INTERNATIONAL STANDARD

ISO 15745-3

First edition
2003-11-15

Industrial automation systems and integration — Open systems application integration framework —

Part 3: Reference description for IEC 61158-based control systems

Systèmes d'automatisation industrielle et intégration — Cadres d'intégration d'application pour les systèmes ouverts —

Partie 3: Description de référence pour les systèmes de contrôle fondés sur la CEI 61158

Reference number
ISO 15745-3:2003(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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

ISO 15745-3 was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration,* Subcommittee SC 5, *Architecture, communications and integration frameworks.*

ISO 15745 consists of the following parts, under the general title *Industrial automation systems and integration — Open systems application integration framework*:

— *Part 1: Generic reference description*

— *Part 2: Reference description for ISO 11898-based control systems*

— *Part 3: Reference description for IEC 61158-based control systems*

— *Part 4: Reference description for Ethernet-based control systems*

Introduction

The application integration framework (AIF) described in ISO 15745 defines elements and rules that facilitate:

— the systematic organization and representation of the application integration requirements using integration models;

— the development of interface specifications in the form of application interoperability profiles (AIPs) that enable both the selection of suitable resources and the documentation of the "as built" application.

ISO 15745-1 defines the generic elements and rules for describing integration models and AIPs, together with their component profiles - process profiles, information exchange profiles, and resource profiles. The context of ISO 15745 and a structural overview of the constituents of an AIP are given in Figure 1 of ISO 15745-1:2003.

This part of ISO 15745 extends the generic AIF described in ISO 15745-1 by defining the technology specific elements and rules for describing both communication network profiles and the communication related aspects of device profiles specific to control systems based on IEC 61158 (P-NET®[1], PROFIBUS[2], WorldFIP®[3], ControlNet™[4], and INTERBUS®[5]). These technologies use profiles of IEC 61158 which are specified in IEC 61784-1. Profiles for ISO/IEC 8802-3-based control systems are outside the scope of this part of ISO 15745 and are specified in ISO 15745-4.

In particular, this part of ISO 15745 describes technology specific profile templates for the device profile and the communication network profile. Within an AIP, a device profile instance or a communication network profile instance is part of the resource profile defined in ISO 15745-1. The device profile and the communication network profile XML instance files are included in a resource profile XML instance using the ProfileHandle_DataType as specified in ISO 15745-1:2003, 7.2.5.

AIFs specified using the elements and rules of ISO 15745-1 can be easily integrated with the component profiles defined using the elements and rules specified in this part.

---

[1] P-NET is the registered trademark of the International P-NET User Organisation Aps (IPUO). Control of trademark use is given to the non profit organisation IPUO. This information is given for the convenience of users of this International Standard and does not constitute an endorsement by ISO of the trademark holder or any of its products. Compliance to this standard does not require use of the trademark P-NET. Use of the trademark P-NET requires permission of the IPUO..

[2] PROFIBUS is the trade name of the PROFIBUS Nutzerorganisation e.V. (PNO), control of trade name use is given to the non profit organisation PNO. This information is given for the convenience of users of this International Standard and does not constitute an endorsement by ISO of the trademark holder or any of its products. Compliance to this standard does not require use of the trade name PROFIBUS. Use of the trade name PROFIBUS requires permission of the PNO.

[3] WorldFIP® is a registered trademark of the WorldFIP Association. Control of trademark use is given to the non profit organisation WorldFIP Association. This information is given for the convenience of users of this International Standard and does not constitute an endorsement by ISO of the trademark holder or any of its products. Compliance to this standard does not require use of the trademark WorldFip. Use of the trademark WorldFIP requires permission of the WorldFIP Association.

[4] ControlNet™ is a trade name of ControlNet International, Ltd. This information is given for the convenience of users of ISO 15745 and does not constitute an endorsement by ISO of the trademark holder or any of its products. Compliance to this standard does not require use of the trade name ControlNet™. Use of the trade name ControlNet™ requires permission of ControlNet International, Ltd.

[5] INTERBUS is a trade name of Phoenix Contact GmbH & Co. KG, control of trade name use is given to the non profit organisation INTERBUS Club. This information is given for the convenience of users of this International Standard and does not constitute an endorsement by ISO of the trademark holder or any of its products. Compliance to this standard does not require use of the trade name INTERBUS. Use of the trade name INTERBUS requires permission of the INTERBUS Club.

Industrial automation systems and integration — Open systems application integration framework —

Part 3:
Reference description for IEC 61158-based control systems

1 Scope

This part of ISO 15745 defines the technology specific elements and rules for describing both communication network profiles and the communication related aspects of device profiles specific to IEC 61158-based control systems. Profiles for ISO/IEC 8802-3-based control systems are outside the scope of this part of ISO 15745.

NOTE Generic elements and rules for describing integration models and application interoperability profiles, together with their component profiles (process profiles, information exchange profiles, and resource profiles) are specified in ISO 15745-1.

This part of ISO 15745 is to be used in conjunction with ISO 15745-1 to describe an application integration framework.

