BS EN 61375-3-3:2012

BSI Standards Publication

Electronic railway equipment — Train communication network (TCN)

Part 3-3: CANopen Consist Network (CCN)

bsi.

...making excellence a habit.™

National foreword

This British Standard is the UK implementation of EN 61375-3-3:2012. It is identical to IEC 61375-3-3:2012.

The UK participation in its preparation was entrusted by Technical Committee GEL/9, Railway Electrotechnical Applications, to Panel GEL/9/-/4, Railway applications - Train communication network and multimedia systems.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Published by BSI Standards Limited 2012

ISBN 978 0 580 68231 5

ICS 45.060

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 August 2012.

Amendments issued since publication

Amd. No.	Date	Text affected

EUROPEAN STANDARD

NORME EUROPÉENNE

EUROPÄISCHE NORM

EN 61375-3-3

August 2012

ICS 45.060

English version

Electronic railway equipment - Train communication network (TCN) - Part 3-3: CANopen Consist Network (CCN)

(IEC 61375-3-3:2012)

Matériel électronique ferroviaire - Réseau embarqué de train (TCN) - Partie 3-3: Réseau de rame CANopen (CCN)
(CEI 61375-3-3:2012)

Elektronische Betriebsmittel für Bahnen - Zug-Kommunikations-Netzwerk (TCN) - Teil 3-3: CCN-CANopen Consist Network Bus
(IEC 61375-3-3:2012)

CENELEC

European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

Management Centre: Avenue Marnix 17, B - 1000 Brussels

Foreword

The text of document 9/1646/FDIS, future edition 1 of IEC 61375-3-3, prepared by IEC/TC 9 "Electrical equipment and systems for railways" was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN 61375-3-3:2012.

The following dates are fixed:

• latest date by which the document has to be implemented at national level by publication of an identical national standard or by endorsement	(dop)	2013-04-26
• latest date by which the national standards conflicting with the document have to be withdrawn	(dow)	2015-07-26

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CENELEC [and/or CEN] shall not be held responsible for identifying any or all such patent rights.

This document has been prepared under a mandate given to CENELEC by the European Commission and the European Free Trade Association, and supports essential requirements of EU Directive(s).

For the relationship with EU Directive(s) see informative Annex ZZ, which is an integral part of this document.

Endorsement notice

The text of the International Standard IEC 61375-3-3:2012 was approved by CENELEC as a European Standard without any modification.

Annex ZA
(normative)

Normative references to international publications with their corresponding European publications

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

NOTE When an international publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

Publication	Year	Title	EN/HD	Year
-	-	Industrial communications subsystem based on ISO 11898 (CAN) for controller-device interfaces - Part 4: CANopen	EN 50325-4	2002
IEC 61131	Series	Programmable controllers	EN 61131	Series
IEC 61375-1	-	Electronic railway equipment - Train communication network (TCN) - Part 1: General architecture	EN 61375-1	-
IEC 61375-2-1	-	Electronic railway equipment - Train communication network (TCN) - Part 2-1: Wire Train Bus (WTB)	EN 61375-2-1	-
IEC 61375-2-2	-	Electronic railway equipment - Train communication network (TCN) - Part 2-2: Wire Train Bus conformance testing	EN 61375-2-2	-
ISO/IEC 646	1991	Information technology - ISO 7-bit coded character set for information interchange	-	-
ISO/IEC 9899	1999	Programming languages - C	-	-
ISO 11898-1	2003	Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signalling	-	-
ISO 11898-2	2003	Road vehicles - Controller area network (CAN) - Part 2: High-speed medium access unit	-	-

Annex ZZ
(informative)

Coverage of Essential Requirements of EU Directives

This European Standard has been prepared under a mandate given to CENELEC by the European Commission and the European Free Trade Association and within its scope the standard covers all relevant essential requirements as given in Annex III of the EU Directive 2008/57/EC.

Compliance with this standard provides one means of conformity with the specified essential requirements of the Directive concerned.

WARNING: Other requirements and other EU Directives may be applicable to the products falling within the scope of this standard.

CONTENTS

INTRODUCTION

TCN is an International Standard with the aim of defining interfaces so as to achieve plug-in compatibility:

a) between equipment located in different vehicles or consists, and

b) between equipment and devices located within the same vehicle or consist.

One of the key success factors for the deployment of any technology is standardization and ensuring interoperability among various implementations. To facilitate interoperability a conformance test should be implemented.

In this part of IEC 61375, the TCN deals with:

the consist network based on CANopen.

In addition gateway devices between the Train Backbone and the CANopen-based consist network are considered.

This standard is structured into 13 clauses.

ELECTRONIC RAILWAY EQUIPMENT – TRAIN COMMUNICATION NETWORK (TCN) –

Part 3-3: CANopen Consist Network (CCN)

1 Scope

This part of IEC 61375 specifies the data communication bus inside consists that are based on CANopen. CANopen was developed for use in, but is not limited to, industrial automation applications. These applications may include devices such as input/output modules, motion controllers, human machine interfaces, sensors, closed-loop controllers, encoders, hydraulic valves or programmable controllers.

In the application field of rail vehicles CANopen networks are utilized to network subsystems in consists such as e.g. brake control system, diesel engine control system and interior or exterior lighting control system. In addition CANopen is utilized as consist network to enable the data exchange between the different subsystems within one single rail vehicle or a group of rail vehicles sharing the same Consist Network.

This part of IEC 61375 applies to all equipment and devices operated on a CANopen-based consist network within TCN architecture as described in IEC 61375-1.

The applicability of this standard to a TCN implementation allows for individual conformance checking of the implementation itself and is a pre-requisite for further interoperability checking between different TCN implementations. In any case, proof of compatibility between Train Backbone and the Consist Network will have to be brought by the supplier.

This part of IEC 61375 applies to the architecture of communication systems in Open trains. In addition it may be applicable to closed trains and multiple unit trains when so agreed between purchaser and supplier.

2 Normative references

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

IEC 61131 (all parts): *Programmable controllers*

IEC 61375-1, ed3, *Electronic railway equipment – Train Communication Network (TCN) – Part 1: General Architecture*

IEC 61375-2-1, *Electronic railway equipment – Train Communication Network (TCN) – Part 2-1: Wire Train Bus (WTB)*

IEC 61375-2-2, *Electronic railway equipment – Train Communication Network (TCN) – Part 2-2: WTB – Wire Train Bus conformance testing*

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

ISO/IEC 9899:1999, *Programming languages – C*

ISO 11898-1:2003, *Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signaling*

ISO 11898-2:2003, *Road vehicles – Controller area network (CAN) – Part 2: High-speed medium access unit*

EN 50325-4:2002, *Industrial communication subsystems based on ISO 11898 (CAN) for controller- device interfaces – Part 4: CANopen*

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 11898-1, ISO 11898-2, IEC 61375-1, IEC 61375-2-1 and EN 50325-4, as well as the following apply.

3.1.1
ASCII
7-bit coded character set according to ISO/IEC 646

3.1.2
CANopen device
End Device connected to a CANopen-based consist network

3.1.3
Field device
networked independent physical entity of an automation system capable of performing specified functions in a particular context and delimited by its interfaces

3.1.4
logical consist network
the way the data passes through the consist network without considering the physical interconnection of the End Devices

3.1.5
logical device
representation of a field device in terms of its objects and behavior according to a field device model that describes the device's data and behavior as viewed through a network

3.1.6
Layer Setting Services
services for adjusting bit rate and Node-ID via the communication interface of the CANopen device

3.1.7
Node-ID
a) network-wide unique identifier for each CANopen device

b) 7-bit coded Device Address in CANopen-based consist networks

3.1.8
Object
entity with a well-defined boundary and identity that encapsulates state and behavior

3.1.9
virtual device
entity of software capable of accomplishing a functional element of a field device

3.2 Abbreviations

For the purposes of this document, abbreviations given in ISO 11898-1, ISO 11898-2, IEC 61375-1, IEC 61375-2-1 and EN 50325-4, as well as the following apply.

ASCII American Standard Code for Information Interchange
BNF Backus Naur Form
CAN Controller Area Network
CPU Central Processing Unit
CR Carriage Return
CRLF Carriage Return and Line Feed
DCF Device configuration file
FSA Finite state automaton
ID Identifier
LF Line Feed
LSS Layer Setting Services
NMT Network management
OSI Open system interconnect
PAS PDO access service
PDO Process data object
SAS Service data object access service
SRD SDO requesting device

3.3 Conventions

The conventions given in IEC 61375-1 shall apply.

4 Architecture

4.1 Content

This clause provides the definition of the architecture for Consist Networks based on CANopen.

4.2 Logical CANopen-based consist network

The overall Logical CANopen consist network, connecting several Virtual Devices, is illustrated in Figure 1. The CANopen-based Consist Network interconnects several Virtual Devices and sub-systems such as e.g. Train operating system (TOS), Monitoring and safety system (MSS), Auxiliary operating system (AUX), Power (drive) system (PDS), Running gear system (RGS), Brake control system (BCS), Ancillary operating system (ANC), Vehicle linkage system (VLS), Exterior lighting system (ELS), Interior lighting system (ILS), Door control system (DCS), HVAC system (HS), Passenger information system (PIS), Diagnostic system (DS), or Train-to-ground communication system (TCS).

The Logical CANopen-based consist netwok is connected to the Train Backbone via a Gateway. The Gateway manages the information exchange as well as the process data marshalling between the Train Backbone and the Consist Network.

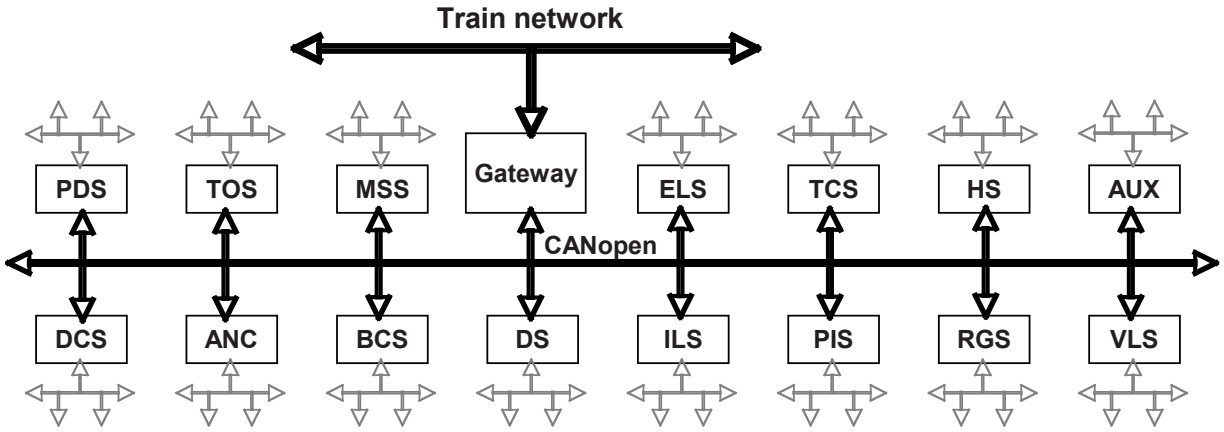


Figure 1 – Logical network architecture of the consist network

4.3 Network topology

The network topology shall be as close as possible to a single line structure with termination resistors on both sides of the network as suggested by ISO11898-2. The total network length shall not exceed 450 m, if a transmission rate of 125 kbit/s is used.

The accumulated stub length shall not exceed 110 m, and the single stub length shall not exceed 22 m, in case a transmission rate of 125 kbit/s is used. It is recommended to keep the stub lengths as short as possible. The CAN transceiver chips shall be galvanically isolated. Optocouplers are placed between CAN controller and CAN transceiver. This affects the maximum bus length depending upon the propagation delay of the optocouplers. The termination resistor on both ends of the bus lines shall be 120 Ω or higher.

The network topology is illustrated in Figure 2.

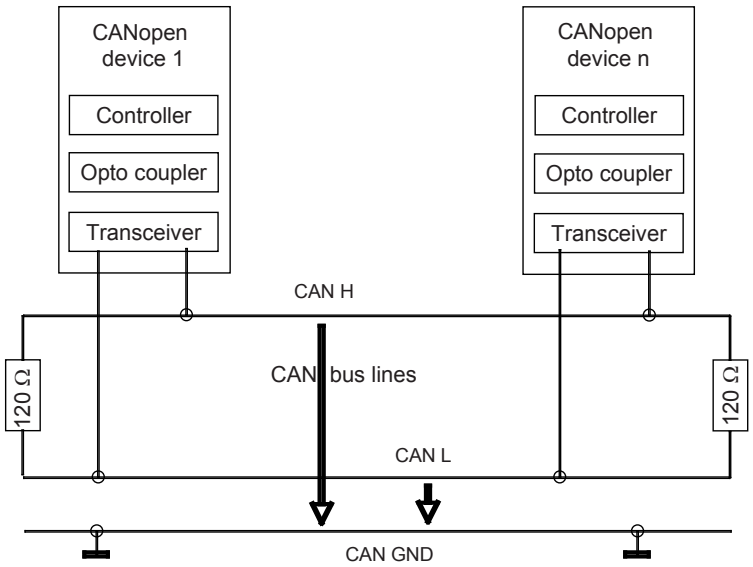


Figure 2 – Network topology of CANopen-based consist network

4.4 Addressing

Any End Device that is connected to a CANopen-based Consist Network requires a unique CANopen Node-ID in the range from 01_h to $7F_h$. The CANopen Node-ID is either adjusted via hardware switches or via software. In case the CANopen Node-ID is adjusted via the CANopen communication interface of the device, the CANopen Layer Setting Services shall

be used. The CANopen Node-ID shall not be adjusted via the CANopen object dictionary of the related device.

The definition of the Layer Setting Services (LSS) is not in the scope of this standard

NOTE LSS are specified in the document CiA 305.

4.5 Data classes

To be able to exchange meaningful data across the CANopen-based Consist Network, it is necessary that the format of this data and its meaning is known by the producer and the consumer(s). This specification models this by the concept of data types.

The encoding rules define the representation of values of data types and the transfer syntax for the representations. Values are represented as bit sequences. Bit sequences are transferred in sequences of octets (bytes). For numerical data types the encoding is little endian style.

Applications often require data types beyond the basic data types. Using the compound data type mechanism, it is possible to extend the list of available data types. Some general extended data types are defined as “Visible String” or “Time of Day”. Compound data types are a means to implement user defined “DEFTYPES” in the terminology of the EN 50325-4 and not “DEFSTRUCTS”.

The data types and encoding rules provided in EN 50325-4 shall apply.

5 Physical layer

5.1 Content

This clause provides definitions for the physical layer of a CANopen-based consist network.

5.2 Cabling

The main trunk circuit shall be at least a single twisted pair of nominal characteristic impedance as defined in ISO 11898-2. In addition a CAN_Ground line as clear reference level for CAN_H and CAN_L voltage level shall be applied.

5.3 Connector

It is recommended that devices connected to a CANopen-based consist network support either the 9-pin D-sub connector or the 5-pin micro style (M12) connector. The connectors are illustrated in Figure 3 and Figure 4. The 9-pin D-sub connector shall support a pinning as defined in Table 1. The 5-pin micro style connector (M12) shall support a pinning as defined in Table 2. As the CANopen devices shall be galvanically isolated, the optional V+ line shall only be used for supply of CAN transceiver and optocoupler, in case no additional power supply is used.

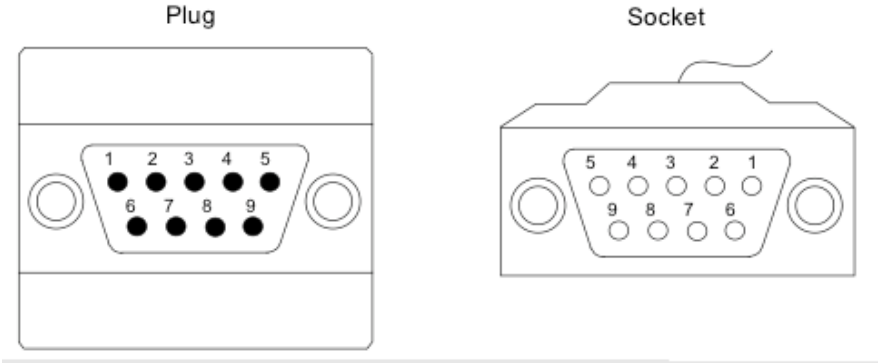


Figure 3 – 9-pin D-sub connector

Table 1 – Pinning for 9-pin D-sub connector

Pin	Signal	Description
1	-	Reserved
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_GND	CAN ground
4	-	Reserved
5	(CAN_SHLD)	Optional CAN shield
6	(GND)	Optional ground
7	CAN_H	CAN_H bus line (dominant high)
8	-	Reserved
9	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers for galvanically isolated bus nodes)

NOTE It is highly recommended not to connect CAN_GND and GND.

In case shielding is required, it is recommended to connect a shield via the metal housing of the used connector. Optionally the CAN_SHLD pin can be used.

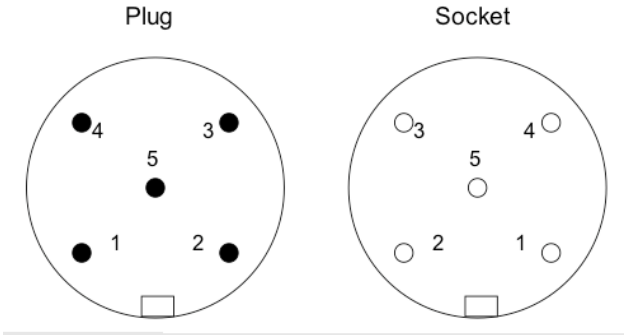


Figure 4 – 5-pin micro style connector

Table 2 – Pinning for 5-pin micro style connector

Pin	Signal	Description
1	(CAN_SHLD)	Optional CAN shield
2	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers, if galvanic isolation of the bus node applies)
3	CAN_GND	Ground / 0V / V-
4	CAN_H	CAN_H bus line (dominant high)
5	CAN_L	CAN_L bus line (dominant low)

5.4 Physical medium attachment

The physical medium for a device connected to CANopen-based consist networks shall be a differentially driven two-wire bus line with common return according to high-speed transmission specification in ISO 11898-2.

Using the high-speed transceiver according to ISO 11898-2 the maximum rating for V_{CAN_H} and V_{CAN_L} shall be +16 V. Galvanic isolation between CANopen devices is optional. It is recommended to use a CAN transceiver that is capable of sustaining misconnection of any of the wires of the connector including the optional V+ voltages of up to 30 V.

5.5 Physical signaling

The bit encoding/decoding and synchronization shall meet the requirements defined in ISO 11898-1.

The bit timing shall meet the requirements defined in ISO 11898-1 and the definitions as given in Table 3. The recommended location of the sample point is as close as possible to 87,5 % of the related bit time.

Table 3 – Bit timing

Bit rate	Nominal bit time t_b µs	Valid range for location of sample point %
1 Mbit/s	1	75 to 90
800 kbit/s	1,25	75 to 90
500 kbit/s	2	85 to 90
250 kbit/s	4	85 to 90
125 kbit/s	8	85 to 90
50 kbit/s	20	85 to 90
20 kbit/s	50	85 to 90
10 kbit/s	100	85 to 90

Devices connected to a CANopen-based consist network shall support a bit rate of 125 kbit/s. Optionally further bit rates as given in Table 3 may be supported.

6 Data Link layer

6.1 Content

This clause provides definitions for the data link layer of a CANopen-based consist network.

6.2 CANopen data link layer

The described CANopen-based consist network shall be based on a data link layer and its sub-layers according to ISO 11898-1.

This specification is based on the CAN base frame format with 11-bit CAN-Identifier. Therefore it is not required to support the CAN extended frame format with 29-bit CAN-Identifier.

NOTE As certain applications require the usage of the CAN extended frame format, the network can be operated in this mode as well, in case all connected CANopen devices support this format.

7 CANopen application layer

7.1 Content

This clause provides definitions with regard to the upper layers of the ISO OSI reference model for devices connected to a CANopen-based consist network.

7.2 Reference model

Devices connected to a CANopen-based consist network compliant to this specification shall use the reference model, as defined in EN 50325-4.

7.3 Field device model

Devices connected to a CANopen-based consist network shall comply with the field device model as given in Figure 5.

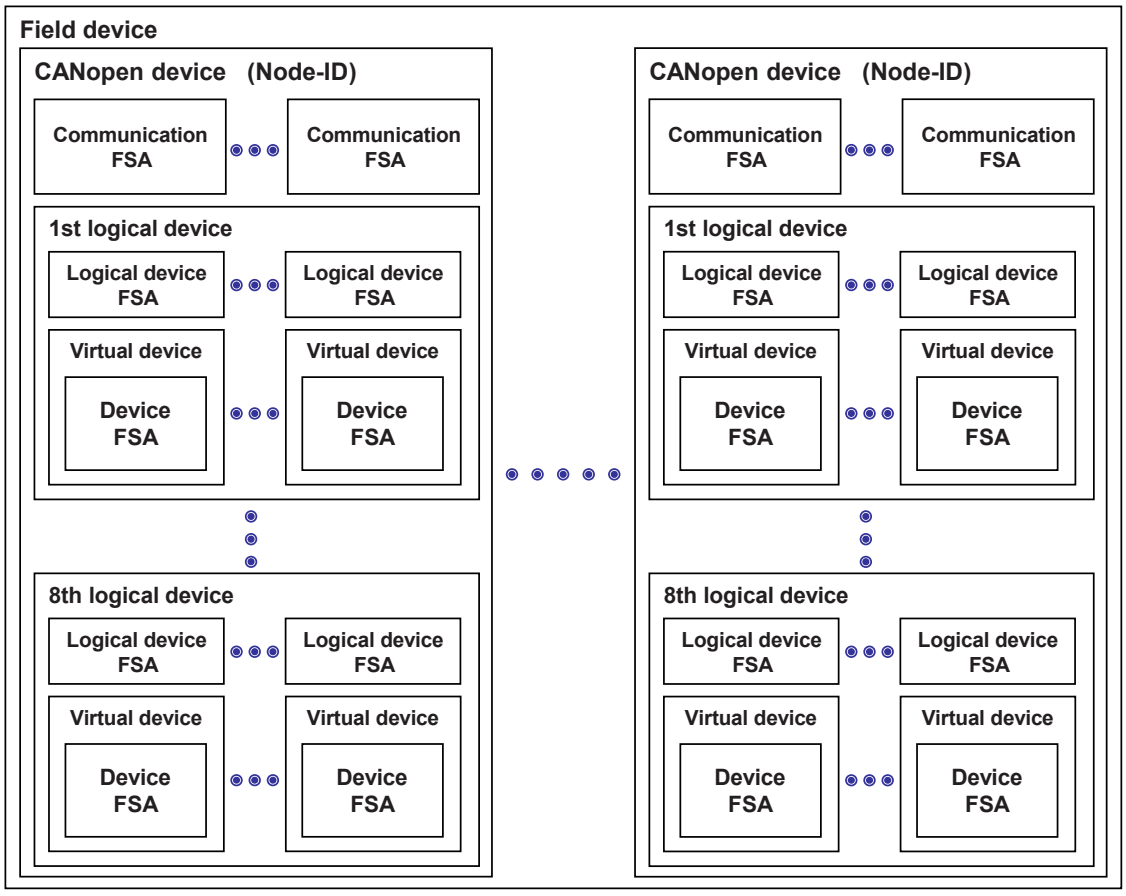


Figure 5 – Field device model

The field device as illustrated in Figure 5 shall provide at least one CANopen device. Each CANopen device within the field device shall provide at least one associated network interface comprising the data link layer protocol (see Clause 6) and the physical layer (see Clause 5), one CANopen Node-ID, and at least one communication FSA. The first communication FSA contains the NMT slave state machine as defined in EN 50325-4. Additional communication FSAs contain an emergency state machine (see EN 50325-4) and others. A CANopen device shall support at least one and may support up to eight logical devices. Each logical device may contain a number of virtual devices and optionally a logical device FSA. A virtual device contains a virtual device FSA and is not distributed to several logical devices. The minimum field device is shown in Figure 6.

