indicates that the service request succeeded.

Address

This parameter specifies the address of the client for response and the address of the server for a confirmation.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Result(-)

This selection type parameter indicates that the service request failed.

6.1.1.3.3 Initiate SDO normal download

This service is used for the server object dictionary write access initialization phase and requests the server to prepare for SDO transfer. Normal indicates normal transfer initialization. This is used by the client if the volume of transfer data is large and the abort conditions are to be checked prior to the actual transfer. The service requires the data write location in the server object dictionary with its multiplexor consisting of index and sub-index, and the data. The server shall respond to the client's request. The service return parameter returns the communication success status to the client. In case of server error, the response must use the SDO abort service. Following successful initialization, this service is always succeeded by the SDO download service. Table 3 shows the service primitives and parameter of the service.

Table 3 – Initiate SDO normal download service

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP	M	M		
Address	M	M		
JobID	M	M (=)		
Index	M	M (=)		
Sub-index	M	M (=)		
Size	M	M (=)		
Result (+)			S	S (=)
Address			M	M
JobID			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 shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Index

This parameter specifies the index in the server object dictionary which is to be downloaded.

Sub-index

This parameter specifies the sub-index in the server object dictionary which is to be downloaded.

Size

This parameter specifies the data volume.

Result(+)

This selection type parameter indicates that the service request succeeded.

Address

This parameter specifies the address of the client for a response and the address of the server for confirmation.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Result(-)

This selection type parameter indicates that the service request failed.

6.1.1.3.4 SDO download

This service is the successor service of a normal-labeled initialization sequence and transfers the actual data to the server. The data write location via multiplexor is no longer required for this service as this has already occurred in the initialization phase. The server shall acknowledge the receipt of data to the client. The service return parameter returns the communication success status to the client. In case of server error, the response shall use the SDO abort service. SDO transfer is completed with the reception of the positive confirmation by the client. Table 4 shows the service primitives and parameter of the service.

Table 4 – SDO download service

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP	M	M		
Address	M	M		
JobID	M	M (=)		
Data	M	M (=)		
Result (+)			S	S (=)
Address			M	M
JobID			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 shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Data

This parameter specifies the data to be downloaded.

Result(+)

This selection type parameter indicates that the service request succeeded.

Address

This parameter specifies the address of the client for a response and the address of the server for confirmation.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Result(-)

This selection type parameter indicates that the service request failed.

6.1.1.3.5 Initiate SDO expedited upload

This service is used for the client read access initialization sequence to the server object dictionary and requests the server to prepare for SDO transfer. In this case, the server responds with expedited upload response in which the data is already transferred and which completes the upload protocol. The service requires the data read location in the server object dictionary with its multiplexor consisting of index and sub-index. The service return parameter returns the communication success status and the requested data to the client. In case of a server error, the response must use the SDO abort service. Table 5 shows the service primitives and parameter of the service.

Table 5 – Initiate SDO expedited upload service

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP	M	M		
Address	M	M		
JobID	M	M (=)		
Index	M	M (=)		
Sub-index	M	M (=)		
Result (+)			S	S (=)
Address			M	M
JobID			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 shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Index

This parameter specifies the index in the server object dictionary which is to be uploaded.

Sub-index

This parameter specifies the sub-index in the server object dictionary which is to be uploaded.

Result(+)

This selection type parameter indicates that the service request succeeded.

Address

This parameter specifies the address of the client for a response and the address of the server for confirmation.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Data

This parameter specifies the data to be transferred.

Result(-)

This selection type parameter indicates that the service request failed.

6.1.1.3.6 Initiate SDO normal upload

This service is used for the client read access initialization sequence to the server object dictionary and requests the server to prepare for SDO transfer. The server responses the transfer as normal as the data volume to be transferred does not allow expedited transfer. The service requires the data read location in the server object dictionary with its multiplexor consisting of index and sub-index. The service return parameter returns the communication success status and, optionally, the requested data volume. In case of server error, the response must use the SDO abort service. Following the successful initialization, this service is always succeeded by the SDO upload service. Table 6 shows the service primitives and parameter of the service.

Table 6 – Initiate SDO normal upload service

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP	M	M		
Address	M	M		
JobID	M	M (=)		
Index	M	M (=)		
Sub-index	M	M (=)		
Result (+)			S	S (=)
Address			M	M
JobID			M	M (=)
Size			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 shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Index

This parameter specifies the index in the server object dictionary which is to be uploaded.