2 Normative references

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

ISO 639-1:2002, *Codes for the representation of names of languages – Part 1: Alpha-2 code*

ISO 639-2:1998, *Codes for the representation of names of languages – Part 2: Alpha-3 code*

ISO 3166-1:1997, *Codes for the representation of names of countries and their subdivisions – Part 1: Country codes*

ISO 9506-1:2000, *Industrial automation systems – Manufacturing Message Specification – Part 1: Service definition*

ISO 15745-1:2003, *Industrial automation and systems integration – Open systems application integration framework – Part 1: Generic reference description*

ISO/IEC 10646-1:2000, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*

IEC 61131-3:2003, *Programmable controllers – Part 3: Programming languages*

IEC 61158 (all parts), *Digital data communications for measurement and control – Fieldbus for use in industrial control systems*

IEC 61784-1:2003, *Digital data communications for measurement and control - Part 1: Profile sets for continuous and discrete manufacturing relative to fieldbus use in industrial control systems*

IEC 61804-2, *Function blocks (FB) for process control – Part 2: Specification of FB concept and electronic device description language (EDDL)*[6]

ANSI TIA/EIA-232-F:1997, *Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange*

ANSI TIA/EIA-485-A:1998, *Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems*

EN 50170:1996 Volume 3 Part 7-3, *General purpose field communication system – WorldFIP – Network Management*

IEEE Std 754-1985 (R1990), *IEEE Standard for Binary Floating Point Arithmetic*

REC-xml-20001006, *Extensible Markup Language (XML) 1.0 Second Edition – W3C Recommendation 6 October 2000*

REC-xmlschema-1-20010502, *XML Schema Part 1: Structures – W3C Recommendation 02 May 2001*

REC-xmlschema-2-20010502, *XML Schema Part 2: Datatypes – W3C Recommendation 02 May 2001*

RFC 1738:1994, *Uniform Resource Locators (URL) – Internet Engineering Task Force (IETF), Request for Comments (RFC)*

RFC 1759:1995, *Printer MIB – Internet Engineering Task Force (IETF), Request for Comments (RFC)*

UML V1.4, *OMG - Unified Modeling Language Specification (Version 1.4, September 2001)*

3 Terms and definitions

NOTE The UML terminology and notation used in this document is described in Annex A of ISO 15745-1:2003.

For the purposes of this document, the terms and definitions given in ISO 15745-1 apply.

[6] Edition 1 to be published

4 Abbreviated terms

AIF	Application Integration Framework
AIP	Application Interoperability Profile
AL	Application Layer
ASCII	American Standard Code for Information Interchange (see ISO/IEC 10646)
ASE	Application Service Element (see IEC 61158-5)
CIP™[7]	Common Industrial Protocol
CP	Communication Profile
CRC	Cyclic Redundancy Check
DL	Data Link Layer
DP	PROFIBUS DP services and protocol
EDD	Electronic Device Description
EDDL	Electronic Device Description Language
EDS	Electronic Data Sheet
FDCML	Field Device Configuration Markup Language
FIP	Field Industrial Protocol
GSD	Generic Station Description
HMI	Human Machine Interface
I/O	Input and Output
IAS	Industrial Automation Systems
ID	Identifier
kbit/s	1024 bit/s
LSB	Least Significant Bit
MAU	Medium Attachment Unit
Mbit/s	1024*1024 bits/s
MCS	Messaging Common Services
MMS	Manufacturing Messages Specifications (see ISO 9506-1)
MPS	Manufacturing Periodic/aperiodic Services
MS1	Master class 1 (see IEC 61784-1:2003)
MS2	Master class 2 (see IEC 61784-1:2003)

[7] CIP™ is a trade name of ControlNet International, Ltd. and Open DeviceNet Vendor Association, Inc. This information is given for the convenience of users of ISO 15745 and does not constitute an endorsement by ISO of the trademark holder or any of its products. Compliance to this standard does not require use of the trade name CIP™. Use of the trade name CIP™ requires permission of either ControlNet International, Ltd. or Open DeviceNet Vendor Association, Inc

NC	Numerical Control
NM	Network Management
OSI	Open System Interconnection
PC	Personal Computer
PID	Proportional Integration Differentiation controller
PLC	Programmable Logic Controller
RC	Robot Control
SM_MPS	System Management Manufacturing Periodic Specification
SMS	System Management Specification
SubMMS	Subset of MMS
SWNo	Softwire Number
UML	Unified Modelling Language (see UML V1.4)
VMD	Virtual Managed Device
XML	eXtensible Markup Language (see REC-xml-20001006)

5 Technology specific elements and rules

5.1 Integration models and IAS interfaces

The AIP developer shall develop the integration model using the rules described in ISO 15745-1, and shall ensure that the IEC 61158-based device and communication network profiles (whether representing the interface requirements or those derived from existing devices/communication networks) include the necessary IAS interfaces. The IAS interfaces included in the profile shall be identified in the header section (see ISO 15745-1:2003, 7.2.2).

NOTE IAS interfaces are described in ISO 15745-1:2003, Annex B.

5.2 Profile templates

5.2.1 General

The IEC 61158 technology specific profile templates are derived from the generic profile templates specified in ISO 15745-1:2003, clause 7.