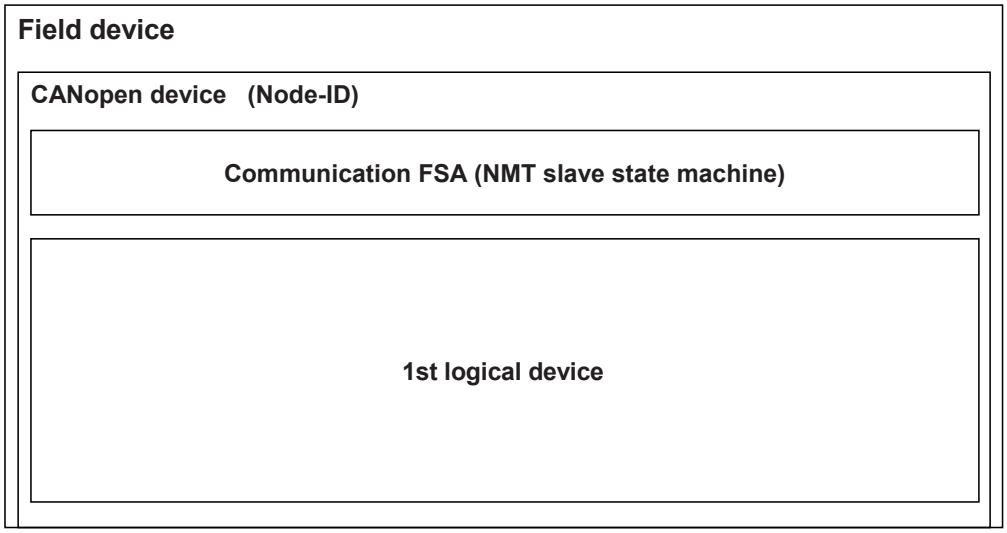


Figure 6 – Minimum field device

A CANopen device is structured as shown in Figure 7:

- Communication – This function unit provides the communication objects and the appropriate functionality to transport data items via the underlying network structure.
- Object dictionary – The object dictionary is a collection of all the data items which have an influence on the behavior of the application objects, the communication objects and the state machine used on this device.
- Application – The application comprises the functionality of the device with respect to the interaction with the process environment.

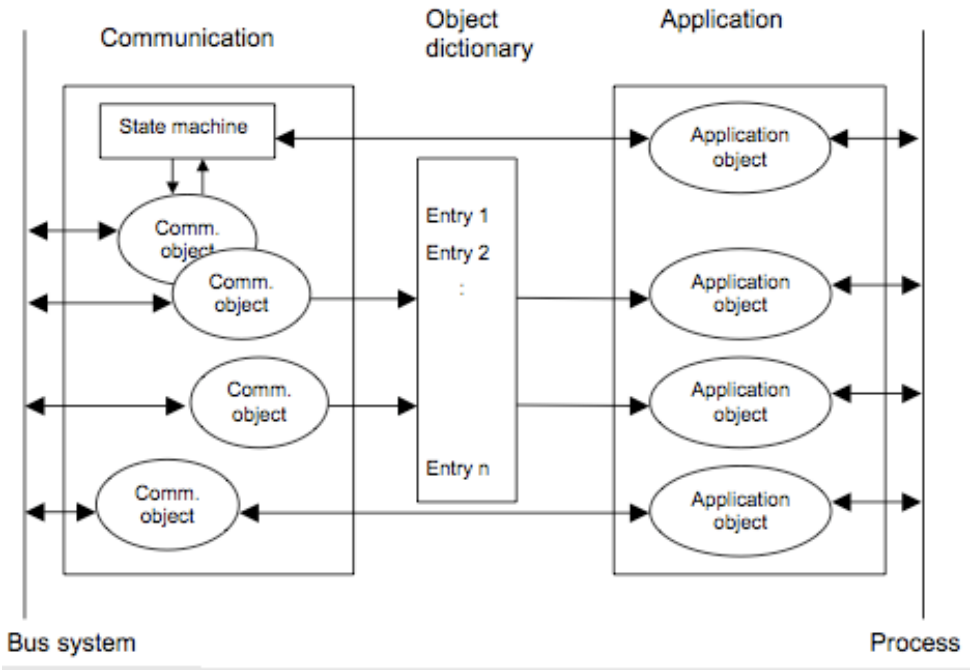


Figure 7 – CANopen device structure

7.4 CANopen communication objects

Devices connected to a CANopen-based consist network shall support all mandatory communication objects of EN 50325-4 as well as the CANopen network management state machine as defined in EN 50325-4.

In order to reduce configuration effort for simple CANopen networks a CAN-ID allocation scheme is defined. These CAN-IDs shall be available in the NMT state Pre-operational directly after the NMT state Initialization (if no modifications have been stored). The objects SYNC, TIME, EMCY write and PDO may be deleted and re-created with new CAN-IDs by means of dynamic distribution. A CANopen device shall provide the corresponding CAN-IDs only for the supported communication objects. The CAN-ID allocation scheme is defined in EN 50325-4.

7.5 CANopen object dictionary

The CANopen object dictionary, as defined in EN 50325-4, contains a maximum of 65536 objects that are addressed through a 16-bit index and up to 256 sub-indices per object, which are addressed through an 8-bit sub-index.

The static data types at indices from 0001_h to $001F_h$ contain type definitions for standard data types like BOOLEAN, INTEGER, UNSIGNED, floating point, string, etc.

Complex data types at indices from 0020_h to $003F_h$ are pre-defined structures that are composed of standard data types and are common to all CANopen devices.

Manufacturer-specific complex data types at indices from 0040_h to $005F_h$ are structures composed of standard data types but are specific to a particular CANopen device.

CANopen device profiles may define additional data types specific to their device type. The static data types and the complex data types defined by the CANopen device profile are listed at indices from 0060_h to $025F_h$.

A CANopen device optionally provides the structure of the supported complex data types (indices from 0020_h to $005F_h$ and from 0060_h to $025F_h$) at read access to the corresponding index. Sub-index 00_h then provides the highest sub-index supported at this index, and the following sub-indices contain the data type encoded as UNSIGNED16 according to EN 50325-4.

The communication profile area at indices from 1000_h to $1FFF_h$ contains the communication specific parameters. These objects are common to all CANopen devices.

The standardized profile area at indices from 6000_h to $9FFF_h$ contains all data objects common to a class of CANopen devices that may be read or written via the network. The CANopen device profiles use objects from 6000_h to $9FFF_h$ to describe parameters and functionality.

The object dictionary concept caters for optional features, which means a manufacturer may not provide certain extended functionality on his CANopen devices but if he wishes to do so he shall do it in a pre-defined fashion. Space is left in the object dictionary at indices from 2000_h to $5FFF_h$ for manufacturer-specific functionality.

The network variables at indices from $A000_h$ to $AFFF_h$ contain input variables and output variables, which are part of a programmable CANopen device.

The system variables at indices from $B000_h$ to $BFFF_h$ contain input variables and output variables, which are part of an underlying CANopen network in a hierarchical sense.

The general CANopen object dictionary structure is illustrated in Table 4.

Table 4 – CANopen object dictionary structure

Index	Object
0000_h	not used
0001_h – $001F_h$	Static data types
0020_h – $003F_h$	Complex data types
0040_h – $005F_h$	Manufacturer-specific complex data types
0060_h – $025F_h$	Device profile specific data types
0260_h – $03FF_h$	reserved
0400_h – $0FFF_h$	reserved
1000_h – $1FFF_h$	Communication profile area
2000_h – $5FFF_h$	Manufacturer-specific profile area
6000_h – $67FF_h$	Standardized profile area 1st logical device
6800_h – $6FFF_h$	Standardized profile area 2nd logical device
7000_h – $77FF_h$	Standardized profile area 3rd logical device
7800_h – $7FFF_h$	Standardized profile area 4th logical device
8000_h – $87FF_h$	Standardized profile area 5th logical device
8800_h – $8FFF_h$	Standardized profile area 6th logical device
9000_h – $97FF_h$	Standardized profile area 7th logical device
9800_h – $9FFF_h$	Standardized profile area 8th logical device
$A000_h$ – $AFFF_h$	Standardized network variable area
$B000_h$ – $BFFF_h$	Standardized system variable area
$C000_h$ – $FFFF_h$	reserved

7.6 Predefined CANopen communication objects

7.6.1 Content

This clause provides the basic CANopen communication capability for CANopen devices that participate in a CANopen-based consist network according to this specification.

7.6.2 Object 1000_h: Device type

This object provides the type of device and its functionality. The value 0000_h for the device profile number indicates a logical device that does not follow a standardized profile. In this case the additional information is 0000_h (if no further logical device is implemented) or $FFFF_h$ (if a further logical device is implemented).

For multiple logical device modules the additional information parameter is $FFFF_h$ and the device profile number referenced by object 1000_h is the profile of the first logical device in the object dictionary. All other profiles of a multiple logical device module identify their profiles at objects $67FF_h + x * 800_h$ with x = internal number of the logical device (from 1 to 8) minus 1. These objects describe the device type of the preceding logical device, having the very same value definition as object 1000_h.

Figure 8 illustrates the structure of the object. The value definition, the object description and the entry description are specified in EN 50325-4.

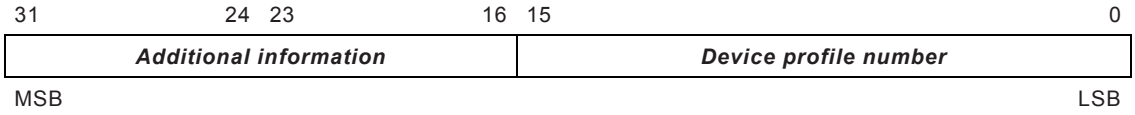


Figure 8 – Structure of the device type object

NOTE Devices connected to CANopen-based consist networks may follow the CANopen application profile for train vehicle control networks CiA 421. This CANopen application profile defines application data, based on UIC 556, that is exchanged in a CANopen-based consist network.

7.6.3 Object 1001_h: Error register

This object provides the error information. The CANopen device maps internal errors into this object. This error information is published as part of the emergency message. The value definition, object description and entry description are specified in EN 50325-4.

7.6.4 Object 1014_h: COB-ID emergency object

This object shall be implemented. It is specified in EN 50325-4. The CAN-ID, which is part of this object, shall not be changed.

7.6.5 Object 1017_h: Heartbeat producer

All CANopen devices that are connected to a CANopen-based consist network shall implement this object. It is specified in EN 50325-4.

The devices shall support heartbeat message transmissions from 100 ms to 1 000 ms.

7.6.6 Object 1018_h: Identity object

This object provides general information about the device as specified in EN 50325-4.

7.6.7 Object 1029_h: Error behavior

This object specifies to which state the device is set, in case a communication error or a device-internal error is detected. It is specified in EN 50325-4.

7.6.8 Object $67FF_h$: Device type

This object shall describe the first logical device in a multiple device module according to EN 50325-4.

7.6.9 Service data objects (SDOs)

Any CANopen device supports the first SDO server channel. CANopen devices connected to a CANopen-based consist network may support additional SDO server or client channels.

In case an additional SDO channel is supported, the related SDO parameter set is supported in the CANopen object dictionary as defined in EN 50325-4.

There are no further SDO channels pre-defined by this specification.

7.6.10 Process data objects (PDOs)

CANopen devices operated in a CANopen-based consist network may support up to 512 PDOs in transmit as well as receive direction.

In case a CANopen device supports a PDO, the related PDO communication parameter and mapping entries are supported in the CANopen object dictionary as defined in EN 50325-4.

There are no PDOs pre-defined.

8 Application data

8.1 Content

This clause provides the representation of application data that is communicated between a CANopen-based consist network and a Train Backbone via a Gateway.

NOTE 1 The application data is described in application profile of TCN.

In general application data is managed in the CANopen object dictionary index ranges 2000_h to $5FFF_h$ and 6000_h to $9FFF_h$. A manufacturer specific device behavior is controlled via the index range 2000_h to $5FFF_h$. A standardized CANopen device behavior is controlled via objects in the index range 6000_h to $9FFF_h$.

NOTE 2 Object dictionary entries in the index range 6000h to 9FFFh are specified in CANopen device and application profiles and available at CAN in Automation.

8.2 CANopen application data representation

As the application data for TCN is not defined yet, the representation of the TCN application data in the standardized object dictionary index range is not defined.

8.3 Recommended representation principle of application data

8.3.1 Content

In this clause a representation principle of application data is provided for the representation of application data in CANopen-based consist networks. Therefore it is possible to map the process to CANopen process data objects (PDOs) for process data transfer.

8.3.2 Application data for door control

This clause provides the recommended representation principle by means of door control information, which is communicated between a CANopen-based consist network and a Train

Backbone via a Gateway. The represented process data is therefore available in the CANopen-based consist network and can be communicated via the CANopen communication objects.

The access type is given for the interface of the gateway device to the CANopen-based consist network. Other devices such as e.g. door controllers or door units, which are connected to a CANopen-based consist network manage the data as defined in this clause, at the given CANopen object dictionary index. The access type of the object may be changed to the appropriate access type as defined in EN 50325-4.

NOTE The illustrated application data is derived from the document UIC 556 and is given in the representation of the CANopen application profile CiA 421 train vehicle control network.

8.3.3 Consumed door control application objects

8.3.3.1 Object 6007_h: External door status word export

This object shall indicate the status of the external doors of the local rail vehicle. Via the gateway to the Train Backbone, this information is available in the TCN. Figure 9 specifies the object structure and Table 5 specifies the value definition. Table 6 specifies the object description and Table 7 specifies the entry description.

NOTE The object shall correspond to R3-telegram octet 20, which is defined in UIC 556.

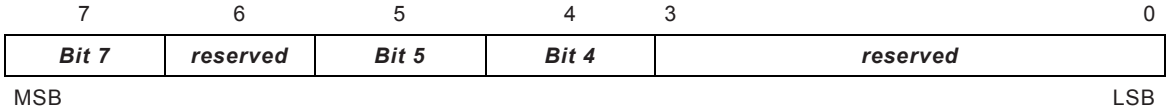


Figure 9 – Object structure

Table 5 – Value definition

Bit	Value	Value definition
Bit4	0	At least one left door is open
	1	All left doors locked
Bit 5	0	At least one right door is open
	1	All right doors locked
Bit 7	0	Side selective door blocking is not in operation
	1	Side selective door blocking is in operation
Reserved		Reserved (shall be ignored)

Table 6 – Object description

Attribute	Value
INDEX	6007_h
Name	External doors status word export
Object code	Variable
Data type	UNSIGNED8
Category	Optional

Table 7 – Entry description

Attribute	Value
Sub-index	00_h
Access	rw
PDO mapping	Optional
Value range	See value definition
Default value	Manufacturer specific

8.3.4 Produced door control application objects

8.3.4.1 Object 6006_h: External door status word import

This object shall provide the status of the external doors of other rail vehicles. Each sub-index shall provide the door status of that vehicle, which correspondes with the UIC vehicle number. Therefore the external door status of other vehicles is available within the CANopen-based consist network. The overall door status of all vehicles is provided in sub-index 21_h. Figure 10 specifies the object structure and Table 8 specifies the value definition. Table 9 specifies the object description and Table 10 specifies the entry description.

NOTE The object shall correspond to R3-telegram octet 20, which is defined in UIC 556.

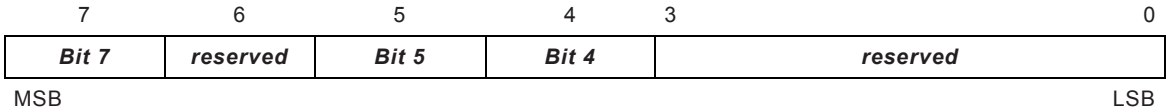


Figure 10 – Object structure

Table 8 – Value definition

Bit	Value	Value definition
Bit4	0	At least one left door is open
	1	All left doors locked
Bit 5	0	At least one right door is open
	1	All right doors locked
Bit 7	0	Side selective door blocking is not in operation
	1	Side selective door blocking is in operation
Reserved		Reserved (shall be ignored)

Table 9 – Object description

Attribute	Value
INDEX	6006_h
Name	External doors status word import
Object code	ARRAY
Data type	UNSIGNED8
Category	Optional

Table 10 – Entry description

Attribute	Value
Sub-index	00_h
Description	Highest subindex supported
Entry category	Mandatory
Access	ro
PDO mapping	No
Value range	01_h to 21_h
Default value	No
Sub-index	01_h
Description	Door status UIC vehicle 1
Entry category	Optional
Access	ro
PDO mapping	Optional
Value range	see Table 46
Default value	No
to	
Sub-index	20_h
Description	Door status UIC vehicle 32
Entry category	Optional
Access	ro
PDO mapping	No
Value range	Optional
Default value	No
Sub-index	21_h
Description	Door status train
Entry category	Optional
Access	ro
PDO mapping	No
Value range	Optional
Default value	No

8.3.4.2 Object 6001_h: External door command

This object shall provide the first control word for the external doors. Figure 11 specifies the object structure and Table 11 specifies the value definition. Table 12 specifies the object description and Table 13 specifies the entry description.

NOTE The object shall correspond to R3-telegram octet 20, which is defined in UIC 556.

7	6	5 4	3	2	1	0
Reserved	***Bit 6***	***Reserved***	***Bit 3***	***Bit 2***	***Bit 1***	***Bit 0***
MSB						LSB

Figure 11 – Object structure

Table 11 – Value definition

Bit	Value	Value definition
Bit 0	0	Inactive
	1	All doors close
Bit 1	0	Inactive
	1	Interrupt door closing
Bit 2	0	Release all left doors
	1	Lock all left doors
Bit 3	0	Release all right doors
	1	Lock all right doors
Bit 6	0	Not extend footstep
	1	Extend footstep
Reserved		Reserved (shall be ignored)

Table 12 – Object description

Attribute	Value
INDEX	6001_h
Name	External door command 1
Object code	Variable
Data type	UNSIGNED8
Category	Optional

Table 13 – Entry description

Attribute	Value
Sub-index	00_h
Access	ro
PDO mapping	Optional
Value range	See value definition
Default value	No

9 CANopen network management

9.1 Content

This clause provides the network management for CANopen-based consist networks. The definition of the network management includes the definition of the network startup behavior as well as definitions that are related to networks that operate without NMT master, networks with one CANopen device capable of the NMT master mode. These definitions are intended to be an add-on to the CANopen application layer and communication profile provided in EN 50325-4.

CANopen consist networks with more than one CANopen device capable of the NMT master mode (NMT flying master) are not in the scope of this specification. However such networks are not excluded.

NOTE With regard to an increased availability, within CANopen-based consist networks several CANopen devices with NMT master functionality may reside. As at a certain time, only one active NMT master is allowed, mechanisms are required, that enable overtaking of the NMT master functionality from one CANopen device to another. These services are part of the CANopen flying master functionality that is described in CiA 302.

9.2 CANopen NMT slave functionality

Devices operated in a CANopen-based consist network shall provide CANopen NMT-slave functionality and shall implement all mandatory communication services and protocols as well as all mandatory objects as specified in EN 50325-4. The minimum communication objects are defined in 7.6.

Optionally further communication objects may be supported. In addition the CANopen manager functionality may be supported.

9.3 CANopen manager functionality

9.3.1 General

This subclause provides the definition of the CANopen manager functionality for CANopen-based consist networks.

Besides the application process several different additional functionalities exist in a CANopen network. These functionalities are referred to by different terms. This subclause is intended to clarify these terms.

Within a distributed system the application process is divided into several parts running on different CANopen devices. From the applications point of view usually one CANopen device is responsible for the control of the system. This CANopen device is called *application master*.

From the network's point of view there are several additional functionalities, which not directly deal with the application but provide application-supporting functions. These additional functionalities are based on a master/slave, client/server or producer/consumer relationship.

As it is common to combine several of the additional functionalities in one CANopen device the term CANopen Manager is introduced.

A CANopen device is referred to as CANopen manager, if it provides the NMT master functionality and at least one of the functionalities SDO manager or Configuration manager.

9.3.1.1 NMT master

The network management (NMT) provides services for controlling the network behavior of CANopen devices as defined in EN 50325-4. All CANopen devices that are participating in a CANopen consist network, referred to as NMT slaves, are controlled by services provided by an NMT master. Usually the NMT master application is also part of the application master.

The device, in which the NMT master functionality resides is a fully CANopen device. In addition to the NMT master functionality it supports all functions and objects, which are indicated as mandatory in EN 50325-4.

9.3.1.2 Flying master

The flying master mechanism provides services for a hot stand-by NMT master within a CANopen network. The flying master is an optional functionality within a CANopen device. The CANopen device implementing the flying master implements the NMT master functionality. The definition of the responsibilities, functionalities, and services of the flying master is not in the scope of this document.

NOTE Definitions for the flying master functionality are provided in CiA 302.

9.3.1.3 SDO manager

The SDO manager is an optional functionality, responsible for handling of the dynamic establishment of SDO connections. If an SDO manager is present in a CANopen network it resides together with the NMT master on the same CANopen device.

NOTE Definitions for the SDO manager functionality are provided in CiA 302.

9.3.1.4 Configuration manager

The Configuration Manager is an optional functionality, which provides mechanisms for configuration of CANopen devices in a CANopen network during boot-up. The mechanisms are called Configuration Management (CMT). If the Configuration Manager is present in a CANopen network it resides together with the NMT master on the same CANopen device.

NOTE Definitions for the Configuration manager functionality are provided in CiA 302.

9.3.1.5 SYNC producer

The SYNC producer is an optional functionality, which is responsible for transmitting the SYNC object. It may reside on any CANopen device in the CANopen-based consist network. The relevant CANopen object dictionary entries are defined in EN 50325-4.

9.3.1.6 TIME producer

The TIME producer is an optional functionality, which is responsible for transmitting the TIME STAMP object. It may reside on any CANopen device in the CANopen-based consist network. The time stamp object is defined in EN 50325-4.

9.3.1.7 LSS master

The layer setting services (LSS) provide services for configuring layer 2 (bit timing) and NMT (CANopen Node-ID) via CAN. The LSS master configures LSS slaves by executing LSS services. The definition of the layer setting services is not in the scope of this document.

NOTE Definitions for the layer setting services are provided in CiA 305.

9.3.2 Object dictionary usage

Several objects relating to the configuration and validation of CANopen devices are of object type ARRAY. The sub-index of the entries of such an ARRAY corresponds to the CANopen Node-ID of the CANopen device. Those objects may have less than 127 entries. In that case all of these objects shall have the very same set of supported entries. Sub-index 00_h shall provide the highest sub-index supported at this index. Related object description in Clause 9 refer to this as *SupportedNodeID* condition.

NOTE For a CANopen manager to be universally usable it is recommended to support the full sub-index range of 01_h to $7F_h$.

9.3.3 Redundant networks

Redundant networks are required for high-availability applications. Redundant networks consist of at least two CAN lines. This allows communication between CANopen devices, in case of a single failure within the physical interconnection between the CANopen devices. The definition of the responsibilities, functionalities, and services for redundant networks is not in the scope of this document.

NOTE Definitions for setting up redundant CANopen networks are provided in CiA 302.

9.4 CANopen NMT start-up

9.4.1 NMT startup

CANopen managers shall behave according to the NMT slave state machine as defined in EN 50325-4. Before transition from NMT state *Pre-operational* to NMT state *Operational* of the CANopen manager, all assigned NMT slaves shall be booted. The main flow chart for the procedure is defined in Figure 12 and Figure 13. The simplest flow chart for the startup is defined in Figure 14.