Sub-index

This parameter specifies the sub-index in the server object dictionary which is to be uploaded.

Result(+)

This selection type parameter indicates that the service request succeeded.

Address

This parameter specifies the address of the client for a response and the address of the server for confirmation.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Size

This parameter specifies the data volume.

Result(-)

This selection type parameter indicates that the service request failed.

6.1.1.3.7 SDO upload

This service is the successor service of a normal-labeled initialization sequence and requests the actual data from the server. The data read location via multiplexor is not required for this service as this has already occurred in the initialization phase. The service return parameter returns the communication success status and the requested data to the client. In case of server error, the response must use the SDO abort service. SDO transfer is completed with the reception of the positive confirmation by the client. Table 7 shows the service primitives and parameter of the service.

Table 7 – SDO upload service

Parameter name	Req	Ind	Rsp	Cnf
Argument				
AREP	M	M		
Address	M	M		
JobID	M	M (=)		
Result (+)			S	S (=)
Address			M	M
JobID			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 shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Result(+)

This selection type parameter indicates that the service request succeeded.

Address

This parameter specifies the address of the client for a response and the address of the server for confirmation.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Data

This parameter specifies the data to be transferred.

Result(-)

This selection type parameter indicates that the service request failed.

6.1.1.3.8 SDO abort

This service aborts the SDO read or write attempt whereby either client or server can execute this service. The abort service is unconfirmed and additionally contains a coded error description. Table 8 shows the service primitives and parameter of the service.

Table 8 – SDO abort service

Parameter name	Req	Ind
Argument		
AREP	M	M
Address	M	M
JobID	M	M (=)
Abort code	M	M (=)

Argument

The argument shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Address

This parameter specifies the address of the server for a request and the address of the client for indication in the case of a service request by a client or the address of the client for a request and the address of the server for indication in the case of a service request by a server.

JobID

This parameter specifies an ID selected by the client for a dedicated job to allow concurrent processing of several jobs.

Abort code

This parameter specifies an error code which characterizes the error.

6.1.1.3.9 Process data write

Process data communication in CeS is handled by process data objects (PDO). In CeS, PDOs are logic data containers transferred over the network or communication sub-system allowing process data communication between individual Type SNpTYPE devices. PDOs encapsulate application objects from the object dictionary. The DL-services which are used for transmission of a certain PDO shall be specified in the PDO transmission type parameters for each PDO.

Two types of PDOs are differentiated:

- Transmit PDOs (TxPDO); and
- Receive PDOs (RxPDO).

In the context of Type SNpTYPE networks, devices transmitting PDOs are PDO producers while devices receiving PDOs are PDO consumer. PDOs are described by PDO communication parameters and mapping parameters. The paired communication parameters and mapping parameters are mandatory for any supported PDO. The PDO communication parameters define the communication behavior with regard to the DLL. The PDO mapping parameters define the content of a PDO and must include at least one application object. The PDO length depends on the application and is defined in the appropriate mapping parameter. PDO configuration can be transmitted using SDO services.

Table 9 shows the service primitives and parameter of the service.

Table 9 – Process data write service

Parameter name	Req	Ind
Argument		
AREP	M	M
PDO number	M	M

Argument

The argument shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

PDO number

This parameter specifies a number which refers to a TxPDO for a request and to an RxPDO for an indication defined within the object dictionary.

6.1.1.3.10 Emergency service (EMCY)

The emergency service (EMCY) provides diagnostic messages for devices on occurrence of an internal error in a device. The diagnostic status is transferred using an emergency message. Only one emergency message is transmitted for each error event. If an error is no longer present, an error reset or no error coded emergency message is sent. The occurrence of errors is stored using the event log object described in 6.1.1.1.4.4. Table 10 shows the service primitives and parameter of the emergency service.

Table 10 – Emergency service (EMCY)

Parameter name	Req	Ind
Argument		
AREP	M	M
Destination device address	M	M (=)
Destination device IP address	O	O (=)
Emergency error code	M	M (=)
Error register	M	M (=)
Manufacturer specific error code	M	M (=)
Time stamp	O	O (=)
Length	O	O (=)
Extended manufacturer information	O	O (=)

Argument

The argument shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Destination device address

This parameter specifies the device address of the destination device.