5.2.2 Contents and syntax

ISO 15745 specifies profile templates that are XML schemas (REC-xmlschema-1-20010502 and REC-xmlschema-2-20010502) and use a common general structure. The device and communication network profiles based on these templates typically contain :

— information needed to identify the connected device,

— a description of device data that can be accessed via the network,

— a description of the communication capabilities supported by the device,

— additional vendor-specific information.

However, some IEC 61158 technologies use specific legacy ASCII syntax. Hence, for backward compatibility, template definitions of any technology (Annex A to Annex E) include all or a relevant subset of the following:

— communication network and device profile templates, as defined in ISO 15745-1,

— ISO 15745 template to encapsulate files with legacy ASCII syntax ("wrapper"),

— legacy ASCII syntax.

5.2.3 Header

The profile template header defined in ISO 15745-1:2003, 7.2.2, is used for IEC 61158 technology specific profile templates. Each technology uses one or more names to identify the technology or its particular component(s) (see Table 1). The selected name shall be stored in the ProfileTechnology attribute in the header section.

Table 1 — ProfileTechnology names

ProfileTechnology name	Technology
ControlNet	ControlNet
CIP	ControlNet
EDS	ControlNet
GSD	PROFIBUS
EDDL	PROFIBUS
P-NET	P-NET
WorldFIP	WorldFIP
INTERBUS	INTERBUS
FDCML	INTERBUS

5.3 Technology specific profiles

The technology specific communication network profile structure and communication related aspects of device profile structure based on IEC 61158 fieldbus technologies are described in clause 6. The technologies included are:

— ControlNet (see 6.1)

— PROFIBUS (see 6.2)

— P-NET (see 6.3)

— WorldFIP (see 6.4)

— INTERBUS (see 6.5).

The related profile template definitions are specified in Annex A to Annex E.

6 Device and communication network profiles for IEC 61158-based control systems

6.1 ControlNet

6.1.1 Device profile

6.1.1.1 General

Figure 1 shows the class structure of the ControlNet device profile.

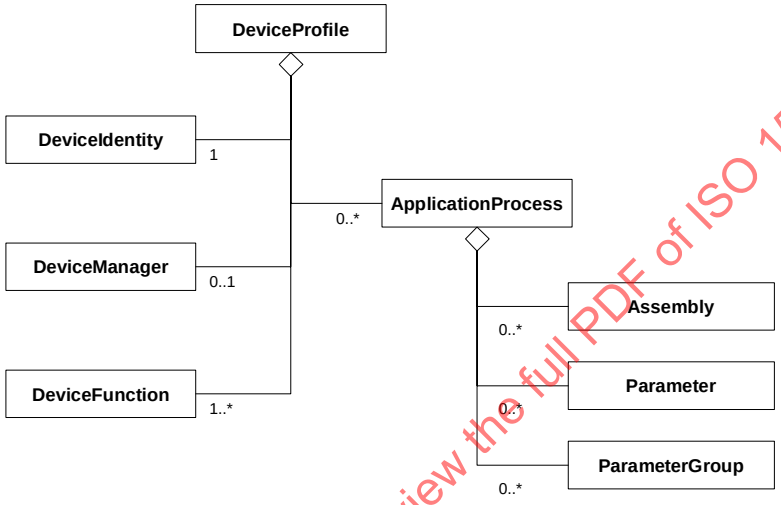

Figure 1 — ControlNet device profile class diagram

The available formats for ControlNet device profiles are described in A.2.

The XML schema representing the ControlNet device profile template is defined in A.2.1.3.3. The file name of this XML schema shall be “CIP_Device_Profile.xsd”.

NOTE The ControlNet device profile class diagram shown in Figure 1 defines the main classes. These classes are further decomposed ; details are defined in Annex A.

The XML schema representing the encapsulation of a legacy ControlNet EDS into the ISO 15745 device profile template is defined in A.2.2.2. The file name of this XML schema shall be “EDS_Device_Profile_wrapper.xsd”. The legacy EDS ASCII syntax itself is described in A.4.

6.1.1.2 Device identity

The DeviceIdentity class contains attributes which uniquely identify the device, and supports services which allow the retrieval of this information from the device.

These attributes provide in particular:

— manufacturer's identification (name and identification code);

— device identification (device type, product name, revision, serial number);

— device classification;

— location of storage of additional information (e.g. icons).

6.1.1.3 Device manager

The DeviceManager class contains attributes and supports services used to monitor and configure the device.

These attributes provide in particular:

— revision of the ControlNet identity object;

— information on device structure (for devices integrated in a modular system).

Services allow:

— device reset;

— retrieval of DeviceManager attributes.

6.1.1.4 Device function

The DeviceFunction class contains attributes and supports services which enable the management (e.g. configuration) of a function of the device.

EXAMPLE Examples of DeviceFunction objects are Overload, Presence Sensing, Analogue Input, and Discrete Output objects.

NOTE The DeviceFunction class is not defined in ISO 15745-3.

6.1.1.5 Application process

Figure 2 shows the class structure of the ApplicationProcess class.