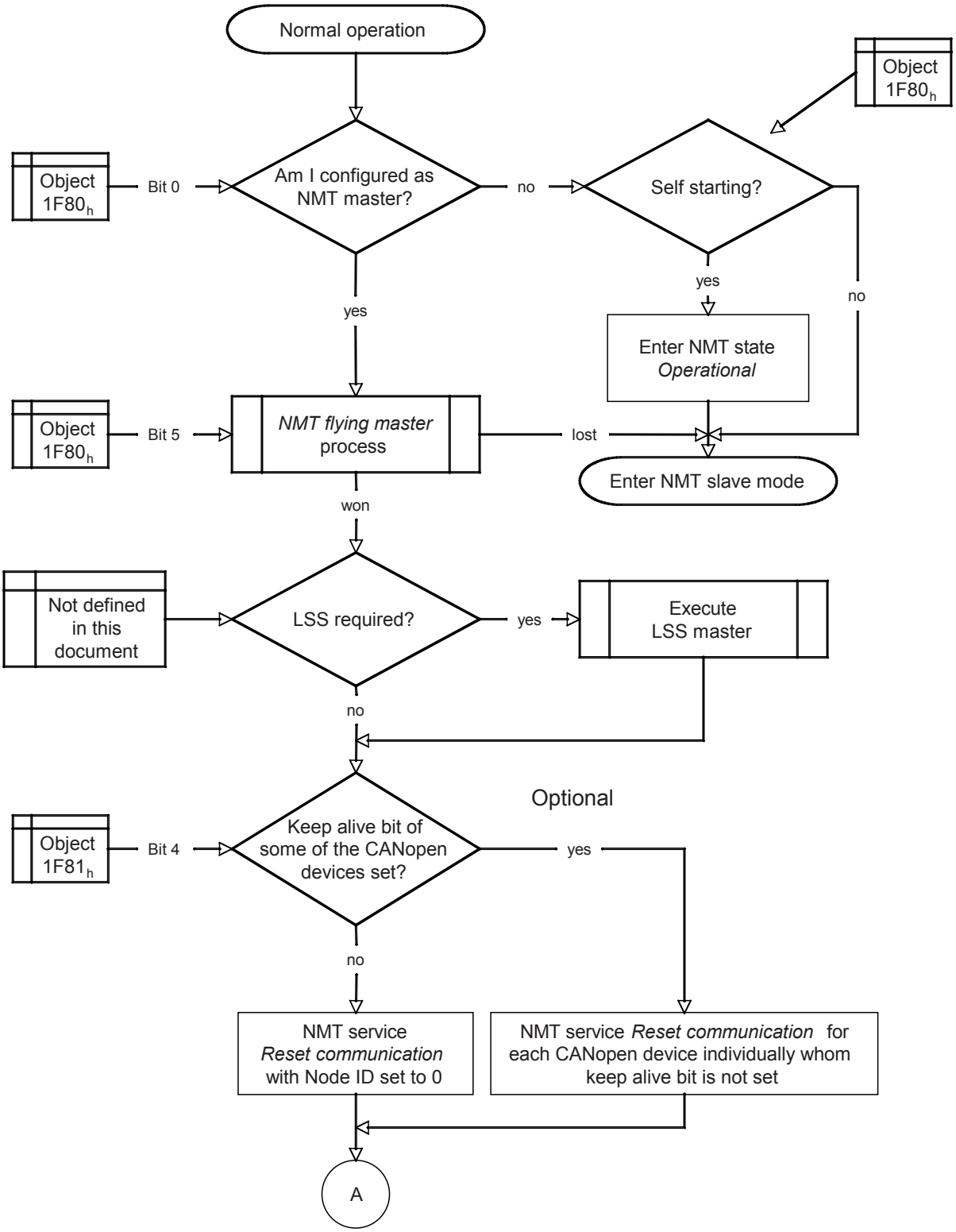


Figure 12 – NMT startup, part 1

The process NMT startup as shown in Figure 12 consists of the following basic steps:

NOTE 1 The process execute LSS master is not in the scope of this standard.

a) Bit 0 of object $1F80_h$ (see clause 9.8.7) is used to decide whether this CANopen device shall be the NMT master. If the CANopen device is NMT master the process shall continue. If the CANopen device is configured as self-starting device, it shall enter the NMT state Operational automatically. In case the CANopen device is not the NMT master the process shall end.

b) Bit 5 of object $1F80_h$ (see clause 9.8.7) is used to decide whether this CANopen device shall participate in the service *NMT flying master negotiation*. If the CANopen device shall participate in the service *NMT flying master negotiation* and the CANopen device lost the service *NMT flying master negotiation* the CANopen device shall not become NMT master. The definition of the NMT flying master negotiation is not in the scope of this document.

NOTE 2 The description of the NMT flying master negotiation is provided in CiA 302.

c) If LSS is required to set the CANopen Node-ID and bit rate of other CANopen devices within the network the NMT master shall execute the LSS master services. The LSS master services may be executed at any time. The precise definition of the LSS master services is not in the scope of this document.

NOTE 3 The description of the layer setting services is provided in CiA 305.

d) Bit 4 of all entries of object $1F81_h$ (see clause 9.8.8) is used to decide whether the NMT master shall perform the NMT service *Reset communication* with CANopen Node-ID set to 0 or if the NMT service *Reset communication* shall be performed for each CANopen device in the network individually. If at least for one entry of object $1F81_h$ bit 4 is set to 1_b and the corresponding CANopen device is in NMT state *Operational* the NMT master shall not issue the NMT service *Reset communication* with CANopen Node-ID set to 0. In this case each CANopen device shall be reset individually. This shall also include all CANopen Node-IDs that are not part of the slave list $1F81_h$.

NOTE 4 This will force potentially existing CANopen devices that are not configured in the slave list to transmit a boot-up message. By this the CANopen Manager will recognize unconfigured CANopen devices via the boot-up handler (Figure 23). The NMT service Reset communication shall not apply to the NMT master itself.

If after step d) the NMT service *Reset communication* is necessary, for any reason, then the NMT service *Reset communication* shall be executed with the CANopen Node-ID of that NMT slave. This shall apply independently from the configuration for that NMT slave in object $1F81_h$. The process NMT startup resumes in Figure 13.

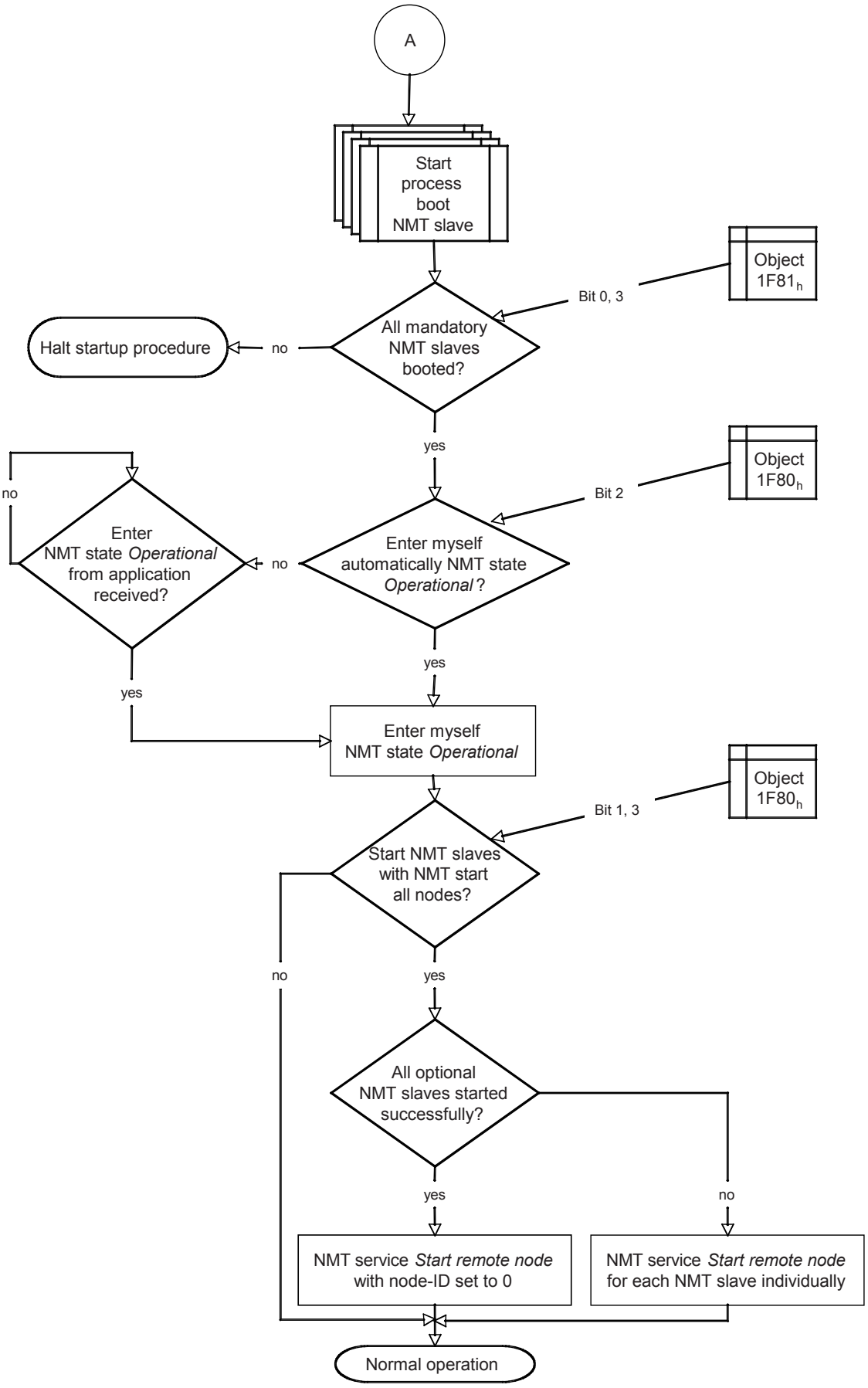


Figure 13 – NMT startup, part 2

e) The NMT master shall start the process *start process boot NMT slave* for all NMT slaves as shown in Figure 15. For all NMT slaves that are marked as mandatory (bit 0 and bit 3 of object $1F81_h$; see 9.8.8) the process *start process boot NMT slave* shall terminate successfully.

f) If an error is detected during process *boot slave* for the NMT slaves that are marked as mandatory the process *NMT startup* shall be stopped.

g) Bit 2 of object $1F80_h$ (see 9.8.7) is used to decide whether the NMT master shall enter the NMT state *Operational* automatically by itself or shall wait until it is requested by the application running on the very same CANopen device.

h) Under the following conditions

   – Bit 3 of object $1F80_h$ is set to 0_b,
   – Bit 1 of object $1F80_h$ is set to 1_b,
   – and all NMT slaves listed in $1F81_h$ booted successfully

   the NMT service *Start remote node* shall be performed with CANopen Node-ID set to 0.

   Under the following conditions

   – Bit 3 of object $1F80_h$ is set to 0_b,
   – Bit 1 of object $1F80_h$ is set to 1_b,
   – and not all NMT slaves listed in $1F81_h$ booted successfully

   the NMT service *Start remote node* shall be performed for each NMT slave individually.

i) The process *NMT startup* ended successfully and the NMT master shall proceed with normal operation.

Occurrence and detection of NMT slaves not listed in $1F81_h$ falls into the responsibility of the application.

9.4.2 NMT startup simple

Since nearly all objects and features are optional it is possible to implement a basic NMT master, which may make sense for some applications. Removing all optional parts in the definitions above results in the process *NMT startup simple* as shown in Figure 14. Even a simple NMT master supports all mandatory objects, which are specified in EN 50325-4 and which are defined in this standard.

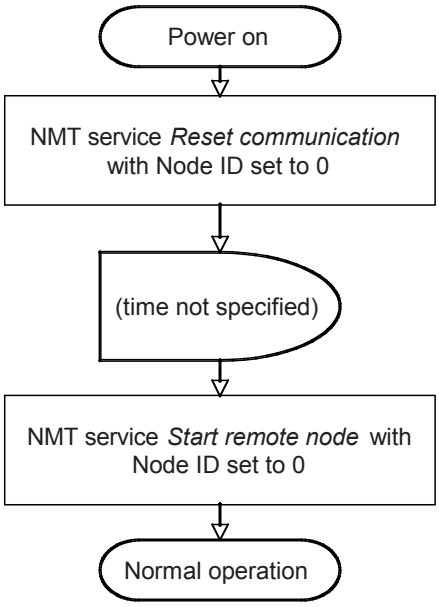


Figure 14 – NMT startup simple

9.4.3 Start process boot NMT slave

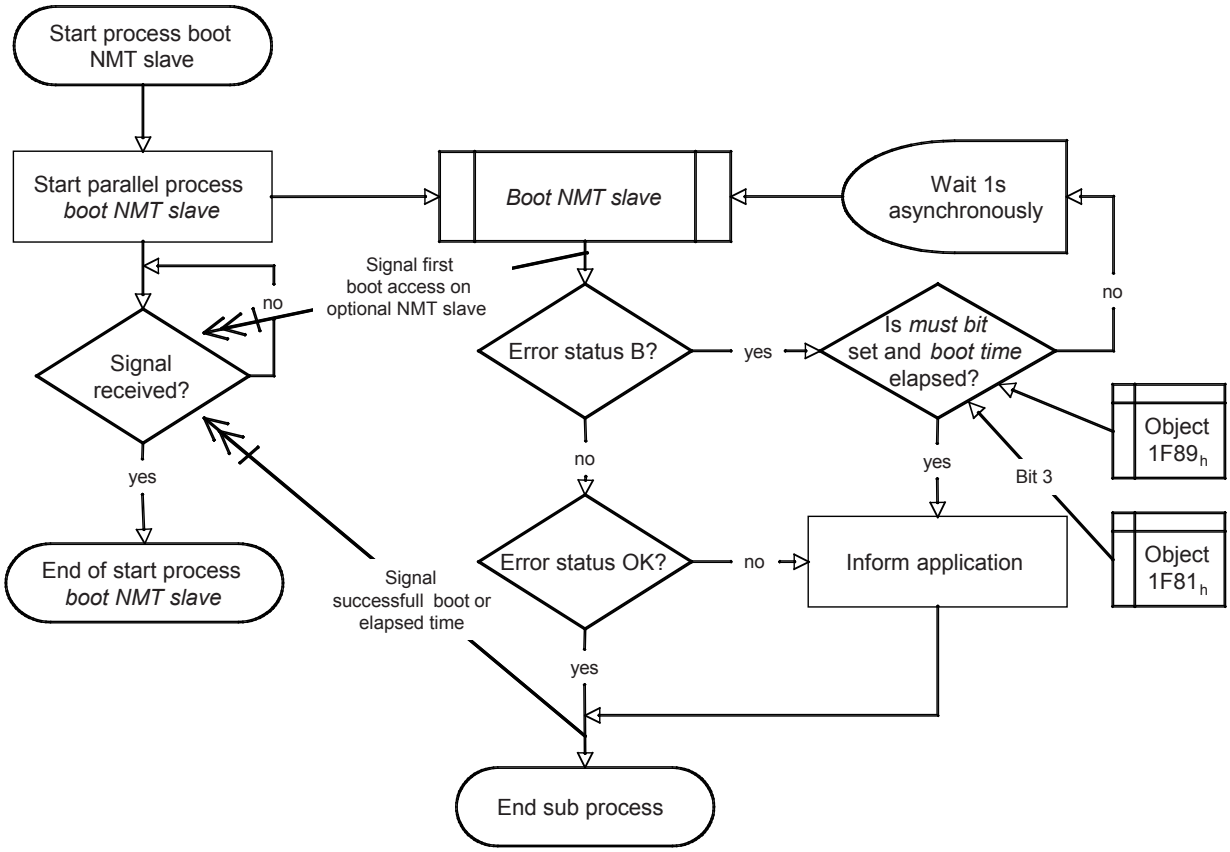


Figure 15 – Start process boot NMT slave

The process *start process boot NMT slave* as shown in Figure 15 shall include the following steps:

a) Start parallel process *boot NMT slave*.

b) Mandatory NMT slaves: wait for completion of the process *boot NMT slave*.
Optional NMT slaves: wait for signal that the process *Boot NMT slave* was performed.

The parallel process shall

c) Perform the process boot NMT slave (see 9.5).

d) Create signal for every try of the process *boot NMT slave*.

If the process *boot NMT slave* returned with status OK the process shall terminate. This process shall run endlessly for any optional NMT slave until the process *Boot NMT slave* finishes with status OK.

NOTE The recommended cycle time is 1 s for bit rate higher than 125 kbit/s.

If the process *boot NMT slave* returned with error status B for mandatory NMT slaves and the elapsed time is greater than the configured value of object $1F89_h$ then the application shall be informed and this sub-process shall end.

The sub-process of the process *start process boot NMT slave* shall run asynchronously to other processes.

9.5 Boot NMT slave

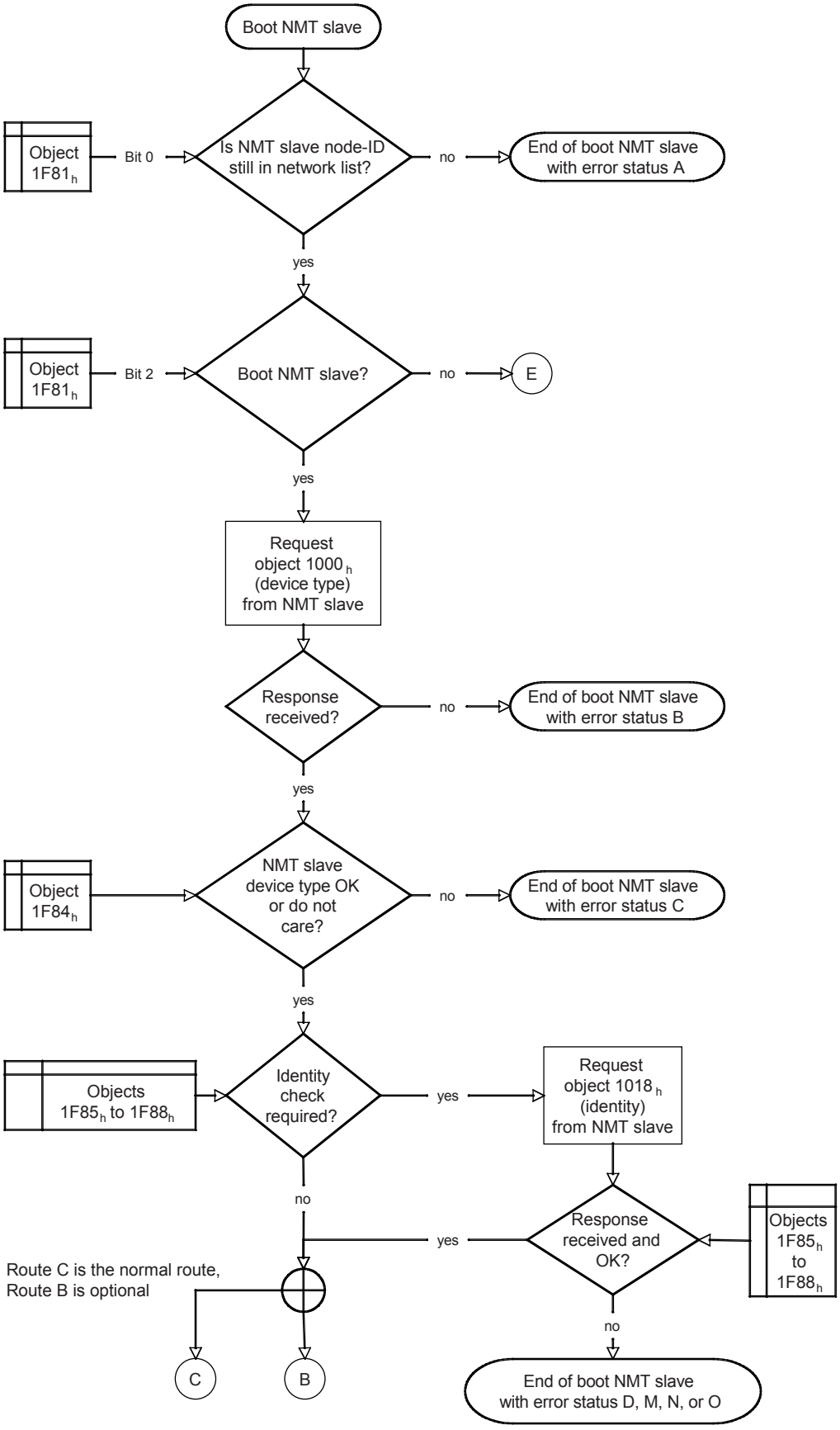


Figure 16 – Boot NMT slave, part 1

The process *boot NMT slave* as shown in Figure 16 consists of the following steps:

a) Bit 0 of object $1F81_h$ is used to decide upon whether the NMT slave shall be processed or if the process shall terminate with error status.

b) Bit 2 of object $1F81_h$ is used to decide upon whether the NMT slave shall be configured and started.

c) Upload object 1000_h from the NMT slave. In case no response is received the process shall terminate with error status.

d) In case the value of object $1F84_h$ for the NMT slave is unequal 0 the value shall be checked against object 1000_h. In case both values are different the process shall terminate with error status.

e) In case the values of objects $1F85_h$ to $1F88_h$ are unequal 0, the particular object values shall be checked against their corresponding values of the object 1018_h from the NMT slave. In case one of the values of object $1F85_h$ to $1F88_h$ is different to the corresponding values of the object 1018_h from the NMT slave the process shall terminate with error status.

The process *boot NMT slave* may continue with an optional part 2 (see Figure 17), which introduces two optional features:

- keeping alive CANopen devices initially in NMT state *Operational*, and
- managing application software versions including automatic update of the applicationsoftware.

If the keep alive bit of a CANopen device is set the NMT master shall not issue the NMT services *Reset node* and *Reset communication* for this CANopen device.

NOTE 1 Such situations may occur, in case the NMT master encounters a failure with a subsequent re-start, e.g. power fail.

Software version control may be used in systems where the version of the application software running on the CANopen device is checked for correctness. The process check and update software version may be used to automatically download the most current version of the application software. CANopen devices that detect an error in their application software, for example by calculating a checksum during startup, may force a download of the most current application software by responding with wrong version information.

Both or only one of the two features may be implemented.

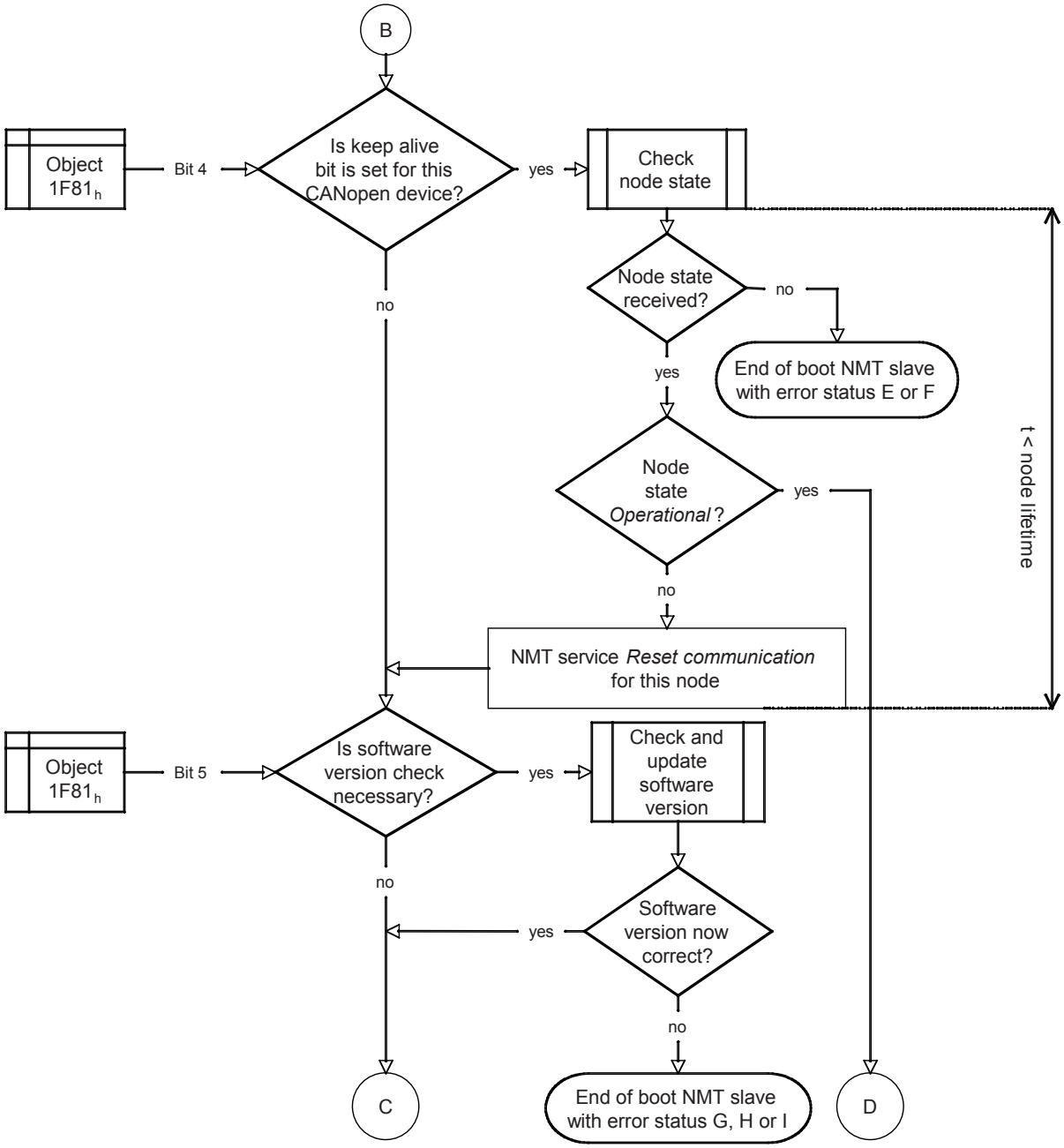


Figure 17 – Boot NMT slave, part 2

f) Bit 4 of object $1F81_h$ is used to decide whether keep alive is requested.