Destination device IP address

This parameter specifies the device IP address of the destination device.

Emergency error code

This parameter shall contain a standardized error code.

Error register

This parameter specifies the emergency error register.

Manufacturer specific error code

This parameter shall contain a manufacturer specific error code.

Time stamp

This optional parameter specifies the time passed since a configurable relative or absolute time.

Length

This parameter shall contain the length of the extended manufacturer information field in octets.

Extended manufacturer information

This parameter shall contain a more detailed description of the error or any additional information on the error.

6.1.1.3.11 Heartbeat service

The heartbeat service allows devices to monitor each other by means of heartbeat messages. A heartbeat producer generates a life-sign which is received by the heartbeat consumers. A heartbeat consumer can configure in the object dictionary which heartbeat producers it wants to monitor. If a heartbeat fails to arrive, a heartbeat event is generated at the heartbeat consumer. Table 11 shows the service primitives and parameter of the service.

Table 11 – Heartbeat service

Parameter name	Req	Ind
Argument		
AREP	M	M
Status	M	M (=)

Argument

The argument shall convey the service specific parameters of the service request.

AREP

This parameter is the local identifier for the desired AR.

Status

This parameter specifies the heartbeat producer status.

6.1.2 Standard Ethernet frame (SEF) communication ASE

6.1.2.1 Overview

6.1.2.1.1 General information

This ASE is an optional ASE. Communication of devices with engineering tools and the possibility of integrated web servers in some devices, require TCP/IP communication in addition to Type SNpTYPE RTFL communication on the same interface. For this purposes it is possible to transfer all types of Ethernet frames over Type SNpTYPE.

Type SNpTYPE utilizes a variant of Standard Ethernet communication, 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 communication cycle time, since the fragment size can be optimized according to the available bandwidth (Ethernet fragmentation instead of IP fragmentation). Type SNpTYPE defines a standard channel, which in principle enables any device to participate in the normal Ethernet traffic.

The Ethernet frame segments are transferred to the destination OD using MSC. The receiving OD reassembles the individual segments back into the standard Ethernet frame.

6.1.2.1.2 Address Resolution

Sending of tunnelled SEFs over MSC requires appropriate addressing. An OD shall be capable of matching a MAC address in the SEF with the relevant device address. Each OD which is using the MSC for SEF communication needs to setup its own table containing matches of MSC addresses and corresponding MAC addresses.

6.1.2.1.3 SEF interaction

The SEF communication service is used to send data from a client to the server. The services are unconfirmed.

Figure 19 shows a successful SEF transfer sequence between client and server and the primitives between client and server.

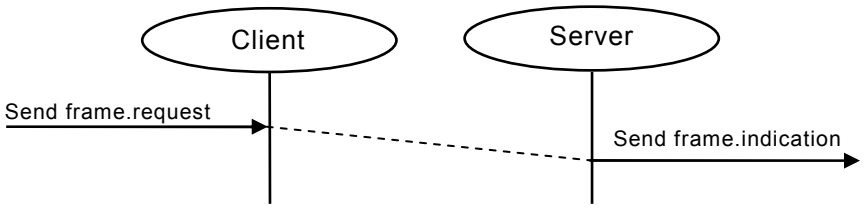

Figure 19 – SEF service sequence

6.1.2.2 SEF class specification

6.1.2.2.1 Formal model

FAL ASE:			**SEF**
CLASS:			**SEF**
CLASS ID:			**not used**
PARENT CLASS:			TOP
ATTRIBUTES:			
1	(m)	Key Attribute:	Implicit
3	(m)	Attribute:	MAC address
4	(o)	Attribute:	Address mapping list
4.1	(o)	Attribute:	MAC address
4.2	(o)	Attribute:	Device address
SERVICES:			
1	(m)	OpsService:	Send frame

6.1.2.2.2 Attributes

Implicit

The attribute indicates that the object is implicitly addressed by the service.

MAC address

This attribute specifies the MAC address of the device.

Address mapping list

The address mapping list is composed of the following elements:

MAC address

This attribute specifies the MAC address of a participating device.

Device address

This attribute specifies the device address to the corresponding MAC address of a participating device.

6.1.2.2.3 Services

Send frame

This service is used to send a standard Ethernet frame.

6.1.2.3 SEF service specification

6.1.2.3.1 Supported services

The SEF ASE defines the services

Send frame