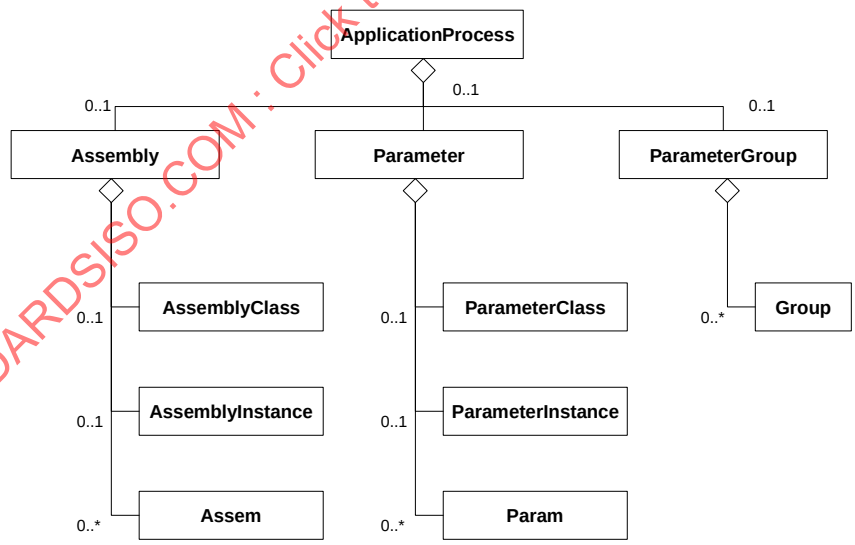

Figure 2 — ControlNet ApplicationProcess class diagram

The Assembly class assembles several application process data items into a single block for optimisation of communications. The Parameter class provides a standardized interface for accessing individual application process data items. The ParameterGroup class specifies groups of related parameters for a specific purpose (e.g. configuration, monitoring).

The Assembly class and the Parameter class support attributes and services both at the class and instance levels.

The Assem, Param and Group classes specify individual instances of the main classes.

NOTE The Assembly class and the Parameter class correspond to the ControlNet Assembly object and Parameter object. The Assembly object is fully specified in IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2).

6.1.2 Communication network profile

6.1.2.1 General

Figure 3 shows the class structure of the ControlNet communication network profile.

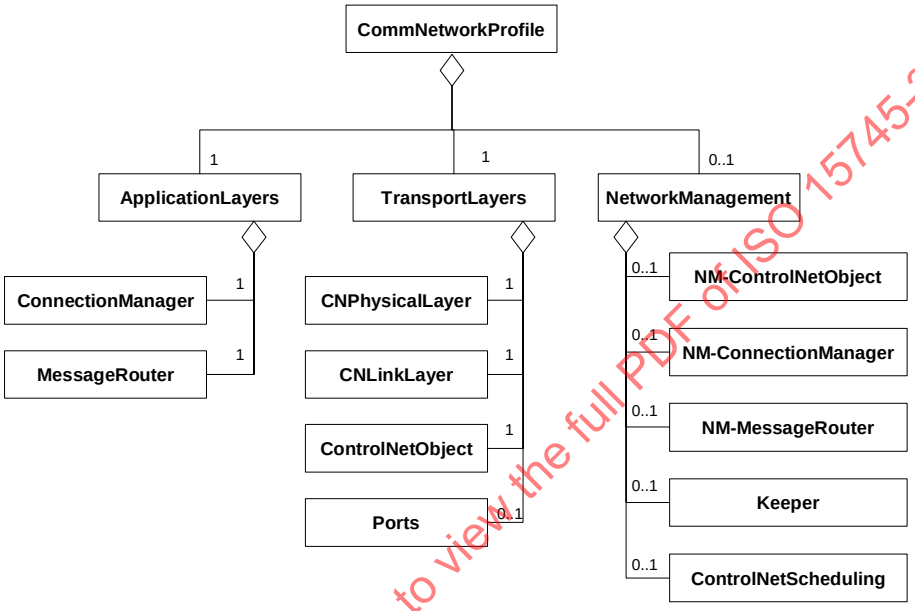

Figure 3 — ControlNet communication network profile class diagram

The available formats for ControlNet communication network profiles are described in A.3.

The XML schema representing the ControlNet communication network profile template is defined in A.3.1.3. The file name of this XML schema shall be "CNet_CommNet_Profile.xsd".

The XML schema representing the encapsulation of a legacy ControlNet EDS into the ISO 15745 communication network profile template is defined in A.3.2.2. The file name of this XML schema shall be "EDS_CommNet_Profile_wrapper.xsd". The legacy EDS ASCII syntax itself is described in A.4.

6.1.2.2 Application layers

The ControlNet ApplicationLayers class represents the combined profiles for the upper 3 OSI layers of the ControlNet communication network integration model.

It is further divided into several classes, as shown in Figure 3:

— ConnectionManager defines the properties associated with connections and connection management;

— MessageRouter defines the properties associated with internal message routing in the device.

NOTE The corresponding Connection Manager object and Message Router object are fully specified in IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2).

6.1.2.3 Transport layers

The ControlNet TransportLayers class represents the combined profiles for the lower 4 OSI layers of the ControlNet communication network integration model.