In case keep alive is requested the NMT master shall request the current NMT state of the NMT slave for which the *boot NMT slave* is performed. In case no current NMT state is received the process shall finish with error status. In case the current NMT state is *Operational* the process shall continue with D. In case the current NMT state is not the NMT state *Operational* the NMT master shall perform the NMT service *Reset communication* for that NMT slave.

The process check NMT state is defined in 9.5.2.

NOTE 2 In case the NMT slave supports Node and Life guarding only the NMT master takes care that the time from requesting the current NMT state from the NMT slave to issuing the NMT service *Reset communication* is less than the life time as set in the NMT slave.

g) Bit 5 of object $1F81_h$ is used to decide whether the application software verification shall be performed.

In case application software verification is requested, the process check and update software version shall be performed.

In case the software version stays incorrect the process Boot NMT slave shall end with error status.

The process *boot NMT slave* continues in part 3 (see Figure 18) and is mandatory.

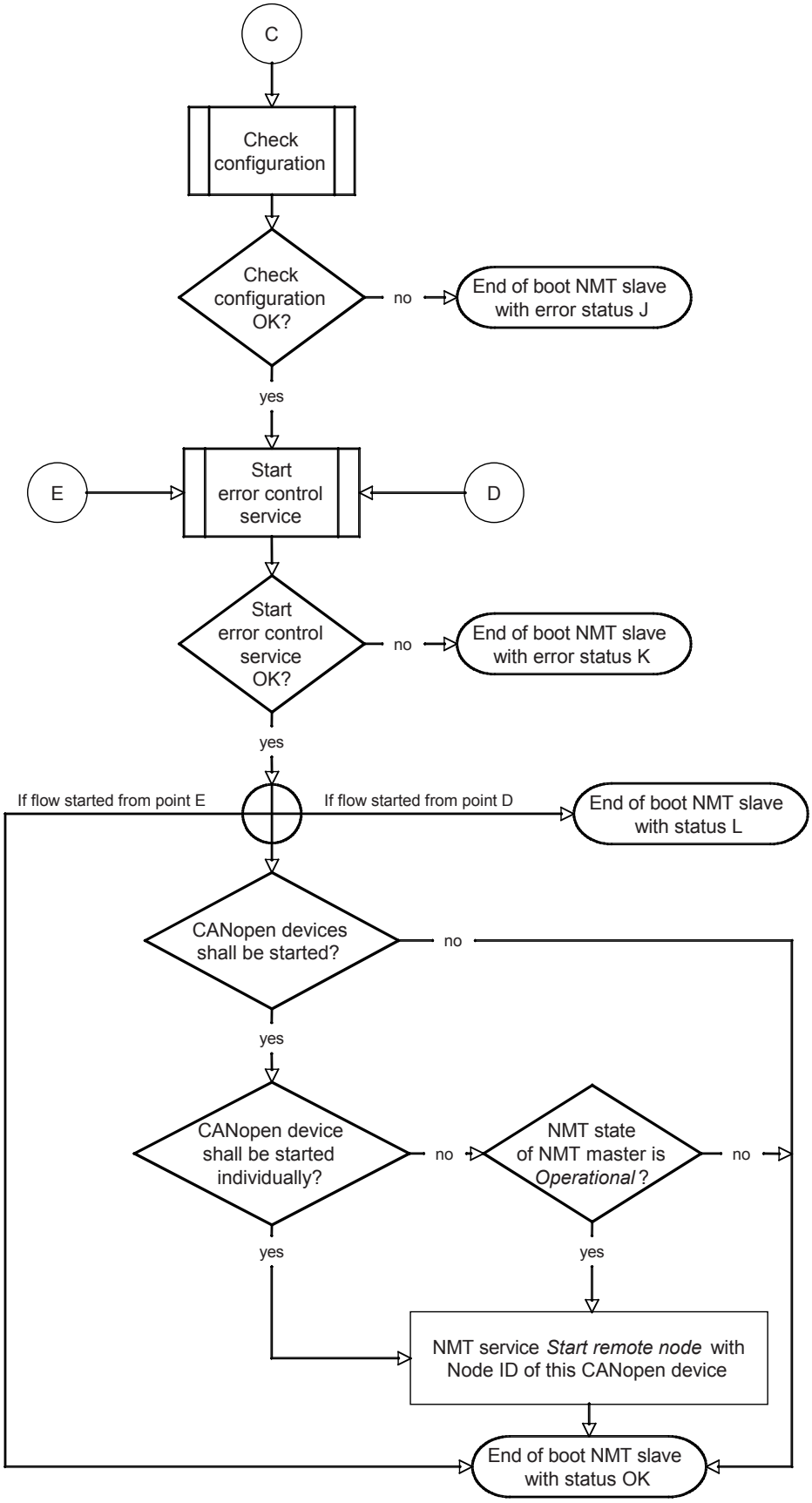


Figure 18 – Boot NMT slave, part 3

h) The process check configuration (see 9.5.1) shall be performed.

i) In case the process check configuration finished unsuccessfully the process *boot NMT slave* shall end with an error status.

j) The process *start error control* (see 9.6.1) shall be performed.

k) In case the process *start error control* finished unsuccessfully the process *boot NMT slave* shall end with an error status.

l) In case the NMT slave is in NMT state *Operational* the process *boot NMT slave* shall finish successfully.

m) Bit 3 of object $1F80_h$ is used to decide whether the NMT master shall execute the NMT service *Start remote node*.

n) In case the NMT master shall not execute the NMT service *Start remote node* the process *boot NMT slave* shall end successfully with status.

o) Bit 1 of object $1F80_h$ is used to decide upon whether the NMT master shall execute the NMT service *Start remote node* with CANopen Node-ID set to 0 or for each NMT slave in the network individually.

p) In case the NMT master shall execute the NMT service *Start remote node* with CANopen Node-ID set to 0 and the NMT master is not in NMT state *Operational* the process *boot NMT slave* shall finish successfully with status OK.

NOTE 3 If the NMT master is in NMT state *Operational* it is regarded as an indication that the initial NMT startup has been completed prior to the startup of this NMT slave.

q) In case the NMT master executes the NMT service *Start remote node* for each NMT slave in the network individually or in case the NMT master executes the NMT service *Start remote node* with CANopen Node-ID set to 0 and the NMT master is in NMT state *Operational* the NMT master shall execute the NMT service *Start remote node* with CANopen Node-ID set to the appropriate value.

r) The process *boot NMT slave* shall finish successfully with status.

9.5.1 Check configuration

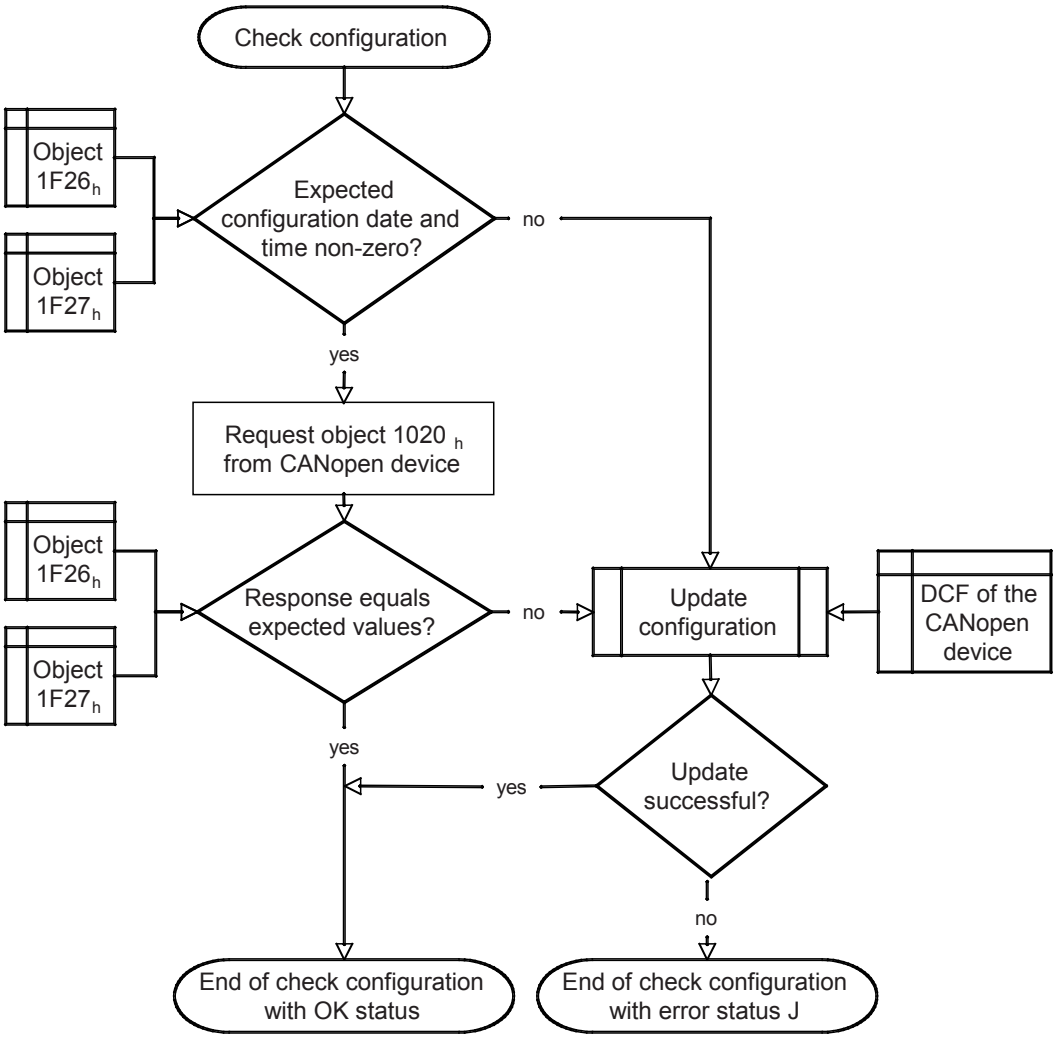


Figure 19 – Check configuration

The process *check configuration* as shown in Figure 19 consists of the following steps:

a) The corresponding entries (sub-index shall equal CANopen Node-ID) of object $1F26_h$ and object $1F27_h$ are used to decide whether the CANopen devices configuration is verified. If the values are equal to 0 the process shall go to step d).
If the values are not equal to 0 the process shall go to step b).

b) Object 1020_h shall be requested from the CANopen device.

c) The values of the corresponding entries (sub-index shall equal CANopen Node-ID) of object $1F26_h$ and object $1F27_h$ shall be compared to the values of the received object 1020_h. If the values are equivalent the process shall finish with status OK.
If the values are not equivalent the process shall go to step d).

d) The configuration shall be updated on the CANopen device.
Depending on the settings of $1F81_h$ and $1F8A_h$ the configured restore operation shall be executed. Only after the restore operation the NMT service reset communication or NMT service reset node for the corresponding NMT slave shall be issued.
The configuration may be derived from any kind of DCF provided for that CANopen device. The DCF may be read from a local file system, object $1F20_h$, or object $1F22_h$ on the configuration manager.

e) If the download is successful then the process shall finish with status OK. If the download is not successful then the process shall finish with error status.

The process *check configuration* may be skipped if the keep-alive is enabled for that CANopen device and the CANopen device is in the NMT state *Operational*. If this occurs the process boot NMT slave shall finish with error status (L — "NMT slave was initially operational").

9.5.2 Check NMT state

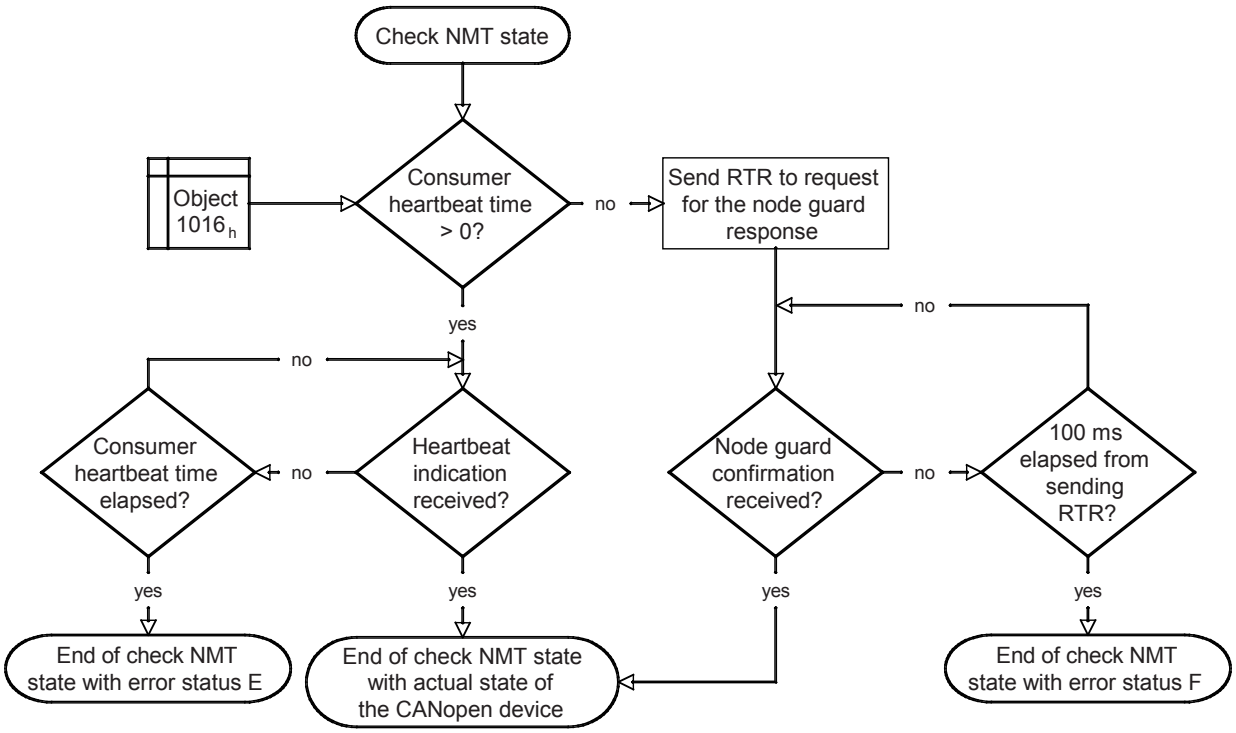


Figure 20 – Check NMT state[1]

The process *check NMT state* as shown in Figure 20 consists of the following steps:

a) Object 1016_h (see EN 50325-4) shall be used to decide whether the CANopen device is set up for heartbeat.

b) In case the CANopen device is set up for heartbeat the NMT master shall check if it received a heartbeat indication in time (heartbeat consumer time is not elapsed). If the heartbeat indication is not received in time the process shall end with an error status. If the heartbeat indication is received the process shall end with the actual NMT state of the CANopen device that is checked.

c) In case the CANopen device that is checked is not set up for heartbeat it is assumed that the CANopen device is set up for CANopen device guarding. In this case the NMT master shall request the actual NMT state of the CANopen device that is checked, the NMT service CANopen device guarding shall be used. In case no confirmation is received and more than 100 ms are elapsed the process shall end with an error status. In case a confirmation is received the process shall and with the actual NMT state of the CANopen device that is checked.

9.5.3 NMT flying master start up

The definition of the flying NMT master is not in the scope of this standard.

NOTE Definitions for the flying NMT master functionality are provided in CiA 302.

---

1 Required, if bit 4 of object $1F81_h$ is set (keep alive supported).

9.5.4 Error status

The errors as shown in Table 14 shall be signaled within the NMT master during startup.

Table 14 – Error status

Error status	Description
A	The CANopen device is not listed in object $1F81_h$.
B	No response received for upload request of object 1000_h.
C	Value of object 1000_h from CANopen device is different to value in object $1F84_h$ (Device type).
D	Value of object 1018_h sub-index 01_h from CANopen device is different to value in object $1F85_h$ (Vendor ID).
E	Heartbeat event. No heartbeat message received from CANopen device.
F	Node guarding event. No confirmation for guarding request received from CANopen device.
G	Object $1F53_h$ or object $1F54_h$ not configured for that CANopen device.
H	Values of object $1F52_h$ from that CANopen device are different to value of object $1F53_h$ or value of object $1F54_h$ and bit 6 of object $1F81_h$ is not set.
I	Values of object $1F52_h$ from that CANopen device are different to value of object $1F53_h$ or value of object $1F54_h$ and download failed.
J	Configuration download failed.
K	Heartbeat event during start error control service. No heartbeat message received from CANopen device during start error control service.
L	NMT slave was initially operational. (CANopen manager may resume operation with other CANopen devices)
M	Value of object 1018_h sub-index 02_h from CANopen device is different to value in object $1F86_h$ (Product code).
N	Value of object 1018_h sub-index 03_h from CANopen device is different to value in object $1F87_h$ (Revision number).
O	Value of object 1018_h sub-index 04_h from CANopen device is different to value in object $1F88_h$ (Serial number).

9.6 Error control

9.6.1 Start error control

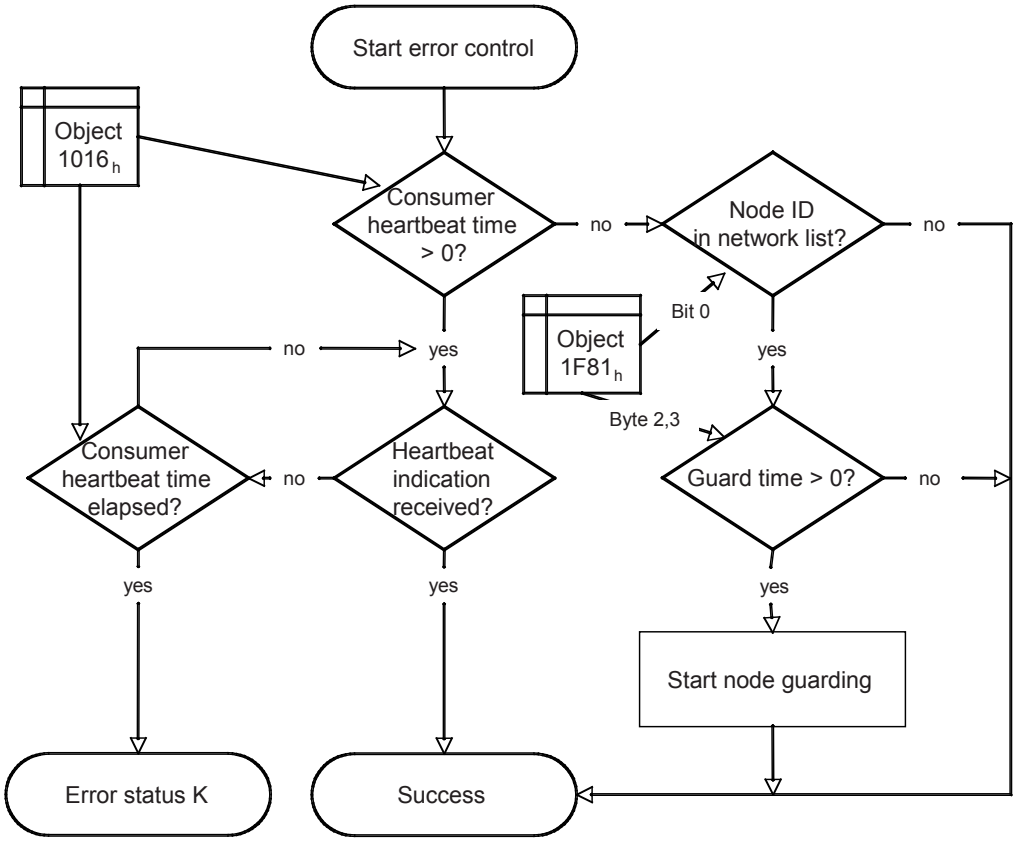


Figure 21 – Start error control

The process *start error control* as shown in Figure 21 consists of the following steps:

a) Object 1016_h (see EN 50325-4) shall be checked to decide whether the CANopen device is set up for heartbeat.

b) In case the CANopen device is set up for heartbeat and no heartbeat indication is received in time the process shall end with an error status. The timer for timeout checking shall be started immediately with the process itself.

c) In case the CANopen device is set up for heartbeat and a heartbeat indication is received in time the process shall end successfully.

d) In case the CANopen device is not set up for heartbeat bit 0 of object $1F81_h$ is used to decide whether the CANopen device shall be guarded.

e) In case the CANopen device is not in the network list the process shall end successfully.

f) In case the CANopen device is in the network list the value of byte 2 and 3 of object $1F81_h$ is used to decide whether the CANopen device shall be guarded.

g) In case the value is greater than 0 guarding shall be started for that CANopen device and the process shall end successfully.

h) In case the value equals 0 the process shall end successfully.

9.6.2 Error handler

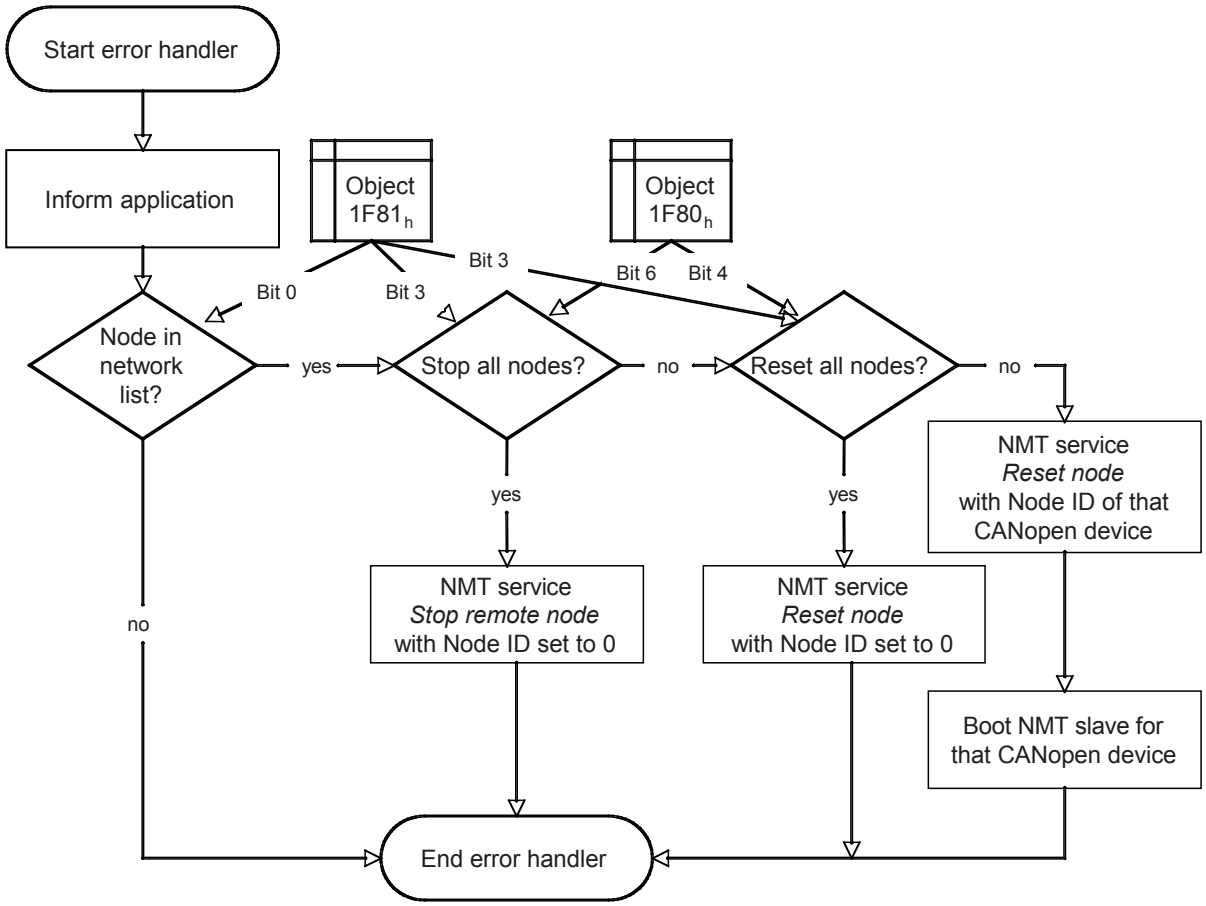


Figure 22 – Error handler

The process *error handler* as defined in Figure 22 shall be initiated any time an NMT error event occurs.