It is further divided into several classes, as shown in Figure 3:

— CNPhysicalLayer identifies the physical layer characteristics (e.g. connectors, delays);

— CNLinkLayer and ControlNetObject define the properties associated with data link layer configuration and monitoring;

— Ports identifies the device ports which are able to route messages from one link to another link.

NOTE The corresponding ControlNet object is fully specified in IEC 61158-4:2003 (Type 2).

6.1.2.4 Network management

The ControlNet NetworkManagement class represents the network configuration and performance adjustment capabilities of the ControlNet communication network integration model.

It is further divided into several classes, as shown in Figure 3:

— Keeper defines the properties associated with network management;

— ControlNetScheduling defines the properties associated with allocation of scheduled transmission time;

— NM-ConnectionManager , NM-MessageRouter and NM-ControlNetObject define the properties associated with class management of the corresponding objects.

NOTE The corresponding Keeper object and ControlNet Scheduling object are fully specified in IEC 61158-4:2003 (Type 2).

6.2 PROFIBUS

6.2.1 Device profile

Figure 4 shows the class structure of the PROFIBUS device profile.

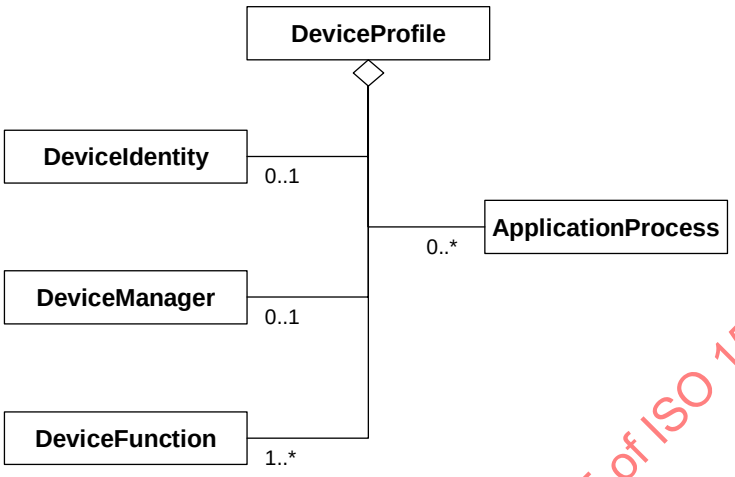

Figure 4 — PROFIBUS device profile class diagram

The class information for DeviceIdentity, DeviceManager, and ApplicationProcess is given by the legacy device profile format EDD and GSD.

The DeviceFunction class contains attributes and supports services which enable the management (e.g. configuration) of a function of the device.

EXAMPLE Examples of DeviceFunction objects are Analogue Input, and Discrete Output objects.

NOTE The DeviceFunction class is not defined in ISO 15745-3.

The available formats for PROFIBUS device profiles are described in B.2.

The XML schema representing the encapsulation of a legacy EDD and GSD of a PROFIBUS device into the ISO 15745 device profile template is defined in B.2. The file name of this XML schema shall be "EDDL_Device_Profile_wrapper.xsd" or "GSD_Device_Profile_wrapper.xsd". The legacy EDDL ASCII syntax itself is described in IEC 61804-2, using the PROFIBUS profile of IEC 61804-2: Ed.1, F.2.

Simple devices, which do not have the need to have an EDD, shall reference the GSD.

Equivalences of the classes DeviceIdentity, DeviceManager, and ApplicationProcess are defined in IEC 61804-2 or in the case of referencing an GSD in B.4.

6.2.2 Communication network profile

6.2.2.1 General

Figure 5 shows the class structure of the PROFIBUS communication network profile.

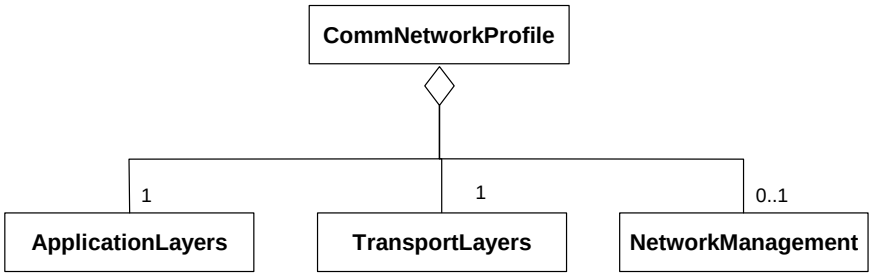

Figure 5 — PROFIBUS communication network profile class diagram

The available formats for PROFIBUS communication network profiles are described in B.3.

The file name of this XML schema shall be “GSD_CommNet_Profile.xsd”.

The XML schema representing the encapsulation of a legacy GSD of PROFIBUS devices into the ISO 15745 communication network profile template is defined in B.3. The file name of this XML schema shall be “GSD_CommNet_Profile_wrapper.xsd”. The legacy GSD ASCII syntax itself is described in B.4 and B.5.

6.2.2.2 Application layers

The PROFIBUS ApplicationLayers class represents the combined profiles for the upper 3 OSI layers of the PROFIBUS communication network integration model, see IEC 61784-1:2003 CP3/1 and CP3/2, especially the subclauses for AL protocol and AL service.