9.6.3 Bootup handler

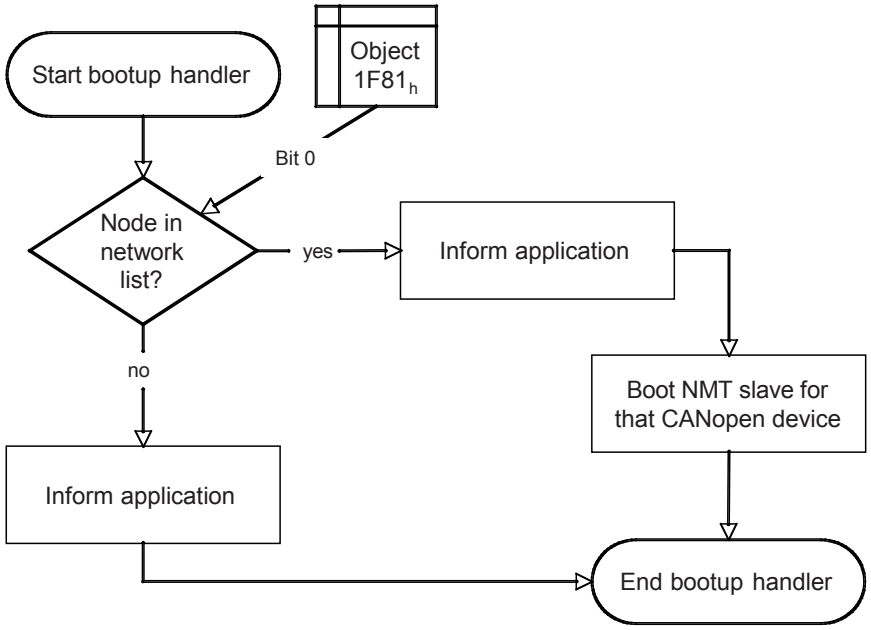


Figure 23 – Bootup handler

The process *bootup handler* as defined in Figure 23 shall be initiated any time a bootup event occurs (see EN 50325-4).

9.7 Additional NMT master services and protocols

The definition of additional NMT master services and protocols is not in the scope of this specification.

NOTE Additional NMT master services and protocols such as NMT master negotiation or NMT master detection services are provided in CiA 302.

9.8 Object dictionary entries

9.8.1 Object 1020_h: Verify configuration

This object shall indicate the downloaded configuration date and time. If a CANopen device supports the saving of parameters in non-volatile memory, a network configuration tool or a CANopen manager uses this object to verify the configuration after a CANopen device reset and to check if a reconfiguration is necessary. The configuration tool stores the date and time in that object and stores the same values in the DCF. Now the configuration tool lets the CANopen device save its configuration by writing to index 1010_h sub-index 01_h the signature "save". After a reset the CANopen device shall restore the last configuration and the signature automatically or by request. If any other command changes boot-up configuration values, the CANopen device shall reset the object Verify Configuration to 0.

The Configuration Manager compares signature and configuration with the value from the DCF and decides if a reconfiguration is necessary or not.

NOTE The usage of this object allows a significant speed-up of the boot-up process. If it is used, the system integrator considers that an user changes a configuration value and afterwards activate the command store configuration 1010_h without changing the value of 1020_h. So the system integrator ensures a 100 % consequent usage of this feature.

Sub-index 01_h (configuration date) shall contain the number of days since January 1, 1984. Sub-index 02_h (configuration time) shall be the number of ms after midnight. Table 15 and Table 16 provide the object description and the entry description.

Table 15 – Object description

Attribute	Value
Index	1020_h
Name	Verify configuration
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 16 – Entry description

Attribute	Value
Sub-index	00_h
Description	Highest sub-index supported
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	02_h
Default value	02_h
Sub-index	01_h
Description	Configuration date
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED32
Default value	Manufacturer-specific
to	
Sub-index	02_h
Description	Configuration time
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED32
Default value	Manufacturer-specific

9.8.2 Object $102A_h$: NMT inhibit time

This object shall indicate the configured inhibit time between two subsequent NMT messages. The outstanding NMT services shall be queued and shall be issued in order of their occurrence respecting the configured inhibit time. Table 17 and Table 18 define the object description and the entry description.

The value shall be given in multiples of 100 μs. The value 0 shall disable the inhibit time.

Table 17 – Object description

Attribute	Value
Index	$102A_h$
Name	NMT inhibit time
Object code	VAR
Data type	UNSIGNED16
Category	Mandatory

Table 18 – Entry description

Attribute	Value
Sub-index	00_h
Access	rw
PDO mapping	No
Value range	UNSIGNED16
Default value	0

9.8.3 Object $1F20_h$: Store DCF

This object shall be used to save the current configuration for a specific CANopen device in the network. Table 19 defines the object description and Table 20 defines the entry description.

The sub-index of an entry shall correspond to the CANopen Node-ID of the CANopen device in the network. The sub-index of the entry corresponding to its own CANopen Node-ID shall be used for self-configuration. If no DCF is saved previously a read access shall be responded with an SDO abort message (error code: 0800 0024_h or 0800 0000_h). The definition of the DCF format is not in the scope of this specification.

NOTE The format of DCF is provided in CiA 302.

Table 19 – Object description

Attribute	Value
Index	$1F20_h$
Name	Store DCF
Object code	ARRAY
Data type	DOMAIN
Category	Optional

Table 20 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	7Fh
Default value	7Fh
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	*See value definition*
Default value	Manufacturer-specific
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition
Default value	Manufacturer-specific

9.8.4 Object $1F22_h$: Concise DCF

This object shall be used to save the current configuration for a specific CANopen device in the network. Table 21 defines the object description and Table 22 defines the entry description.

The sub-index of an entry shall correspond to the CANopen Node-ID of the CANopen device in the network. The sub-index of the entry corresponding to its own CANopen Node-ID shall be used for self-configuration. The format of the Device Configuration File (DCF) shall be as defined in Figure 24. If no DCF was saved previously the number of entries shall be 0. If the DCF for a specific entry shall be deleted the number of entries shall be set to 0.

Number (n) of entries		UNSIGNED32
Entry 1	Index	UNSIGNED16
	Sub-index	UNSIGNED8
	Size (m) of data in bytes	UNSIGNED32
	Data byte 1	UNSIGNED8
	Data byte 2	UNSIGNED8
	:::::	:::::
	Data byte m	UNSIGNED8
Entry 2	Index	UNSIGNED16
	Sub-index	UNSIGNED8
	Size (m) of data bytes	UNSIGNED32
	Data byte 1	UNSIGNED8
	Data byte 2	UNSIGNED8
	:::::	:::::
	Data byte m	UNSIGNED8
:::::::::::		
Entry n	Index	UNSIGNED16
	Sub-index	UNSIGNED8
	Size (m) of data bytes	UNSIGNED32
	Data byte 1	UNSIGNED8
	Data byte 2	UNSIGNED8
	:::::	:::::
	Data byte m	UNSIGNED8

Figure 24 – Data stream definition of concise DCF

Table 21 – Object description

Attribute	Value
Index	$1F22_h$
Name	Concise DCF
Object code	ARRAY
Data type	DOMAIN
Category	Optional

Table 22 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See definition
Default value	Manufacturer-specific
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See definition
Default value	Manufacturer-specific

9.8.5 Object $1F26_h$: Expected configuration date

This object shall be used for verification of the configuration date of the CANopen devices in the network.

The configuration date (see 9.8.1 object 1020_h sub-index 01_h) of the CANopen devices in the network shall be matched against the value of this object in case the value is unequal to 0. In case the value of this object is 0 the CANopen device shall be configured.

The sub-index of an entry shall correspond to the CANopen Node-ID of the CANopen devices in the network.

Table 23 – Object description

Attribute	Value
Index	$1F26_h$
Name	Expected configuration date
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 24 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See definition
Default value	Manufacturer-specific
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See definition
Default value	Manufacturer-specific

9.8.6 Object $1F27_h$: Expected configuration time

This object shall be used for verification of the configuration time of the CANopen devices in the network.

The configuration time of the CANopen devices (object 1020_h sub-index 02_h) in the network shall be matched against the value of this object in case the value is unequal to 0. In case the value of this object is 0 the CANopen device shall be configured.

The sub-index of an entry shall correspond to the CANopen Node-ID of the CANopen devices in the network.

NOTE Object description and entry description of index 1020_h sub-index 02_h are provided in the latest CANopen version; document CiA 301.

Table 25 – Object description

Attribute	Value
Index	$1F27_h$
Name	Expected configuration time
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 26 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition
Default value	Manufacturer-specific
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See definition
Default value	Manufacturer-specific

9.8.7 Object $1F80_h$: NMT startup

This object shall configure the startup behavior of a CANopen device. Internal state transitions shall not change the value of this object. An attempt to change a bit of a functionality that is not supported by the CANopen device shall be responded with an abort message (abort code: $0800\ 0000_h$ or $0609\ 0030_h$). Figure 25 and Figure 26 define the bit-oriented structure of the value. Table 27, Table 28, Table 29, Table 30, Table 31, Table 32, and Table 33 define the allowed values. Table 34 defines exceptions for start-up capable devices. Table 35 and Table 36 define the object description and the entry description.

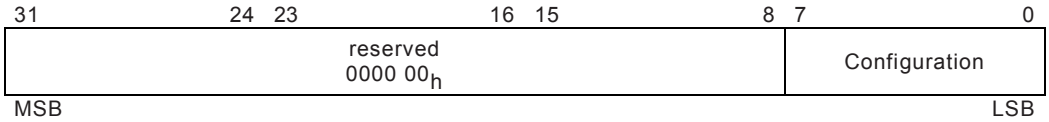


Figure 25 – Object structure

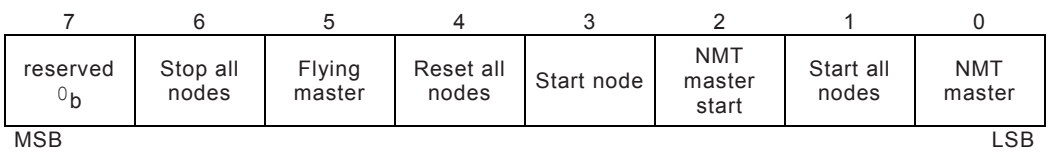


Figure 26 – Bit structure of the configuration value

Table 27 – Value NMT master (bit: 0)

Value	Description
0_b	CANopen device is not NMT master The entries of the object $1F81_h$ shall be ignored. All other bits of object $1F80_h$ shall be ignored with the exceptions defined in Table 34.
1_b	CANopen device is the NMT master

Table 28 – Value Start all nodes (bit: 1)

Value	Description
0_b	NMT service *start remote node* for each CANopen Node-ID
1_b	NMT service *start remote node* with CANopen Node-ID = 0

Table 29 – Value NMT master start (bit: 2)

Value	Description
0_b	Shall switch into NMT state *Operational* in the process NMT startup (see Figure 13)
1_b	Shall not switch into the NMT state *Operational* by itself.

Table 30 – Value Start node (bit: 3)

Value	Description
0_b	The NMT master shall start the NMT slaves.
1_b	The NMT master shall not start the NMT slaves and the application may start the NMT slaves.

Table 31 – Reset all nodes (bit: 4)

Value	Description
0_b	In case of error control event of a CANopen device defined as mandatory (*see object* $1F81_h$) the NMT service *reset node* with CANopen Node-ID of the CANopen device that caused the error control event shall be executed.
1_b	In case of error control event of a CANopen device defined as mandatory (*see object* $1F81_h$) the NMT service *reset node* with CANopen Node-ID = 0 shall be executed.

Table 32 – Flying master (bit: 5)

Value	Description
0_b	CANopen device shall not participate the NMT flying master negotiation
1_b	Reserved (for NMT flying master capable devices)

NOTE The definition of the NMT flying master functionality is not in the scope of this standard. Definitions are provided in CiA 302.

Table 33 – Stop all nodes (bit: 6)

Value	Description
0_b	In case of error control event of a CANopen device defined as mandatory (*see object* $1F81_h$) the action as defined by bit 4 shall be executed.
1_b	In case of error control event of a CANopen device defined as mandatory (*see object* $1F81_h$) the NMT service *Stop remote node* with CANopen Node-ID = 0 shall be executed. Bit 4 shall be ignored.

Table 34 – Exceptions for NMT start-up capable devices

Value	Description
00000000 00000000 00000000 00001000$_b$	NMT slave that shall enter the NMT state *Operational* after the NMT state *Initialisation* autonomously (self starting)
00000000 00000000 00000000 00000010$_b$	NMT slave that shall execute the NMT service *start remote node* with CANopen Node-ID set to 0

Table 35 – Object description

Attribute	Value
Index	1F80$_h$
Name	NMT startup
Object code	VAR
Data type	UNSIGNED32
Category	Conditional; Mandatory if the CANopen device is a CANopen manager or a start-up capable CANopen device.

Table 36 – Entry description

Attribute	Value
Sub-index	00$_h$
Access	rw
PDO mapping	No
Value range	See value definition
Default value	Manufacturer-specific

9.8.8 Object 1F81$_h$: NMT slave assignment

This object shall assign CANopen devices to the CANopen manager, the device that shall implement this object. Each sub-index of this object shall correspond to the CANopen Node-ID of the related CANopen device in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored. An attempt to change a bit of a functionality that is not supported by the CANopen device shall be responded with an abort message (abort code: 0800 0000$_h$ or 0609 0030$_h$). Figure 27 and Figure 28 define the bit-oriented structure of the value. Table 37, Table 38, Table 39, Table 40, Table 41, Table 42, and Table 43 define the value contents. Table 44 and Table 45 define the object description and the entry description.

The value for the retry factor indicates the number of retries the NMT master shall issue in case of a node guarding event. The value 0 shall disable node guarding for the CANopen device.

The value for the guard time shall indicate the cycle time for the node guarding of the CANopen device. The value shall be indicated in multiples of ms. The value 0 shall disable node guarding for the CANopen device.

NOTE If the heartbeat consumer object is configured to a value unequal to 0, then the heartbeat mechanism will have priority over node guarding.

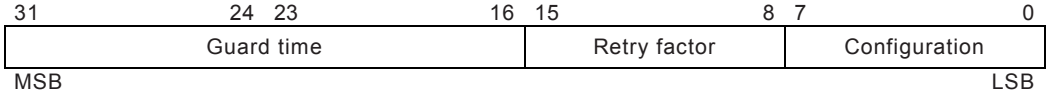


Figure 27 – Object structure of the value

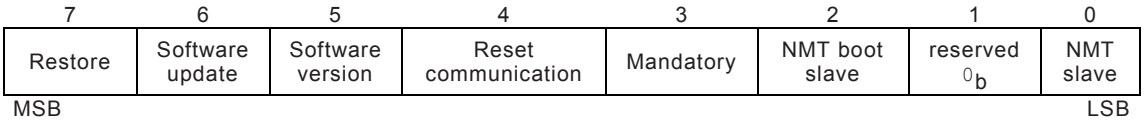


Figure 28 – Bit structure of the configuration value

Table 37 – NMT slave (bit: 0)

Value	Description
0_b	NMT master or not available in the network
1_b	NMT slave and available in the network

Table 38 – NMT boot slave (bit: 2)

Value	Description
0_b	Configuration and NMT service *Start remote node* shall not be allowed in case of error control event or NMT service *Bootup* NOTE The application is responsible for the NMT slave startup.
1_b	Configuration and NMT service *Start remote node* shall be performed in case of error control event or NMT service *Bootup*

Table 39 – Mandatory (bit: 3)

Value	Description
0_b	CANopen device may be present prior to network startup (CANopen device is optional)
1_b	CANopen device shall be present prior to network startup (CANopen device is mandatory)

Table 40 – Reset communication (bit: 4)

Value	Description
0_b	NMT service *Reset communication* may be executed for the CANopen device at any time
1_b	NMT service *Reset communication* shall not be executed for the CANopen device in case the CANopen device is in NMT state *Operational*

Table 41 – Software version (bit: 5)

Value	Description
0_b	Software version verification shall not be performed for the CANopen device
1_b	Software version verification shall be performed for the CANopen device

Table 42 – Software update (bit: 6)

Value	Description
0_b	Software update shall not be performed for the CANopen device
1_b	Software update shall be performed for the CANopen device

Table 43 – Restore (bit: 7)

Value	Description
0_b	CANopen device may be used without prior resetting.
1_b	CANopen device shall be reset to factory defaults by issuing a restore to defaults (object 1011_h)

Table 44 – Object description

Attribute	Value
Index	$1F81_h$
Name	NMT slave assignment
Object code	ARRAY
Data type	UNSIGNED32
Category	Conditional; mandatory if the CANopen device is a CANopen manager

Table 45 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	see Figure 27, Figure 28,Table 37, Table 38, Table 39, Table 40, Table 41, Table 42, and Table 43
Default value	$0000\ 0000_h$
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	see Figure 27, Figure 28,Table 37, Table 38, Table 39, Table 40, Table 41, Table 42, and Table 43
Default value	$0000\ 0000_h$

9.8.9 Object $1F82_h$: Request NMT

This object shall request a specific NMT service for a unique CANopen device in the network or for all CANopen devices in the network in case the CANopen device implementing this

object is in NMT master mode. Normally, the request is issued by another CANopen device (e.g. configuration tool) in the network or by the application on the very same CANopen device (e.g. in an IEC 61131 environment).

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. These requests may be applied for the NMT master itself. Table 46 defines the value contents. Table 47 and Table 48 define the object description and the entry description.

The values from 84_h to $8F_h$ require the knowledge of the CANopen Node-ID of the requesting CANopen device. If the CANopen Node-ID is unknown the service shall be aborted (abort code: 0800 0000_h or 0609 0030_h).

An attempt to download a value that is reserved shall be responded with an SDO abort message (abort code: 0800 0000_h or 0609 0030_h).

NOTE The values from 00_h to $7F_h$ have to be applied carefully to not unintentionally affect the NMT master or the requesting CANopen device itself.

Table 46 – Value definition

Value	Description	
	on upload (read)	**on download (write)**
00_h	NMT state unknown	reserved
01_h	CANopen device missing	reserved
02_h	reserved	
03_h	reserved	
04_h	NMT state *Stopped*	NMT service *Stop remote node*
05_h	NMT state *Operational*	NMT service *Start remote node*
06_h	reserved	NMT service *Reset node*
07_h	reserved	NMT service *Reset communication*
08_h	reserved	
:::::	:::::	
$7E_h$	reserved	
$7F_h$	NMT state *Pre-operational*	NMT service *Enter pre-operational*
80_h	reserved	
:::::	:::::	
83_h	reserved	
84_h	NMT state *Stopped*	NMT service *Stop remote node* (excluding NMT master and requesting CANopen device)
85_h	NMT state *Operational*	NMT service *Start remote node* (excluding NMT master and requesting CANopen device)
86_h	reserved	NMT service *Reset node* (excluding NMT master and requesting CANopen device)
87_h	reserved	NMT service *Reset communication* (excluding NMT master and requesting CANopen device)
88_h	reserved	
:::::	:::::	
$8E_h$	reserved	
$8F_h$	NMT state *Pre-operational*	NMT service *Enter pre-operational* (excluding NMT master and requesting CANopen device)
90_h	reserved	
:::::	:::::	
FF_h	reserved	

Table 47 – Object description

Attribute	Value
Index	$1F82_h$
Name	Request NMT
Object code	ARRAY
Data type	UNSIGNED8
Category	Optional

Table 48 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	80_h
Default value	80_h
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	see Table 46
Default value	00_h
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	see Table 46
Default value	00_h
Sub-index	80_h
Description	All nodes
Entry category	Mandatory
Access	wo
PDO mapping	No
Value range	see Table 46
Default value	00_h

9.8.10 Object $1F83_h$: Request node guarding

This object shall request node guarding for a unique CANopen device in the network or for all CANopen devices in the network in case the CANopen device implementing this object is in NMT master mode and node guarding is enabled. Normally, the request is issued by another CANopen device (e.g. configuration tool) in the network or by the application on the very same CANopen device (e.g. in an IEC 61131 environment).

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored. Table 49 defines the value contents. Table 50 and Table 51 define the object description and the entry description.

An attempt to download a value that is reserved shall be responded with an abort message (abort code: 0800 0000_h or 0609 0030_h).

NOTE Heartbeat consumer is covered by object 1016_h (see CiA301).

Table 49 – Value definition

Value	Description	
	on download (write)	**on upload (read)**
00_h	Stop node guarding	Node guarding stopped
01_h	Start node guarding	Node guarding started
02_h	reserved	
:::::	:::::	
FF_h	reserved	

Table 50 – Object description

Attribute	Value
Index	$1F83_h$
Name	Request node guarding
Object code	ARRAY
Data type	UNSIGNED8
Category	Optional

Table 51 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	80_h
Default value	80_h
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	see Table 49
Default value	00_h
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	see Table 49
Default value	00_h
to	
Sub-index	80_h
Description	All nodes
Entry category	Mandatory
Access	wo
PDO mapping	No
Value range	see Table 49
Default value	00_h

9.8.11 Object $1F84_h$: Device type identification

This object is used for verification of the device type of the CANopen devices in the network.

The device type (object 1000_h – see EN 50325-4) of the CANopen device in the network shall be matched against the value of this object in case the value is unequal to 0. An error event shall be generated if the values mismatch. In case the value of this object is 0 the device type of the CANopen device in the network may not be verified. Table 52 and Table 53 define the object description and the entry description.

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored.

Table 52 – Object description

Attribute	Value
Index	$1F84_h$
Name	Device type identification
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 53 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition of object 1000_h – *EN 50325-4*
Default value	$0000\ 0000_h$
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition in *EN 50325-4* of object 1000_h
Default value	$0000\ 0000_h$

9.8.12 Object $1F85_h$: Vendor identification

This object shall be used for verification of the Vendor ID of the CANopen devices in the network.

The Vendor ID (object 1018_h sub-index 01_h – see EN 50325-4) of the CANopen device in the network shall be matched against the value of this object in case the value is unequal to 0. An error event shall be generated if the values mismatch. In case the value of this object is 0 the Vendor ID of the CANopen device in the network may not be verified. Table 54 and Table 55 define the object description and the entry description.

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored.

Table 54 – Object description

Attribute	Value
Index	$1F85_h$
Name	Vendor identification
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 55 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition in EN 50325-4 object 1018_h sub-index 01_h
Default value	$0000\ 0000_h$
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition in EN 50325-4 object 1018_h sub-index 01_h
Default value	$0000\ 0000_h$

9.8.13 Object $1F86_h$: Product code

This object shall be used for verification of the product code of the CANopen devices in the network.

The product code (object 1018_h sub-index 02_h – see EN 50325-4) of the CANopen device in the network shall be matched against the value of this object in case the value is unequal to 0. An error event shall be generated if the values mismatch. In case the value of this object is 0 the product code of the CANopen device in the network may not be verified. Table 56 and Table 57 define the object description and the entry description.

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored.