6.2.2.3 Transport layers

The PROFIBUS TransportLayers class represents the combined profiles for the lower 4 OSI layers of the PROFIBUS communication network integration model, see IEC 61784-1:2003 CP3/1 and CP3/2, especially the subclauses for DL protocol and DL service.

6.2.2.4 Network management profile

The NetworkManagement profile class represents the network configuration and performance adjustment capabilities of the PROFIBUS communication network integration model.

6.3 P-NET

6.3.1 Device profile

Variables in P-NET devices are normally organised into Channels. A Channel is a collection of related variables and functions for a single process signal. A Channel can hold up to 16 registers, each having its own SoftWire number (SWNo). The contents of these registers can be of any data type, including complex structures, such as multi-dimensional arrays and databases.

At least one Channel, the ServiceChannel, shall be present in every P-NET device. The ServiceChannel contains information used for making easy service on the device, e.g. a Globally Unique Identifier (GUID), error information, etc. Any other Channels that are included in a device depend on the device type. A device can consist of multiples of the same Channel type and/or a mixture of different Channel types.

EXAMPLE 1 Examples of various standardised Channel types include: Digital Input, Digital Output, Analogue measurement, PID, etc. Additional Channel types are user definable.

A P-NET device profile shall describe all network accessible variables and channels in the device in the ApplicationProcess object. The device profile also includes objects for device management and identification (see Figure 6).

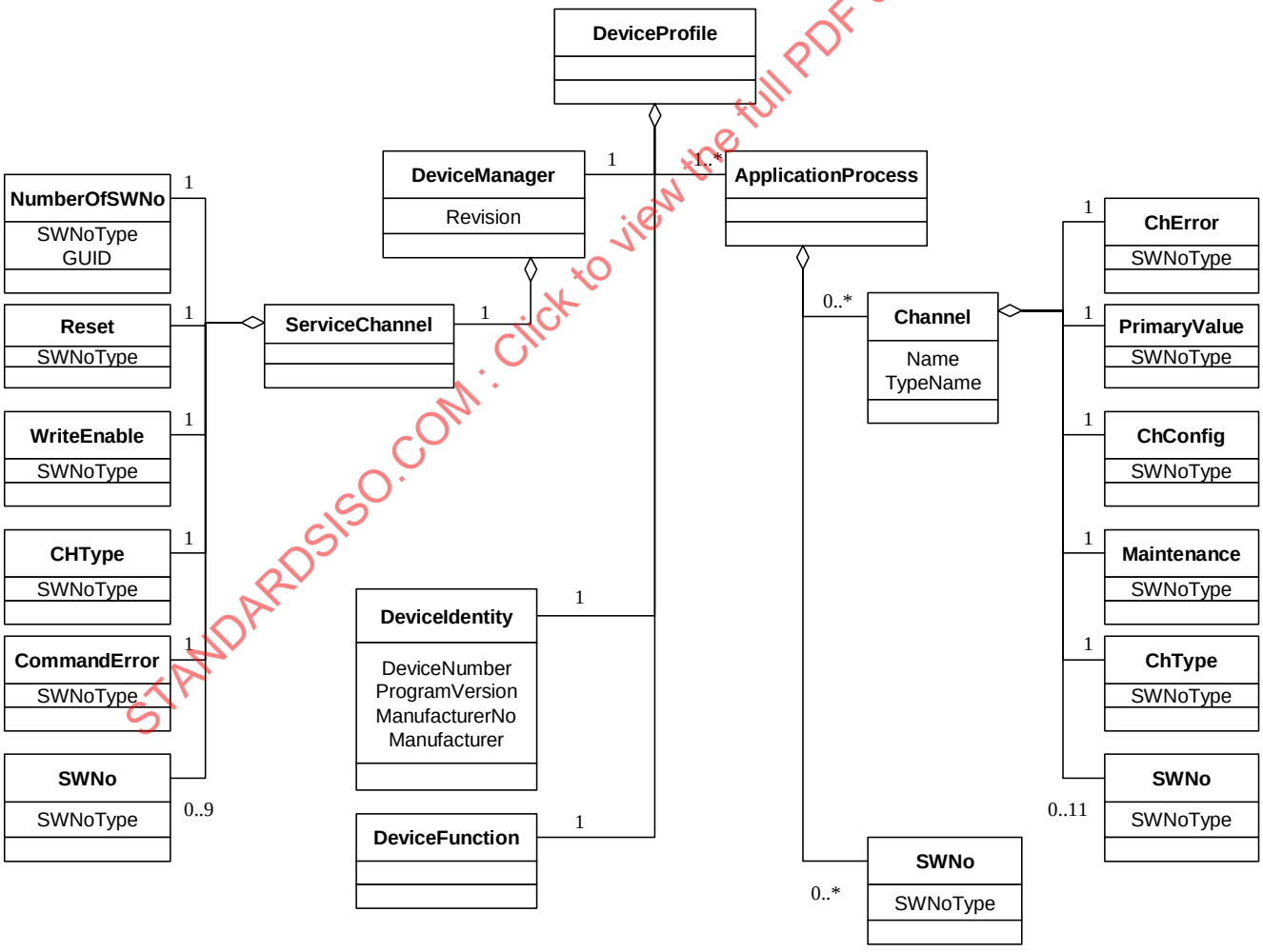

Figure 6 — P-NET device profile class diagram

EXAMPLE 2 Examples of Device Function objects are Flowmeter, Controller, Analogue Input 4-20 mA, Digital I/O objects.