Table 56 – Object description

Attribute	Value
Index	$1F86_h$
Name	Product code
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 57 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition of EN 50325-4 object 1018_h sub-index 02_h
Default value	$0000\ 0000_h$
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition in EN 50325-4 object 1018_h sub-index 02_h
Default value	$0000\ 0000_h$

9.8.14 Object $1F87_h$: Revision number

This object shall be used for verification of the revision number of the CANopen devices in the network.

The revision number (object 1018_h sub-index 03_h – see EN 50325-4) of the CANopen device in the network shall be matched against the value of this object in case the value is unequal to 0. An error event shall be generated if the values mismatch. A mismatch is defined as:

- the major revision number is unequal to the expected major revision number, or
- the minor revision number is less than the expected minor revision number,

In case the value of this object is 0 the revision number of the CANopen device in the network shall not be verified. Table 58 and Table 59 define the object description and the entry description.

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored.

Table 58 – Object description

Attribute	Value
Index	$1F87_h$
Name	Revision number
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 59 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition of object 1018_h sub-index 03_h – EN 50325-4
Default value	$0000\ 0000_h$
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition of object 1018_h sub-index 03_h – EN 50325-4
Default value	$0000\ 0000_h$

9.8.15 Object $1F88_h$: Serial number

This object shall be used for verification of the serial number of the CANopen devices in the network.

The serial number (object 1018_h sub-index 04_h – see EN 50325-4) of the CANopen device in the network shall be matched against the value of this object in case the value is unequal to 0. An error event shall be generated if the values mismatch. In case the value of this object is 0 the serial number of the CANopen device in the network may not be verified. Table 60 and Table 61 define the object description and the entry description.

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored.

Table 60 – Object description

Attribute	Value
Index	$1F88_h$
Name	Serial number
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 61 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition of EN 50325-4 object 1018_h sub-index 04_h
Default value	$0000\ 0000_h$
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	See value definition of EN 50325-4 object 1018_h sub-index 04_h
Default value	$0000\ 0000_h$

9.8.16 Object $1F89_h$: Boot time

The object defines the time out between start of the process *Start process boot NMT slave* and signaling of successful boot of all mandatory NMT slaves. The details are defined in 9.4.3.

The value shall be given in multiples of ms. The value 0 shall disable the timer. Table 62 provides the object description and Table 63 the entry description.

Table 62 – Object description

Attribute	Value
Index	$1F89_h$
Name	Boot time
Object code	VAR
Data type	UNSIGNED32
Category	Optional

Table 63 – Entry description

Attribute	Value
Sub-index	00_h
Access	rw
PDO mapping	No
Value range	UNSIGNED32
Default value	$0000\ 0000_h$

9.8.17 Object $1F8A_h$: Restore configuration

This object is used to define the allowed restore procedure for a CANopen device during startup.

The restore procedure configured in this object shall be used to restore, during startup, each CANopen device in the network (object 1011_h – see EN 50325-4) according to the object entries. Table 64 and Table 65 define the object description and the entry description.

The sub-index shall correspond to the CANopen Node-ID of the CANopen devices in the network. The sub-index corresponding to its own CANopen Node-ID shall be ignored.

Table 64 – Object description

Attribute	Value
Index	$1F8A_h$
Name	Restore configuration
Object code	ARRAY
Data type	UNSIGNED8
Category	Optional

Table 65 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	$7F_h$
Default value	$7F_h$
Sub-index	01_h
Description	CANopen Node-ID 1
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	01_h to $7F_h$
Default value	01_h
to	
Sub-index	$7F_h$
Description	CANopen Node-ID 127
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	01_h to $7F_h$
Default value	01_h

9.8.18 Object $1F91_h$: Self-starting nodes timing parameters

This object defines parameters that shall be configured to apply a determined behavior for self-starting CANopen devices. Table 66 and Table 67 define the object description and entry description.

The NMT master detection timeout, NMT master request delay time and node time slot shall be given in multiples of ms.

Table 66 – Object description

Attribute	Value
Index	$1F91_h$
Name	Self-starting nodes timing parameters
Object code	ARRAY
Data type	UNSIGNED16
Category	Optional; Mandatory for CANopen devices supporting self-starting

Table 67 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	03_h
Default value	03_h
Sub-index	01_h
Description	NMT master detection timeout
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED16
Default value	100
Sub-index	02_h
Description	NMT master request delay time
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED16
Default value	500
Sub-index	03_h
Description	Node time slot
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED 16
Default value	15

10 Gateway functions

10.1 Content

This clause provides the gateway functions for gateways between a Train Backbone and a CANopen-based consist network.

Network access services are specified to enable the access to CANopen-based consist networks. In addition a mapping of these CANopen access services to an ASCII protocol is defined. In order to fit into a TCN architecture, the mapping of this protocol to management messages is defined in Clause 11.

Depending on the implemented capabilities, this clause introduces also several gateway classes.

10.2 Gateway architecture

The architecture of a gateway between a Train backbone and a CANopen consist network is shown in Figure 29. In the illustration the Train backbone is implemented as WTB.

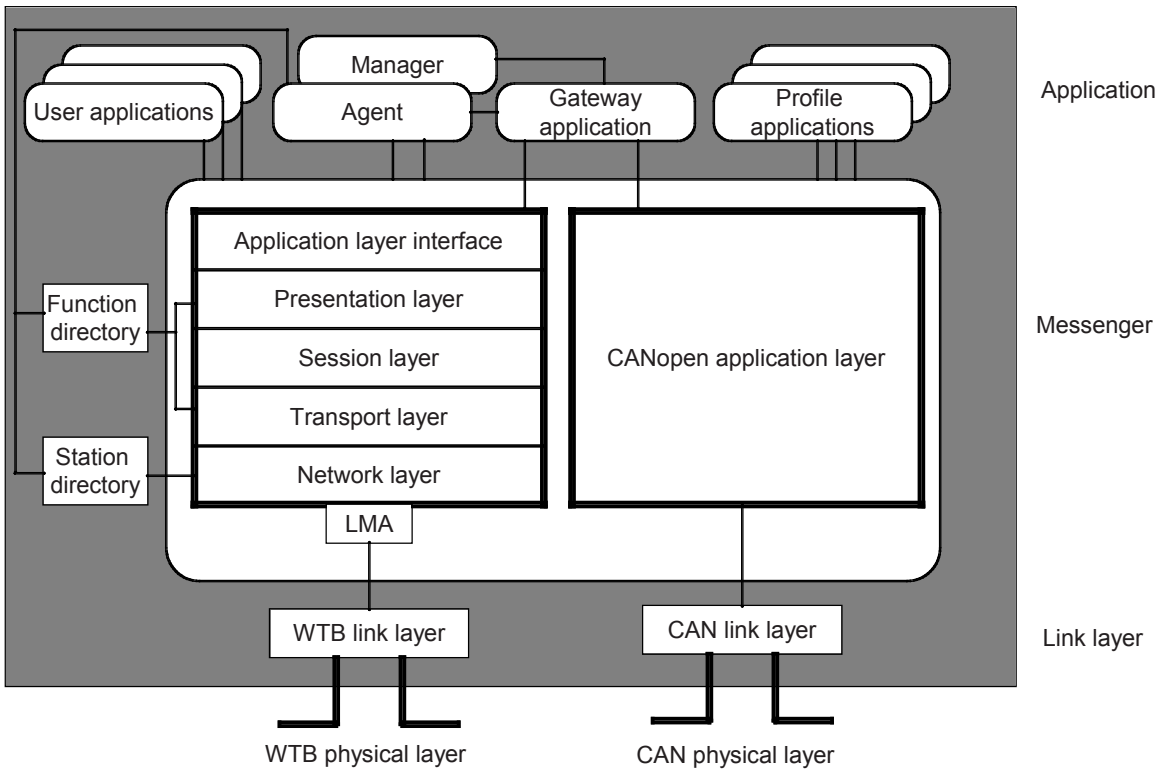


Figure 29 – Gateway between Train backbone and CANopen consist network

A gateway interprets between a Train Backbone and a CANopen-based consist network. It transfers the process data via the gateway application and ensures the correct endian style for the data representation on both sides of the gateway.

NOTE The ordering scheme for transferring data in CANopen is little endian in contrast to WTB, which uses big endian style.

By means of TNM management messages TNM services may be requested from both sides of the gateway. In addition to the services specified in IEC 61375-2-1, further TNM services are defined (see clause 11.5) that enable full control of the sub-layered CANopen consist network (see network access services as defined in 10.4). The network access services provided by a gateway - between the Train Backbone and a CANopen-based consist network - give one Node, which is connected to the Train Backbone, access to End Devices connected to a CANopen-based consist network.

As a mapping of TNM management messages to CANopen communication objects (COBs) is defined, the TNM management services are available within CANopen-based consist networks as well.

As shown in Figure 29, the Manager functionality as well as the Agent functionality reside on application level, as defined in IEC 61375-2-1. In addition manufacturer-specific user applications as well as standardized applications according to CANopen profiles (profile applications) may reside on this level.

10.3 General principles and services

10.3.1 Content

This subclausse provides the network access services provided by a gateway device that enables to access from the Train Backbone the CANopen-based consist network.

10.3.2 Gateway class definitions

A CANopen device, in which the gateway functionality to the Train Backbone resides, may support one or more of the following classes:

Class 0: The gateway is a device, acting as network slave (NMT slave functionality) within the CANopen network. The device supports only the process data transfer capability. A gateway of this class is not capable to communicate TNM managment messages via the CANopen network.

Class 1: The gateway is a device, acting as network slave (NMT slave functionality) within the CANopen network. The device shall provide, in addition to the functionality defined for gateways of Class 0, the SDO client functionality.

Class 2: The gateway is a device implementing the functionality of a Class 1 device, which additionally implements SDO requesting device (SRD) functionality.

Class 3: The gateway is a device within the CANopen network acting as the CANopen manager. The CANopen manager functionality shall be achived by supporting the NMT master functionality as well as the SDO manager functionality.

NOTE Only gateway devices of class 1, 2 or 3 are capable to communicate TNM management messages within CANopen-based consist networks.

10.3.3 Service primitives definitions

Via service primitives the services are utilized by which the gateway application and the network application layer interact. The service primitives are defined in IEC 61375-1.

10.4 Network access service specification

10.4.1 SDO access services

10.4.1.1 Content

The services specified in this clause are used to initiate and configure SDO services accessing any object in the object dictionary of any node on any of the CANopen networks linked to the gateway device.

10.4.1.2 Upload SDO

This service shall initiate an SDO upload service. The parameters for this service are defined in Table 68.

Table 68 – Upload SDO service

Parameter	Indication	Response
Argument	**Mandatory**	
Network	Optional	
Node-ID	Optional	
Multiplexor	Mandatory	
Data type	Optional	
BOOLEAN	Selection	
UNSIGNED8	Selection	
UNSIGNED16	Selection	
UNSIGNED24	Selection	
UNSIGNED32	Selection	
UNSIGNED40	Selection	
UNSIGNED48	Selection	
UNSIGNED56	Selection	
UNSIGNED64	Selection	
INTEGER8	Selection	
INTEGER16	Selection	
INTEGER24	Selection	
INTEGER32	Selection	
INTEGER40	Selection	
INTEGER48	Selection	
INTEGER56	Selection	
INTEGER64	Selection	
REAL32	Selection	
REAL64	Selection	
TIME_OF_DAY	Selection	
TIME_DIFFERENCE	Selection	
OCTET_STRING	Selection	
VISIBLE_STRING	Selection	
UNICODE_STRING	Selection	
DOMAIN	Selection	
Offset	Optional	
Length	Optional	
Remote result		**Mandatory**
Success		Selection
Data		Mandatory
Length		Optional
Failure		Selection
Reason		Mandatory

10.4.1.3 Download SDO

This service shall initiate an SDO download service. The parameters for this service are defined in Table 69.

Table 69 – Download SDO parameters

Parameter	*Indication*	*Response*
Argument	**Mandatory**	
Network	Optional	
Node-ID	Optional	
Multiplexor	Mandatory	
Data type	Optional	
BOOLEAN	Selection	
UNSIGNED8	Selection	
UNSIGNED16	Selection	
UNSIGNED24	Selection	
UNSIGNED32	Selection	
UNSIGNED40	Selection	
UNSIGNED48	Selection	
UNSIGNED56	Selection	
UNSIGNED64	Selection	
INTEGER8	Selection	
INTEGER16	Selection	
INTEGER24	Selection	
INTEGER32	Selection	
INTEGER40	Selection	
INTEGER48	Selection	
INTEGER56	Selection	
INTEGER64	Selection	
REAL32	Selection	
REAL64	Selection	
TIME_OF_DAY	Selection	
TIME_DIFFERENCE	Selection	
OCTET_STRING	Selection	
VISIBLE_STRING	Selection	
UNICODE_STRING	Selection	
DOMAIN	Selection	
Offset	Optional	
Length	Optional	
Data	Mandatory	
Remote result		**Mandatory**
Success		Selection
Failure		Selection
Reason		Mandatory

10.4.1.4 Configure SDO timeout

This service shall configure the time-out for all Client-SDOs on the gateway device. The parameters for this service are defined in Table 70.

Table 70 – Configure SDO timeout parameters

Parameter	*Indication*	*Response*
Argument	**Mandatory**	
Network	Optional	
SDO Timeout	Mandatory	
Remote result		**Mandatory**
Success		Selection
Failure		Selection
Reason		Mandatory

10.4.2 PDO access services

10.4.2.1 Content

The services specified in this clause are used to configure and initiate PDO services in the gateway device. They include:

- Configure a receive PDO

- Configure a transmit PDO
- Request to read a PDO
- Request to write a PDO
- Indicate a received PDO

The two PDO configuration services are intended to create PDOs in the gateway device. If the gateway device implements an object dictionary, the PDO communication and mapping parameter entries shall be set-up accordingly.

The two PDO request services are intended to control the PDOs in accordance to the configured PDO transmission type.

The data types VISIBLE_STRING, OCTET_STRING, and UNICODE_STRING, as well as DOMAIN shall not be used as PAS data type parameter.

NOTE The PAS services are not intended to configure the PDO communication and mapping entries of the object dictionary of the remote nodes. Accessing the CANopen nodes individually by means of SAS services may do the PDO configuration.

10.4.2.2 Configure RPDO

This service shall create an RPDO in the gateway device. Table 71 defines the parameters for this service.

Table 71 – Configure RPDO service parameters

Parameter	*Indication*	*Response*
Argument Network PDO number COB-ID TxType Nbr_objects 1st mapped object Data type Multiplexor 2nd mapped object Data type Multiplexor … 64th mapped object Data type Multiplexor	**Mandatory** Optional Mandatory Mandatory Mandatory Mandatory Mandatory Selection Selection Optional Selection Selection … Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.2.3 Configure TPDO

This service shall create a TPDO in the gateway device. Table 72 defines the parameters for this service.

Table 72 – Configure TPDO service parameters

Parameter	*Indication*	*Response*
Argument Network PDO number COB-ID TxType Nbr_objects 1st mapped object Data type Multiplexor 2nd mapped object Data type Multiplexor … 64th mapped object Data type Multiplexor	**Mandatory** Optional Mandatory Mandatory Mandatory Mandatory Mandatory Selection Selection Optional Selection Selection … Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.2.4 Read PDO data

This service shall read the data received by an RPDO. If an RPDO is configured with transmission type 252 or 253, the gateway devices shall trigger it by means of an RTR.

NOTE It is not recommended to use RTR.

The received data shall be transmitted in the remote result. Table 73 defines the parameters for this service.

Table 73 – Read PDO data service parameters

Parameter	*Indication*	*Response*
Argument Network PDO number	**Mandatory** Optional Mandatory	
Remote result Success Network PDO number Nbr_objects Data 1st object … Data 64th object Failure Reason		**Mandatory** Selection Optional Mandatory Mandatory Conditional … Conditional Selection Mandatory

10.4.2.5 Write PDO data

This service shall trigger the transmission of a PDO. The actual transmission of the PDO shall be triggered according to the configured PDO transmission type.

The parameters of this service are defined in Table 74.

Table 74 – Write PDO data service parameters

Parameter	*Indication*	*Response*
Argument Network PDO number Nbr_objects Data 1st object … Data 64th object	**Mandatory** Optional Mandatory Mandatory Conditional … Conditional	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.2.6 RPDO received

This service shall signal that new PDO data has been received and shall provide the received data.

The parameters of this service are defined in Table 75.

Table 75 – RPDO received service parameters

Parameter	*Request*
Argument Network PDO number Nbr_objects 1st object value … 64th object value	**Mandatory** Optional Mandatory Mandatory Conditional … Conditional

10.4.3 CANopen NMT services

10.4.3.1 Content

The services specified in this clause are used to control a CANopen device or a CANopen network and associated error control services.

10.4.3.2 Start node

This service shall set CANopen nodes into NMT state OPERATIONAL. For Class 1 and Class 2 devices this service shall trigger a CANopen Request NMT service. For Class 3 devices the service shall trigger a Start Remote Node service. For Class 0 devices this shall trigger a failure notification as remote result. Table 76 defines the parameters for this service.

Table 76 – Start node service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID All	**Mandatory** Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.3 Stop node

This service shall set CANopen nodes into NMT state STOPPED. For class 1 and class 2 devices this service shall trigger a CANopen request NMT service. For class 3 devices the service shall trigger a stop remote node service. For Class 0 devices this shall trigger a failure notification as remote result. Table 77 defines the parameters for this service.

NOTE The remote result for class 1 and 2 devices is only the confirmation of the SDO request; for class 3 devices the remote result is based on the error control services as defined in EN 50325-4.

Table 77 – Stop node service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID All	**Mandatory** Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.3.1 Set node to pre-operational

This service shall set CANopen nodes into NMT state PRE-OPERATIONAL. For class 1 and class 2 devices this service shall trigger a CANopen request NMT service. For class 3 devices the service shall trigger an Enter Pre-operational service. For Class 0 devices this shall trigger a failure notification as remote result. Table 78 defines the parameters for this service.

Table 78 – Set node to pre-operational service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID All	**Mandatory** Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.4 Reset node

This service shall set CANopen nodes into NMT state RESET APPLICATION. For class 1 and class 2 devices this service shall trigger a CANopen request NMT service. For class 3 devices the service shall trigger a reset node service. For Class 0 devices this shall trigger a failure notification as remote result. Table 79 defines the parameters for this service.

Table 79 – Reset node service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID All	**Mandatory** Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.5 Reset communication

This service shall set CANopen nodes into NMT state RESET COMMUNICATION. For class 1 and class 2 devices this service shall trigger a CANopen request NMT service. For class 3 devices the service shall trigger a reset communication service. For Class 0 devices this shall trigger a failure notification as remote result. Table 80 defines the parameters for this service.

Table 80 – Reset communication service parameters

Parameter	Indication	Response
Argument Network Node-ID All	**Mandatory** Optional Selection Selection	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.6 Enable node guarding

This service is only available for class 3 devices. It shall start node guarding for the device specified by CANopen Node-ID with the parameters given by GuardTime and LifeTimeFactor. If heartbeat is already activated on the addressed node, the service request shall be rejected. Table 81 defines the parameters for this service.

Table 81 – Enable node guarding service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID GuardTime LifeTimefactor	**Mandatory** Optional Mandatory Mandatory Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.7 Disable node guarding

This service is only available for class 3 devices. It shall stop node guarding for the device specified by CANopen Node-ID. Table 82 defines the parameters for this service.

Table 82 – Disable node guarding service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID	**Mandatory** Optional Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.8 Start heartbeat consumer

This service shall start the consumption of heartbeat messages transmitted by a CANopen device specified by CANopen Node-ID. Table 83 defines parameters for this service.

Table 83 – Start heartbeat consumer service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID HeartbeatConsumerTime	**Mandatory** Optional Mandatory Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.9 Disable heartbeat consumer

This service shall stop the consumption of heartbeat messages transmitted by a CANopen device specified by CANopen Node-ID. Table 84 defines parameters for this service.

Table 84 – Disable heartbeat consumer service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID	**Mandatory** Optional Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.3.10 Error control event received

This service shall signal the NMT status or error control events received from a CANopen node specified by CANopen Node-ID. Table 85 defines the parameters for this service.

Table 85 – Error control event received parameters

Parameter	*Request*
Argument Network Node-ID Status Error Code	**Mandatory** Optional Mandatory Selection Selection

10.4.4 Device failure management services

10.4.4.1 General

The services specified in this clause are used to manage failures within the gateway device or within any other CANopen device.

10.4.4.2 Read device error

This service shall read EMCY message information received from the CANopen device specified by the CANopen Node-ID parameter. Table 86 defines the parameters for this service.

Table 86 – Read device error service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID	**Mandatory** Optional Optional	
Remote result Success Network Node-ID Error Error Msg number Error Msg text Emergency Emergency code Error register Manufacturer error Failure Reason		**Mandatory** Selection Optional Mandatory Selection Mandatory Optional Selection Mandatory Optional Optional Selection Mandatory

10.4.4.3 Emergency event received

This service shall signal the reception of an emergency message in the gateway device transmitted by a CANopen device specified by the CANopen Node-ID. Table 87 defines the parameters for this service.

Table 87 – Emergency event received service parameters

Parameter	*Request*
Argument Network Node-ID Emergency code Error register Manufacturer error	**Mandatory** Optional Mandatory Mandatory Mandatory Optional

10.4.5 CANopen interface configuration services

10.4.5.1 General

The services described in this clause are used to configure and parameterize the CANopen interface of the gateway device.

10.4.5.2 Initialize gateway

This service shall initiate the CANopen initialization of the gateway device. It shall perform a power-on equivalent reset of the CANopen interface. It is used to initialize the bit-timing parameters. Table 88 defines the parameters for this service.

Table 88 – Initialize gateway service parameters

Parameter	*Indication*	*Response*
Argument Network CAN bit timing	**Mandatory** Optional Optional	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.5.3 Store configuration

This service shall command the gateway device to store its CANopen interface configuration. Table 89 defines the parameters for this service.

Table 89 – Store configuration service parameters

Parameter	*Indication*	*Response*
Argument Network Storage specifier	**Mandatory** Optional Optional	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.5.4 Restore configuration

This service shall command the gateway device to restore its CANopen interface configuration. Table 90 defines the parameters for this service.

Table 90 – Restore configuration service parameters

Parameter	*Indication*	*Response*
Argument Network Storage specifier	**Mandatory** Optional Optional	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.5.5 Set heartbeat producer

This service shall set the CANopen heartbeat producer time in the gateway device. Table 91 defines the parameters for this service.

Table 91 – Set heartbeat producer service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID HeartbeatProducerTime	**Mandatory** Optional Mandatory Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.5.6 Set node-ID

This service shall set the CANopen Node-ID in the gateway device for the CANopen network given in the *network* parameter. Table 92 defines the parameters for this service.

Table 92 – Set node-ID service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID	**Mandatory** Optional Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.5.7 Start emergency consumer

This service shall start the consumption of emergency messages. The relation between CANopen Node-ID producing an emergency message and the COB-ID has to be explicitly known. Table 93 defines the parameters for this service.

Table 93 – Start emergency consumer service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID COB-ID	**Mandatory** Optional Mandatory Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.5.8 Stop emergency consumer

This service shall stop the consumption of emergency messages. The relation between CANopen Node-ID producing an emergency message and the COB-ID has to be explicitly known. Table 94 defines the parameters for this service.

Table 94 – Stop emergency consumer service parameters

Parameter	*Indication*	*Response*
Argument Network Node-ID COB-ID	**Mandatory** Optional Mandatory Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.6 Gateway management services

10.4.6.1 General

The services specified in this clause are used to manage the gateway device.

10.4.6.2 Set default network

This service shall set the default network number, which shall be used for all services. Table 95 defines the parameters for this service.

Table 95 – Set default network service parameters

Parameter	*Indication*	*Response*
Argument DefaultNetwork	**Mandatory** Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.6.3 Set default node-ID

This service shall set the default CANopen Node-ID, which shall be used for all services. Table 96 defines the parameters for this service.

Table 96 – Start default node-ID service parameters

Parameter	*Indication*	*Response*
Argument Default node-ID	**Mandatory** Mandatory	
Remote result Success Failure Reason		**Mandatory** Selection Selection Mandatory

10.4.6.4 Get version

This service shall get information on the gateway device and its CANopen interface. Table 97 defines the parameters for this service.