NOTE In P-NET, there are no special services for accessing properties of the DeviceFunction, DeviceManager and DeviceIdentity objects. Instead, these properties can be mapped to the ApplicationProcess objects, and accessed from the network by using the Application Protocol Data Unit (APDU) specified in the P-NET ApplicationLayers class (see 6.3.2.2).

Attributes and sub-classes for the device profile classes are detailed in C.1, which specifies the XML schemas required for device profiles. The file name of the XML schema shall be "P-NetDeviceProfile.XSD".

6.3.2 Communication network profile

6.3.2.1 General

Figure 7 shows the class structure of the P-NET communication network profile.

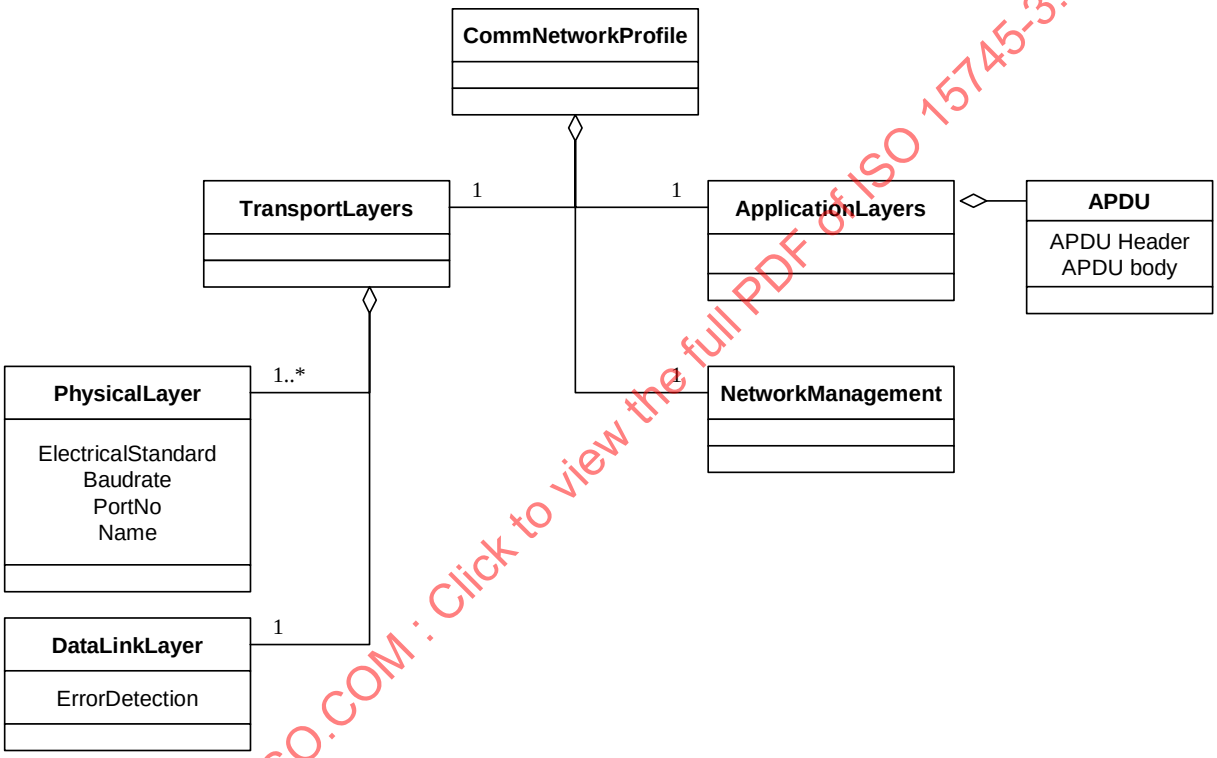

Figure 7 — P-NET communication network profile class diagram

The XML schema representing the P-NET communication network profile is defined in C.2. The file name of the XML schema shall be "P-NetCommNetworkProfile.XSD".

6.3.2.2 Application layers

The application layer of a P-NET device shall always provide the services and protocol elements needed for accessing the variables defined in the P-NET device profile. The services and protocol elements are represented in an APDU.

6.3.2.3 Transport layers

6.3.2.3.1 General

The TransportLayers class shall represent the combined profiles for the lower 4 OSI layers of the communication network integration model. The TransportLayers object is composed of one or more PhysicalLayer objects (one for each physical port) and a DataLinkLayer object.

6.3.2.3.2 PhysicalLayer

The PhysicalLayer object(s) shall specify the supported electrical standard and baud rate(s). Valid electrical standards are RS485 and RS232. Valid baud rates for RS232 are 1200, 2400, 4800, 9600, 19200, and 38400 bits/sec. For RS485 the only valid baud rate is 76800 bits/sec.