Table 97 – Get version service parameters

Parameter	*Indication*	*Response*
Argument Network	**Mandatory** Optional	
Remote Result Success Vendor-ID Product code Revision number Serial number Gateway class Protocol version Implementation class Failure Reason		**Mandatory** Selection Mandatory Mandatory Mandatory Mandatory Mandatory Mandatory Mandatory Selection Mandatory

10.4.7 Manufacturer-specific services

The manufacturer of a gateway device may define additional services.

10.5 ASCII mapping of network access services

10.5.1 Content

This clause defines the mapping of the network access services to an ASCII-based communication syntax for CANopen gateway devices. This protocol is mapped to TCN message data.

10.5.2 Definitions

10.5.2.1 Command

A Command controls the gateway and interacts with CANopen devices. It may have a long form and a short form. The short form is a one or two letter abbreviation of the long form. The long form is obtained by concatenating the short form and the string enclosed in brackets “[“, “]”.

NOTE In the given examples it is assumed that network address and node address are preset.

10.5.2.2 Data type syntax

The mandatory data types, provided in Table 98, shall be supported.

Table 98 – Syntax and CANopen data types

Syntax	CANopen Type	Category
b	Boolean	Mandatory
u8	UNSIGNED8	Mandatory
u16	UNSIGNED16	Mandatory
u24	UNSIGNED24	Optional
u32	UNSIGNED32	Mandatory
u40	UNSIGNED40	Optional
u48	UNSIGNED48	Optional
u56	UNSIGNED56	Optional
u64	UNSIGNED64	Optional
i8	INTEGER8	Mandatory
i16	INTEGER16	Mandatory
i24	INTEGER24	Optional
i32	INTEGER32	Mandatory
i40	INTEGER40	Optional
i48	INTEGER48	Optional
i56	INTEGER56	Optional
i64	INTEGER64	Optional
r32	REAL32	Optional
r64	REAL64	Optional
t	Time of day (with two arguments: day ms)	Optional
td	Time difference	Optional
vs	Visible string	Optional
os	Octet string	Optional
us	Unicode string	Optional
d	Domain	Optional

NOTE The value of the data type *domain*, *octet string*, and *unicode string* is encoded in mime-base64 as described in RFC 2045.

All wrapping CRLF shall be stripped from the encoded data to have one long string.

10.5.2.3 Whitespace

Whitespaces as specified in ISO/IEC 9899 except of CR and LF.

NOTE A visible string with whitespace is enclosed with double quotes to denote it as single argument of the command. If a double quote is used within the string, the quotes are escaped by a second quotes, e.g. “Hello ““World””, CANopen is great”.

10.5.2.4 Command structure

10.5.2.4.1 General

The principle communication is based on non-case-sensitive ASCII strings according to ISO/IEC 646 instead of architecture and CPU/compiler depending binary structures. Due to this, no application handles with things like endianness, data size and byte alignment. In all cases where numbers are used, the typical representation is as specified in ISO/IEC 9899.

100	- decimal, starting with a number
0x64	- hexadecimal, starting with the string 0x
1.22	- float
.22e10	- float
22e3	- float

10.5.2.4.2 Request

The CANopen gateway is controlled by commands. A command is composed of tokens, which are separated by any number of whitespaces and is closed with a CRLF.

All commands are confirmed. Commands start with a sequence number which is enclosed by square brackets [ ]. The sequence number is a 4-byte value. It is not used for event-triggered messages. According to the addressing principle, a network number and a node number follow the sequence number. Network number and node number are optional, when the CANopen gateway only provides one Train Backbone to CANopen Consist Network interface or when a client presets them. Commands that affect only the server not a remote node but a net and node are given, net and node are ignored. In BNF notation the command notation is provided in Table 99:

Table 99 – Command notation in BNF

```
<command-request>      ::= "["<sequence>"]" [[<net>] <node>] <command>
<sequence>             ::= UNSIGNED32
<net>                  ::= UNSIGNED8
<node>                 ::= UNSIGNED8
<command>              ::= <command-specifier> | <compound-command>
<compound-command>     ::= <command-specifier> <parameter>
<parameter>            ::= <value> | <compound-parameter>
<compound-parameter>   ::= <value> <parameter>
<map-object>           ::= <datatype> | <multiplexor>
<datatype>             ::= 'b' | 'u8' | 'u16' | 'u32' | 'u40' | 'u48' |
                           'u56' | 'u64' | 'i8' | 'i16' | 'i24' | 'i32' |
                           'i40' | 'i48' | 'i56' | 'i64' | 'r32' | 'r64' |
                           't' | 'td' | 'vs' | 'os' | 'us' | 'd'
<multiplexor>          ::= <index> <subindex>
<index>                ::= UNSIGNED16
<subindex>             ::= UNSIGNED8
```

Net numbers are starting with 1. Node numbers are starting with 1. The value 0 for net or node is used to address all networks or all nodes.

The token <value> designates a value of the possible CANopen data types. Within the description of the commands the sequence number is omitted for reasons of readability.

10.5.2.4.3 Response

The CANopen gateway shall respond with the same sequence number at the first position as given by the request. This number shall be given in decimal format. There shall be only one response to a request. The response notation is provided in Table 100.

Table 100 – Response notation

```
<command-response>     ::= "["<sequence>"]" <response>
<response>             ::= <value> | <error-string> | <emcy-list> |
                       "OK"
<error-string>         ::= "Error:" <error code>
<error-code>           ::= <internal-error-code> | <sdo-abort-code>
<internal-error-code>  ::= see table 2
<sdo-abort-code>       ::= UNSIGNED32
<emcy-list>            ::= [<emcy1> " "..<emcy254>]
```

The sdo-abort-codes (SAC) are defined in EN 50325-4. Allowed internal-error-codes (InEC) are listed in Table 101:

Table 101 – Internal error code (InEC)

InEC	Message text
100	Request not supported
101	Syntax error
102	Request not processed due to internal state
103	Time-out (where applicable)
200	Lost guarding message
201	Lost connection
202	Heartbeat started
203	Heartbeat lost
205	Boot-up
300	Error passive
301	Bus off
303	CAN buffer overflow
304	CAN init
305	CAN active (at init or start-up)
400	PDO already used
401	PDO length exceeded

NOTE After bus-off, the command init should be invoked to reset the CAN controller.

10.5.2.4.4 Event triggered messages

Messages due to errors in the CANopen network or the occurrence of communication objects using the producer-consumer principle shall not use a sequence number. The notation for event-triggered messages is provided in Table 102.

Table 102 – Notation for event triggered messages

```
<event-trigged-message>::= [[net] node] <event-specifier> <parameter>
<event-specifier>      ::= "EMCY" | "ERROR" | "PDO" | "SYNC" | "USER"
```

The content of event-triggered messages is described within the command description that enables the specific service.

10.5.3 Network access command specification

10.5.3.1 SDO access commands

10.5.3.1.1 General

The following command definitions shall be used to implement the SDO access services as defined in 10.4.

SDO access services are addressing a specific object at an SDO server via index and sub-index and a transfer data type.

10.5.3.1.2 Upload SDO command

Indication syntax is defined in Table 103.

Table 103 – Syntax for upload SDO command

```
[[net] node] r[ead] <multiplexor> <datatype>
```

Examples are provided in Table 104.

Table 104 – Examples for upload SDO command

```
[21] r 0x1000 0 u32
[4096] read 0x1008 0 vs
```

Response syntax:

See 10.5.2.4.3.

10.5.3.1.3 Download SDO command

Indication syntax is defined in Table 105.

Table 105 – Syntax for Download SDO command

```
[[net] node] w[rite] <multiplexor> <datatype> <value>
```

Examples are provided in Table 106.

Table 106 – Examples for download SDO command

```
[20] 1 23 w 0x1016 0 u16 100
[23] write 0x1016 0 u16 0x64
```

Response syntax:

See 10.5.2.4.3.

10.5.3.1.4 Configure SDO timeout command

The timeout delay time for abort error code ‘SDO protocol timed out’, used by the gateway’s SDO client, may be set.

Indication syntax is defined in Table 107.

Table 107 – Syntax for configure SDO timeout command

```
[net] set sdo_timeout <ms>
```

Response syntax:

See chapter 10.5.2.4.3.

10.5.3.2 PDO access commands

10.5.3.2.1 General

The following command definitions shall be used to implement the PDO access services as defined in 10.4. Normally a PDO is first configured before transmission and reception is possible.

A PDO is seen from the view of the gateway. An RPDO therefore receives data from the CANopen network and a TPDO sends the data into the CANopen network.

NOTE In order to disable/delete a gateway PDO, the bit 31 in the COB-ID is used. For details refer to EN 50325-4.

10.5.3.2.2 Configure RPDO command

Indication syntax is defined in Table 108.

Table 108 – Syntax for configure RPDO command

```
[[net] node] set rpdo <nr> <COB> <tx-type> <nr-of-data> <map-obj1>
[..<map-obj64>]
<tx-type>::= "rtr" | "event" | "sync<0..240>"
```

NOTE In case a <map-obj> is given in form of index and sub-index, e.g. <multiplexor>, they are counted as 1 in the <nr-of-data>. The <nr> is an offset; it starts with 1.

Examples are provided in Table 109.

Table 109 – Examples for configure RPDO command

```
[12] set rpdo 1 0x180 event 3 u8 u8 u16
[24] 2 set rpdo 1 0x180 event 3 u8 u8 i16
```

Response syntax:

See 10.5.2.4.3.

10.5.3.2.3 Configure TPDO command

Indication syntax is provided in Table 110.

Table 110 – Syntax for configure TPDO command

```
[[net] node] set tpdo <nr> <COB> <tx-type> <nr-of-data> <map-obj1>
[..<map-obj64>]
```

NOTE In case a <map-obj> is given in form of index and subindex, i.e. <multiplexor>, they are counted as 1 in the <nr-of-data>. The <nr> is an offset; it starts with 1.

Example is provided in Table 111.

Table 111 – Examples for configue TPDO command

```
[13] set tpdo 1 0x201 rtr 4 u8 u16 u16 u8
```

Response syntax:

See 10.5.2.4.3.

NOTE It is not recommended to support RTR. Therefore it is recommended to set bit 30 of the COB-ID accordingly. For details see EN 50325-4.

10.5.3.2.4 Read PDO data command

Indication syntax is defined in Table 112.

Table 112 – Syntax for read PDO data command

```
[net] r[ead] p[do] <nr>
```

If the transmission type is different than RTR, the gateway answers with the values of the mapped objects.

Response syntax is defined in Table 113.

Table 113 – Response syntax for read PDO data command

```
[net] pdo <nr> <nr-of-data> <value1>[..<value64>]
```

10.5.3.2.5 Write PDO data command

Indication syntax is defined in Table 114.

Table 114 – Syntax for write PDO data command

```
[net] w[rite] p[do] <nr> <nr-of-data> <value1>[..<value64>]
```

Response syntax:

See 10.5.2.4.3.

10.5.3.2.6 RPDO received command

Indication syntax is defined in Table 115.

Table 115 – Syntax for RPDO receive command

```
[net] pdo <nr> <nr-of-data> <value1>[..<value64>]
```

Examples are provided in Table 116.

Table 116 – Examples RPDO received command

```
1 pdo 1 2 123 4        ;# gateway with more than one network
                       ;# received RPDO1 at net 1 with two objects
                       ;# mapped
pdo 2 1 1234           ;# RPDO2 with one object mapped
pdo 2 3 100 2 4        ;# three objects mapped
```

10.5.3.3 CANopen NMT commands

10.5.3.3.1 General

The following command definitions shall be used to implement the CANopen NMT services as defined in 10.4. The supported services depend on the gateway class.

10.5.3.3.2 Start node command

Indication syntax is defined in Table 117.

Table 117 – Syntax for start node command

```
[[net] node] start
```

Response syntax:

See 10.5.2.4.3.

10.5.3.3.3 Stop node command

Indication syntax is defined in Table 118.

Table 118 – Syntax for stop node command

[[net] node] stop

Response syntax:

See 10.5.2.4.3.

10.5.3.3.4 Set node to pre-operational command

Indication syntax is defined in Table 119.

Table 119 – Syntax set node to pre-operational command

[[net] node] preop[erational]

Response syntax:

See 10.5.2.4.3.

10.5.3.3.5 Reset node command

Indication syntax is defined in Table 120.

Table 120 – Syntax reset node command

[[net] node] reset node

Response syntax:

See 10.5.2.4.3.

10.5.3.3.6 Reset communication command

Indication syntax is defined in Table 121.

Table 121 – Syntax reset communication command

[[net] node] reset comm[unication]

Response syntax:

See 10.5.2.4.3.

10.5.3.3.7 Enable node guarding command

Activating node-guarding functionality enables another event-triggered response message to the clients of the gateway. Only in the case that one of the monitored CANopen nodes violates the guarding protocol, an event message shall be sent to the clients.

Indication syntax is defined in Table 122.

Table 122 – Syntax enable node guarding command

[[net] node] enable guarding <guardingtime> <lifetimefactor>

Response syntax:

See 10.5.2.4.3.

10.5.3.3.8 Disable node guarding command

Indication syntax is defined in Table 123.

Table 123 – Syntax disable node guarding command

[[net] node] disable guarding

Response syntax:

See 10.5.2.4.3.

10.5.3.3.9 Start heartbeat consumer command

Activating the heartbeat consumer at the gateway enables another event-triggered response message to the gateway's clients. Only in the case that one of the monitored CANopen nodes violates the guarding protocol, an event message shall be sent to the clients.

Indication syntax is defined in Table 124.

Table 124 – Syntax start heartbeat consumer command

[[net] node] enable heartbeat <heartbeattime>

Response syntax:

See 10.5.2.4.3.

10.5.3.3.10 Disable heartbeat consumer command

Indication syntax is defined in Table 125.

Table 125 – Syntax disable heartbeat consumer command

[[net] node] disable heartbeat

Response syntax:

See 10.5.2.4.3.

10.5.3.3.11 Error control event received command

Response syntax is defined in Table 126.

Table 126 – Syntax for error control event received command

[[net] node] ERROR <internal-error-code>

10.5.3.4 Device failure management commands

10.5.3.4.1 General

The following command definitions shall be used to implement the device failure management services as defined in 10.4.4.

10.5.3.4.2 Read device error command

Indication syntax is defined in Table 127.

Table 127 – Syntax for read device error command

[[net] node] r[ead] error

Response syntax:

See 10.5.2.4.3.

10.5.3.4.3 Emergency event received command

Response syntax is defined in Table 128.

Table 128 – Syntax for emergency event received command

[[net] node] EMCY <emcy-code> <error-register> <m-error-code>

The (manufacturer)-error-code shall be returned as five decimal values corresponding to the manufacturer-specific error code in the EMCY message.

10.5.3.5 CANopen interface configuration commands

10.5.3.5.1 General

The following command definitions shall be used to implement the CANopen interface configuration services as defined in 10.4.5. Settings are valid for the default network, if no network address is given. The node address shall be omitted, in case it is given.

10.5.3.5.2 Initialize gateway command

Indication syntax is defined in Table 129.

Table 129 – Syntax for initialize gateway command

[net] init <bitrate>

The bit rate shall be given as table index of the standard CANopen bit rate table specified in Table 130.

Table 130 – Bit rate indices

Index	Bit rate
0	1 Mbit/s
1	800 kbit/s
2	500 kbit/s
3	250 kbit/s
4	125 kbit/s
5	reserved
6	50 kbit/s
7	20 kbit/s
8	10 kbit/s
9	Automatic bit rate detection

Response syntax:

See 10.5.2.4.3.

10.5.3.5.3 Store configuration command

All settings may be stored. Storage of settings may be selective to a special service. If no argument is given all settings shall be stored.

Indication syntax is defined in Table 131. The storage identifier is specified in Table 132.

Table 131 – Syntax for store configuration command

```
[net] store <storage-specifier>
```

Table 132 – Storage specifier

Storage specifier	Description
CFG	id, bitrate, default node, default net
PDO	PDO-number, transmission type, number of mapping, mapping
SDO	sdo timeout
NMT	- Node guarding: node, guarding time, life time factor
	- Heartbeat: node, heartbeat time
	- Heartbeat time of the server

Response syntax:

See 10.5.2.4.3.

10.5.3.5.4 Restore configuration command

Indication syntax is defined in Table 133.

Table 133 – Syntax restore configuration command

```
[net] restore <storage-specifier>
```

Response syntax:

See chapter 10.5.2.4.3.

10.5.3.5.5 Set heartbeat producer command

Indication syntax is defined in Table 134.

Table 134 – Syntax set heartbeat producer command

```
[net] set heartbeat <ms>
```

Response syntax:

See 10.5.2.4.3.

10.5.3.5.6 Set node-ID command

Indication syntax is defined in Table 135.

Table 135 – Syntax set node-ID command

```
[net] set id <value>
```

Response syntax:

See 10.5.2.4.3.

10.5.3.5.7 Start emergency consumer command

This command is not specified.

10.5.3.5.8 Stop emergency consumer command

This command is not specified.

10.5.3.6 Gateway management commands

10.5.3.6.1 General

The following command definitions shall be used to implement the gateway management services as defined in 10.4.6.

10.5.3.6.2 Set default network command

Indication syntax is defined in Table 136.

Table 136 – Syntax set default network command

`[net] set network <value>`

Response syntax:

See 10.5.2.4.3.

10.5.3.6.3 Set default node-ID command

Indication syntax is defined in Table 137.

Table 137 – Syntax set default node-ID command

`[net] set node <value>`

Response syntax:

See chapter 10.5.2.4.3.

10.5.3.6.4 Get version command

Indication syntax is defined in Table 138.

Table 138 – Syntax for get version command

`info version`

The response to the command is the current version information of the gateway as a whitespace separated list. The first list elements are values normally contained in the gateway's object dictionary at object 1018_h. The following listed elements contain the gateway class, as a combination of possible classes and the version number of the implemented protocol corresponded to 10.4.6.4.

Response syntax is defined in Table 139.

Table 139 – Response syntax for get version command

`<version-string>::= <vendor-id> <product-code>`
`<version-high>.<version-low> <serial-number>`
`<gateway-class> <protocol-version>`
`<implementation-class>`

Result example is provided in Table 140.

Table 140 – Example for get version response

[1234] 52 100 1.01 1234567 128 0.85 0.10

10.5.3.7 Manufacturer-specific commands

The following command definitions shall be used to implement the manufacturer-specific services. Gateway manufacturer may add commands to their gateway to provide further features. In order to avoid syntax errors due to an unknown command all user-specific commands shall be prepended with an underscore "_".

If there is the need for event-triggered messages not specified in this part of the specification the event specifier "USER" shall be used.

11 Train network management

11.1 Content

This clause extends the set of train network management (TNM) services for a WTB Node specified in IEC 61375-2-1. Train network management specifies services to assist commissioning, testing, operation and maintenance of a TCN. These services comprise station identification and control, distribution of routing and topography, remote reading and forcing variables, downloading and uploading, and management of the train network as well as of the consist network's link layers. A Manager process requests these services remotely and each station through an Agent process executes them.

For informative reasons, the handling of management messages is illustrated in Figure 30, as defined in IEC 61375-2-1.

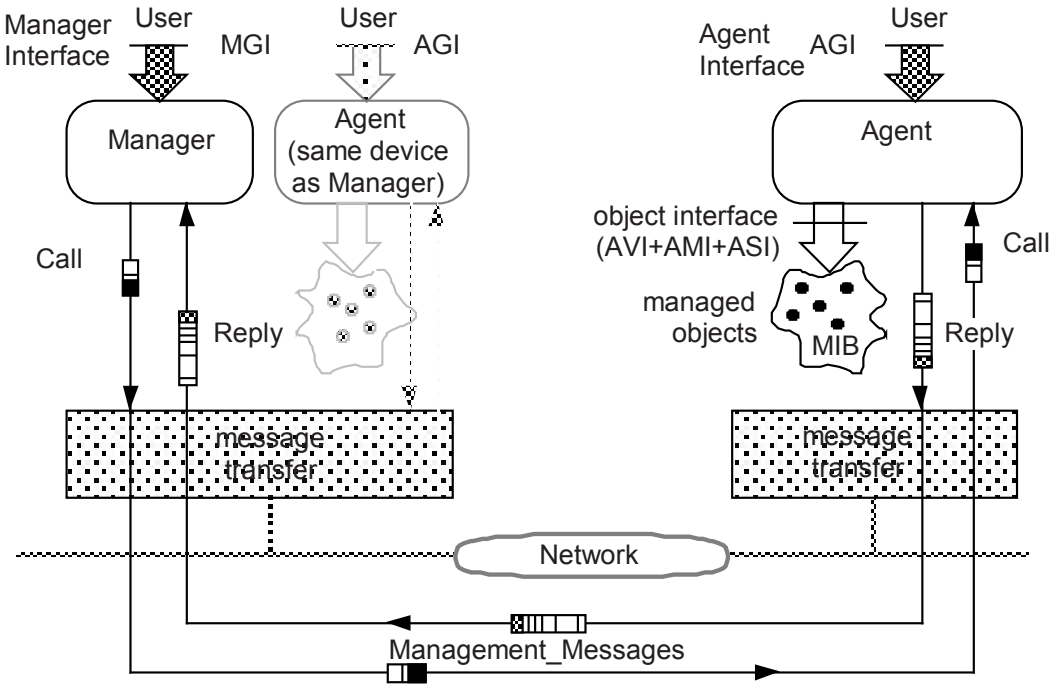


Figure 30 – Management messages (informative)

The additional services are specific for Stations connected to a CANopen consist network. A Manager shall request these services by sending a management Call_Massage to the Agent in the target Station. The Agent shall execute the service and shall respond by means of a management Reply_Message with the result of the service. The CANopen-related services, which may be requested at the Agent, are specified in 10.4.

11.2 Manager, Agents and interfaces (informative)

As defined in IEC 61375-2-1 the Train Network Management services are provided in each Station by an Agent. The Station_Id of the Station on which it resides identifies the Agent. A Manager requests the Train Network Management services.

11.3 Management message protocol (informative)

For the purpose of Train Network Management, Manager and Agents communicate over the network by exchanging management messages, using the Messages Services of the Train Communication Network, as illustrated in Figure 30 and defined in IEC 61375-2-1. The Manager acts as a Caller and the Agent as a Replier.

The Manager accesses a remote object in two steps; the Manager sends a management Call_Message and the Agent decodes the message, accesses the actual object and sends back a management Reply_Message, with the result of the service.

11.4 Object interfaces (informative)

According to IEC 61375-2-1, TCN Communication objects are related to the network communication, while TCN non-communication objects are related to other properties of a Station.

NOTE1 Examples are provided in IEC 61375-2-1.

The Agent accesses communication objects through the interfaces defined for general access in this standard, and in particular through the:

a) AVI (Application_Variables_Interface) for Variables;

b) AMI (Application_Messages_Interface) for Messages;

c) ASI (Application_Supervisory_Interface) for objects not accessible by user processes.

The ASI provides access to objects located in the different communication layers. The Agent accesses these objects through the Layer Management Interface of the layer in which they reside.

NOTE 2 The entity, which effectively accesses the objects in each layer, is called the Layer Management Entity, or LME.

The Agent shall also be able to access non-communication objects. The interface for doing this is not specified.

11.5 CANopen-specific management services

11.5.1 General

CANopen-specific management messages identify a service by two identifiers, the SIF_code and the Command_code. The SIF_code determines one of two groups of services – services that are executed with and without the reservation by the Manager for its exclusive use. The Command_code selects the particular service of the group. The payload of the call message is the request for a service coded as an ASCII string, the payload of the reply message is the response coded in the same way. The ASCII coding is specified in 10.5.

11.5.2 Agent interfaces on a Station connected to CANopen consist network

The definitions for the Agent interfaces, provided in IEC 61375-2-1 apply for CANopen devices, that communicate TCN message data, as well. The Agent interfaces are illustrated in Figure 31 for a gateway between the Train backbone and CANopen-based consist network. In Figure 31 the Train Backbone is implemented as WTB.

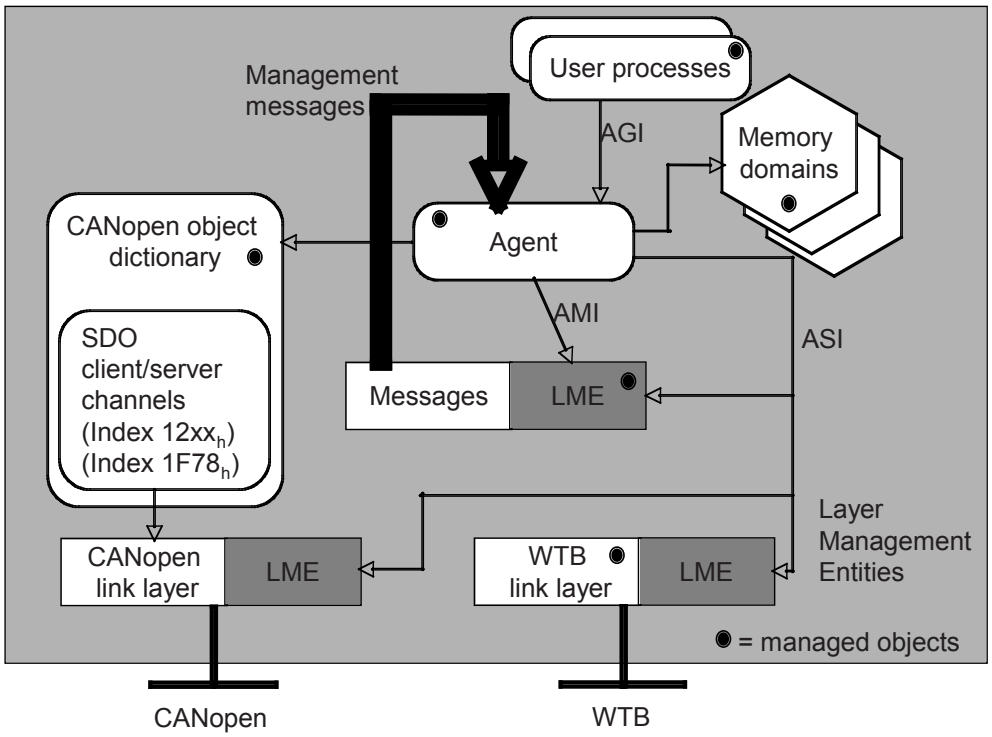


Figure 31 – Agent interface on a CANopen (gateway) station for message data

On a CANopen station the AVI is not used. The application data is directly represented in dedicated objects in the manufacturer specifc as well as in the standardized CANopen object dictionary index range, as described in 8.2.

11.5.3 Management message structure for CANopen consist networks

Each service shall be invoked by a Call/Reply Management message exchange with the format defined in Table 141.

Table 141 – Management message structure

```
Management_Message_Can::= RECORD
  {
  tnm_key   ENUM8 --    first octet
    {
    CALL     ('02'H),         --    Call (request)
    REPLY    ('82'H)          --    Reply (response)
},
  message   ONE_OF [tnm_key] --    selects call or reply
    {
    [CALL] Call_Mgt_Message_Can,  --    see 11.5.5
    [REPLY]Reply_Mgt_Message_Can  --    see 11.5.6

    }
  }
```

The most significant bit of 'tnm_key' shall indicate if this is a Call or a Reply message.

11.5.4 Notation for the CANopen specific SIF_codes

In management message services, the SIF_code indicates the requested services. In case of CANopen consist networks, the SIF_code indicates the group of requested CANopen service. As defined in Table 142, the least significant bit indicates if this is a read or a write (modifying) group of services. Read services may be used without previous reservation.

Table 142 – CANopen specific SIF_codes

```
Sif_Code_Can::= ENUM8  --    choice of the group of services
  {
  WRITE_CANOPEN_COMMAND              (91),
  READ_CANOPEN_COMMAND               (90)
  }
```

11.5.5 Notation for a call CANopen management message

The notation for a call CANopen management message is defined in Table 143.

Table 143 – Notation for a call CANopen management message

```
Call_Mgt_Message_Can::=     RECORD
  {
  sif_code  Sif_Code_Can,    --    the second octet is the
              sif_code
  message_body   ONE_OF [sif_code]
    {
    [WRITE_CANOPEN_COMMAND]       Call_Write_CANopen_Command,
    [READ_CANOPEN_COMMAND]        Call_Read_CANopen_Command
    }
  }
```

11.5.6 Notation for a reply CANopen management message

The notation for a reply CANopen management message is defined in Table 144.

Table 144 – Notation for a reply CANopen management message

```
Reply_Mgt_Message_Can::= RECORD
  {
  sif_code  Sif_Code_Can,    --    the second octet is the
              sif_code
  message_body   ONE_OF [sif_code]
    {
    [WRITE_CANOPEN_COMMAND]       Reply_Write_CANopen_Command,
    [READ_CANOPEN_COMMAND]        Reply_Read_CANopen_Command
    }
  }
```

11.5.7 Notation for the TNM CANopen services command codes

The Command code determines the particular service of the group of services identified by SIF_code. Table 145 and Table 146 define the TNM CANopen services command codes for services that require reservation or not.

Table 145 – TNM CANopen services command codes (reservation required)

```
Cmd_Code_R::= ENUM8    -- choice of the service that needs reservation
  {
  UPLOAD_SDO                    (1),
  DOWNLOAD_SDO                  (2),
  CONFIGURE_SDO_TIMEOUT         (3),
  CONFIGURE_RPDO                (4),
  CONFIGURE_TPDO                (5),
  WRITE_PDO_DATA                (6),
  START_NODE                    (7),
  STOP_NODE                     (8),
  RESET_NODE                    (9),
  RESET_COMMUNICATION           (10),
  ENABLE_NODE_GUARDING          (11),
  DISABLE_NODE_GUARDING         (12),
  START_HEARTBEAT_CONSUMER      (13),
  DISABLE_HEARTBEAT_CONSUMER    (14),
  INITIALIZE_GATEWAY            (15),
  STORE_CONFIGURATION           (16),
  RESTORE_CONFIGURATION         (17),
  SET_HEARTBEAT_PRODUCER        (18),
  SET_NODE_ID                   (19),
  SET_DEFAULT_NETWORK           (20),
  SET_DEFAULT_NODE              (21)
  }
```

NOTE Notification services are not considered, as they cannot be implemented (request, response).

Table 146 – TNM CANopen services command codes (reservation not required)

```
Cmd_Code_NR::= ENUM8   -- choice of the service that does not need
        reservation
  {
  READ_PDO_DATA                 (30),
  READ_DEVICE_ERROR             (31),
  GET_VERSION                   (32)
  }
```

11.6 TNM CANopen services

11.6.1 Content

This subclause defines the messages, identified by command code and specified by service command string transferred in the body of the message, by means of which the related TNM CANopen service shall be requested and responded.

11.6.2 Call_Write_CANopen_Command (with reservation)

By this message the services of the group “with reservation” (see Table 145) identified by the command code and specified by the service command string transferred in the body of the message shall be requested. Figure 32 provides the command structure and Table 147 provides the value definition.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tnm_key								sif_code = 91							
reserved1								cmd_code							
reserved2															
string_size															
command: ARRAY ALIGN16 [string_size] OF															
(CHARACTER8)								CHARACTER8 or '00'H							

Figure 32 – Call_Write_CANopen_Command

Table 147 – Value definition for Call_Write_CANopen_Command

```
Call_Write_CANopen_Command::= RECORD
  {
  reserved1 WORD8 (=0),       --    reserved
  cmd_code  Cmd_Code_R,       --    command code of the service
  reserved2 WORD16 (=0),      --    reserved
  string_size    UNSIGNED16,--    up to 65535 characters
  command   ARRAY ALIGN16 [string_size] OF CHARACTER8 -- service
            command  string
  }
```

11.6.3 Reply_Write_CANopen_Command (with reservation)

By this message the Call_Write_CANopen command of the group "with reservation" (see 11.6.2) identified by the command code and specified by the service command string transferred in the body of the message shall be replied. Figure 33 provides the command structure and Table 148 provides the value definition.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tnm_key								sif_code = 91							
reserved1								get_sif_code							
reserved2															
string_size															
response: ARRAY ALIGN16 [string_size] OF															
(CHARACTER8)								CHARACTER8 or '00'H							

Figure 33 – Reply_Write_CANopen_Command

Table 148 – Value definition Reply_Write_CANopen_Command

```
Reply_Write_CANopen_Command::= RECORD
  {
  reserved1 WORD8 (=0),       --    reserved
  cmd_code  Cmd_Code_R,       --    command code of the service
  reserved2 WORD16 (=0),      --    reserved
  string_size    UNSIGNED16,--    up to 65535 characters
  response  ARRAY ALIGN16 [string_size] OF CHARACTER8    --    service
            response string
  }
```

11.6.4 Call_Read_CANopen_Command (without reservation)

By this message the services of the group "without reservation" (see Table 146) identified by the command code and specified by the service command string transferred in the body of the message shall be requested. Figure 34 provides the command structure and Table 149 provides the value definition.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tnm_key								sif_code = 90							
reserved1								cmd_code							
reserved2															
string_size															
command: ARRAY ALIGN16 [string_size] OF															
(CHARACTER8)								CHARACTER8 or '00'H							

Figure 34 – Call_Read_CANopen_Command (without reservation)

Table 149 – Value definition for Call_Read_CANopen_Command (without reservation)

```
Call_Read_CANopen_Command::= RECORD
  {
  reserved1 WORD8 (=0),       --    reserved
  cmd_code  Cmd_Code_NR,      --    command code of the service
  reserved2 WORD16 (=0),      --    reserved
  string_size    UNSIGNED16,--    up to 65535 characters
  command   ARRAY ALIGN16 [string_size] OF CHARACTER8    --    service
            command string
  }
```

11.6.5 Reply_Read_CANopen_Command (without reservation)

By this message the Call_Write_CANopen command of the group "without reservation" (see 11.6.4) identified by the command code and specified by the service command string transferred in the body of the message shall be replied. Figure 35 provides the command structure and Table 150 provides the value definition.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tnm_key								sif_code = 90							
reserved1								get_sif_code							
reserved2															
string_size															
response: ARRAY ALIGN16 [string_size] OF															
(CHARACTER8)								CHARACTER8 or '00'H							

Figure 35 – Reply_Read_CANopen_command (without reservation)

Table 150 – Value definition for Reply_Read_CANopen_Command (without reservation)

```
Reply_Read_CANopen_Command::= RECORD
  {
  reserved1 WORD8 (=0),       --    reserved
  cmd_code  Cmd_Code_NR,      --    command code of the service
  reserved2 WORD16 (=0),      --    reserved
  string_size    UNSIGNED16,--    up to 65535 characters
  response  ARRAY ALIGN16 [string_size] OF CHARACTER8    --    service
            response string
  }
```

12 CANopen management message data handling

12.1 General

CANopen management massages may be also communicated between CANopen devices within a CANopen-based consist network. Figure 36 shows a CANopen device that is capable to communicate message data via the CANopen based consist network.

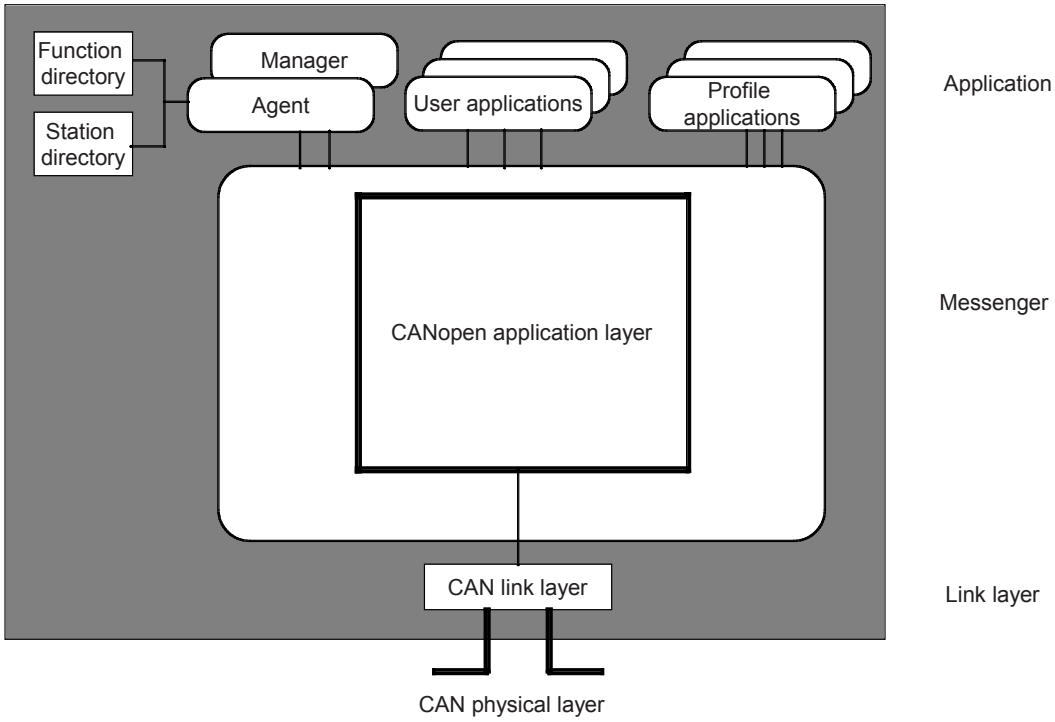


Figure 36 – CANopen device capable to handle TNM management messages

In additon to user- and CANopen profile-specific applications the Manager- and Agent-application reside in such a CANopen device. In addition the Function- and Station-directory are supported within the CANopen device's application. For transmission and reception of message data, the CANopen device, as shown in Figure 36, supports SDO server and client channels and the CANopen object dictionary entry $1F78_h$.

As defined and described in IEC 61375-2-1, Message Data is transmitted as datagrams. A datagram is similar to a letter: each datagram carries all addresses needed to route it from end to end (and send the acknowledgements back). This scheme is advantageous when several busses are interconnected, since the routers do not need to keep knowledge about previous messages.

Each Message Data frame carries two types of addresses: the source and destination Device Addresses for the communication within one bus (Device Addresses), and the addresses of the origin source and final destination (Network Addresses).

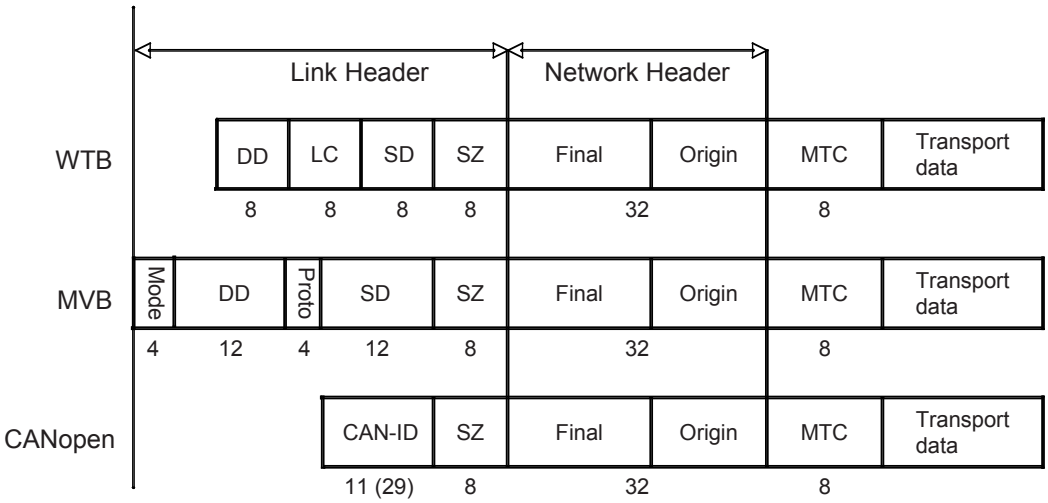
The source and destination Device Addresses apply only within the same bus. When Message Data is sent to another consist network, the destination device is the Node at the Train Backbone, which acts as Gateway. When the Node receives Message Data from another consist vehicle, it inserts its Device Address as source device.

Message Data carry two types of addresses:

- a source address and a destination address, which identify the devices communicating over the same bus and which are bus-specific. These addresses are specific to the Link Layer;
- an Origin Address and a Final Address, which identify the Stations communicating over the network and which are known in the whole network. The Origin Address and the Final Address identify the Producer and the Consumer, and in turn the Caller and the Replier. These addresses belong to the Network Layer.

12.2 Message data format

As defined in IEC 61375-2-1, with regard to Message Data, the frames on the WTB, the MVB, CANopen or any other bus system, shall only differ in their Link Header as shown in Figure 37.



DD = destination Device Address
SD = source Device Address
LC = Link Control
MTC Message Transport Control
SZ = Link Data Size
Proto = Protocol Type

CAN-ID = CAN Identifier; determined by selected CANopen SDO channel
SDO channel = Communication channel between DD and SD

Figure 37 – Message data format comparison

With regard to CANopen, the Link header comprises the CAN-Identifier. As message data shall be communicated via SDO – a confirmed point-to-point communication - the destination as well as the source device address is implicitly determined by the selection of the SDO communication channel. Each SDO communication channel provides, by means of the SDO parameter set, an unique CAN-Identifier for the request as well as for the response direction.

12.3 Requirements for message data communication within CANopen networks

Any CANopen device, connected to a CANopen-based consist network, which shall communicate Message Data, shall support SDO client as well as SDO server functionality.

Each CANopen device that shall call services at another device within the CANopen consist network, shall support a SDO client channel to call a service. By means of an SDO write access to object $1F78_h$ (see clause 12.4), via the SDO server channel of the “calling CANopen device”, the CANopen device that executes and replies to received message data shall transfer the reply.

Each CANopen device that shall receive a request of a service from another CANopen device within the CANopen consist network, shall support a SDO server channel to receive the request of the service. By means of an SDO write access via its SDO client channel this CANopen device shall transmit the response to the “calling CANopen device”.

NOTE It is recommended to support as many SDO client and -server channels in such a CANopen device, as device are in the CANopen network, that are capable to communicate message data. To each of these devices a pair of client/server channels is pre-configured.

The Message Data may be transferred either by SDO segmented transfer or SDO block transfer as defined in EN 50325-4. The message data shall be received within the appropriate sub-index of CANopen object dictionary index $1F78_h$ (see 12.4). As each sub-index of this object is related to one SDO server channel, the sub-index to be written shall be equal to the number of the utilized SDO server channel.

12.4 Object $1F78_h$: CANopen message data reception

TCN Message Data shall be received via SDO write accesses to this object as defined in 12.3. Each sub-index of this object shall be related to a supported SDO server channel.

The data written to this object is of type domain and shall be interpreted as defined in IEC 61375-2-1 TCN message data.

Object and entry description are provided in Table 151 and Table 152.

Table 151 – Object description

Attribute	Value
Index	$1F78_h$
Name	CANopen message data reception
Object code	ARRAY
Data type	DOMAIN
Category	Conditional; mandatory if message data communication is supported

Table 152 – Entry description

Attribute	Value
Sub-index	00_h
Description	Number of entries
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	01_h to 80_h
Default value	Device-specific
Sub-index	01_h
Description	TCN message data received via SDO server channel 1 specified by Index 1200h
Entry category	Mandatory
Access	wo
PDO mapping	No
Value range	See Clause 11 and IEC 61375-2-1
Default value	Manufacturer specific

Attribute	Value
Sub-index	02_h
Description	TCN message data received via SDO server channel 2 specified by Index 1201h
Entry category	Conditional; mandatory if message data communication is supported via SDO server channel 2
Access	wo
PDO mapping	No
Value range	See Clause 11 and IEC 61375-2-1
Default value	Manufacturer specific
to	
Sub-index	80_h
Description	TCN message data received via SDO server channel 128 specified by Index 127Fh
Entry category	Conditional; mandatory if message data communication is supported via SDO server channel 128
Access	wo
PDO mapping	No
Value range	See Clause 11 and IEC 61375-2-1
Default value	Manufacturer specific

13 Conformance testing

The relevant controller-device interface standards shall specify the type tests required to verify compliance of the design of a device to this standard. The equipment to be tested shall include

- power supply,
- network device,
- controller,
- communication medium,
- electromechanics.

These tests shall include electrical tests, electromagnetic compatibility tests, and logical tests. For gateway devices between the Train Backbone and the CANopen-based consist network, all communication interfaces shall be tested.

The conformance test plan for WTB is provided in IEC 61375-2-2. The conformance test plan for CANopen devices is not in the scope of this standard.

NOTE 1 The CANopen conformance test plan is specified in the document CiA 310.

NOTE 2 Information on conformance testing services is offered at CAN in Automation (CiA) GmbH.

Bibliography

IEC 61375-2-4, *Electronic railway equipment – Train Communication Network (TCN) – Part 2-4: Application profile*

CiA 301, *CANopen application layer and communication profile. CAN in Automation e.V., Nuremberg*

CiA 302, *CANopen additional application layer functions. CAN in Automation e.V., Nuremberg*

CiA 305, *CANopen layer setting services (LSS) and protocols. CAN in Automation e.V., Nuremberg*

CiA 303-1, *CANopen additional specification – Part 1: Cabling and connector pin assignment. CAN in Automation e.V., Nuremberg*

CiA 310, *CANopen conformance test plan. CAN in Automation e.V., Nuremberg*

CiA 421, *CANopen application profile for train vehicle control networks. CAN in Automation e.V., Nuremberg*

RFC 2045; *RFC 2045 Multipurpose Internet mail extensions*. www.rfc.net

UIC 556, *Information transmission in trains (train bus). International Union of Railways (UIC), Brussels*

_____

This page deliberately left blank

This page deliberately left blank

British Standards Institution (BSI)

BSI is the national body responsible for preparing British Standards and other standards-related publications, information and services.

BSI is incorporated by Royal Charter. British Standards and other standardization products are published by BSI Standards Limited.

About us

We bring together business, industry, government, consumers, innovators and others to shape their combined experience and expertise into standards -based solutions.

The knowledge embodied in our standards has been carefully assembled in a dependable format and refined through our open consultation process. Organizations of all sizes and across all sectors choose standards to help them achieve their goals.

Information on standards

We can provide you with the knowledge that your organization needs to succeed. Find out more about British Standards by visiting our website at bsigroup.com/standards or contacting our Customer Services team or Knowledge Centre.

Buying standards

You can buy and download PDF versions of BSI publications, including British and adopted European and international standards, through our website at bsigroup.com/shop, where hard copies can also be purchased.

If you need international and foreign standards from other Standards Development Organizations, hard copies can be ordered from our Customer Services team.

Subscriptions

Our range of subscription services are designed to make using standards easier for you. For further information on our subscription products go to bsigroup.com/subscriptions.

With **British Standards Online (BSOL)** you'll have instant access to over 55,000 British and adopted European and international standards from your desktop. It's available 24/7 and is refreshed daily so you'll always be up to date.

You can keep in touch with standards developments and receive substantial discounts on the purchase price of standards, both in single copy and subscription format, by becoming a **BSI Subscribing Member**.

PLUS is an updating service exclusive to BSI Subscribing Members. You will automatically receive the latest hard copy of your standards when they're revised or replaced.

To find out more about becoming a BSI Subscribing Member and the benefits of membership, please visit bsigroup.com/shop.

With a **Multi-User Network Licence (MUNL)** you are able to host standards publications on your intranet. Licences can cover as few or as many users as you wish. With updates supplied as soon as they're available, you can be sure your documentation is current. For further information, email bsmusales@bsigroup.com.

BSI Group Headquarters

389 Chiswick High Road London W4 4AL UK

Revisions

Our British Standards and other publications are updated by amendment or revision.

We continually improve the quality of our products and services to benefit your business. If you find an inaccuracy or ambiguity within a British Standard or other BSI publication please inform the Knowledge Centre.

Copyright

Useful Contacts:

Customer Services
Tel: +44 845 086 9001
Email (orders): orders@bsigroup.com
Email (enquiries): cservices@bsigroup.com

Subscriptions
Tel: +44 845 086 9001
Email: subscriptions@bsigroup.com

Knowledge Centre
Tel: +44 20 8996 7004
Email: knowledgecentre@bsigroup.com

Copyright & Licensing
Tel: +44 20 8996 7070
Email: copyright@bsigroup.com

bsi.

...making excellence a habit.™