6.3.2.3.3 DataLinkLayer

The Data-Link layer of a P-NET device shall always provide the protocol elements needed according to the device class, specified by the NetworkManagement object. Valid device classes are Slave, Simple node or Master.

The DataLinkLayer object shall specify the supported error detection method(s). Valid methods are Normal and Reduced.

6.3.2.4 NetworkManagement

The NetworkManagement class diagram is shown in Figure 8. It shall comprise of at least one object of type Master, Simple Node or Slave for each physical connection point defined in the OSI-U profile.

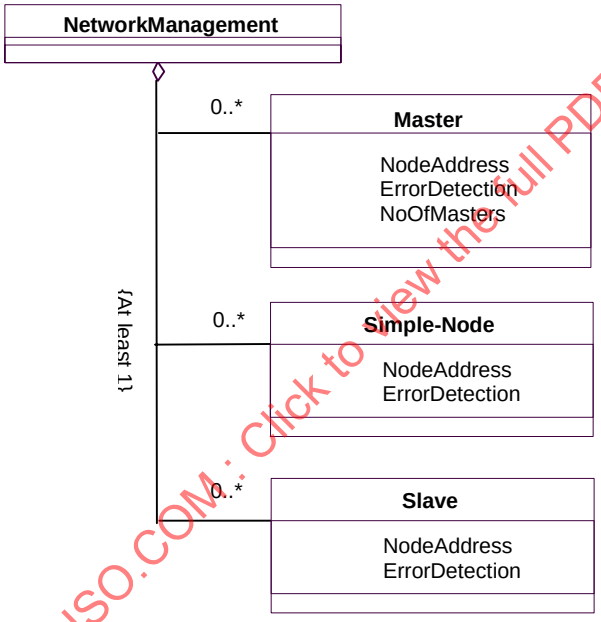

Figure 8 — P-NET NetworkManagement profile class diagram

6.4 WorldFIP

6.4.1 Device profile

6.4.1.1 Main class structure

Figure 9 shows the class structure of the WorldFIP device profile.

The required format for WorldFIP device profiles is described in D.1. The XML schema representing the WorldFIP device profile template is defined in D.1.3. The file name of the XML schema shall be 'WFIPDEVP.XSD'.

NOTE 1 For better readability the WorldFIP device profile class diagram has been divided in four class diagrams.

NOTE 2 All these classes are mapped to the same XML schema defined in D.1.3.

NOTE 3 The WorldFIP device profile class diagrams shown in Figure 9 to Figure 12 define the main classes. Some classes are further decomposed; details are defined in D.1.

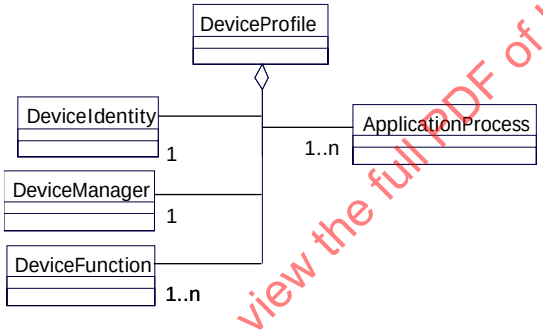

Figure 9 — WorldFIP device profile class diagram

6.4.1.2 Device identity

The DeviceIdentity class shall consist of the child classes shown in Figure 10 and specified in Table 2.

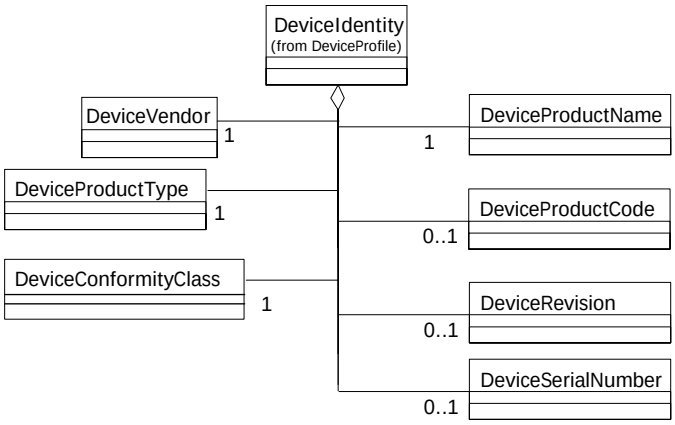

Figure 10 — WorldFIP DeviceIdentity class diagram

Table 2 — Decomposition of DeviceIdentity class

Class	Description	Profile	Type	Instance
DeviceVendor	name of the manufacturer or vendor of the device	X	X	X
DeviceProductType	device type	X	X	X
DeviceConformityClass	class of conformity (see D.1.2)	X	X	X
DeviceProductName	vendor specific name of the product	X	X	X
DeviceProductCode	unique ID, identifying the device type, the format is at the vendor's discretion		X	X
DeviceRevision	revision of the specification to which this device conforms	X	X	X
DeviceSerialNumber	serial number of device instance			X
NOTE The columns Profile, Type and Instance indicate whether a certain child class is suitable for usage in a device profile, device type description or device instance description.				

6.4.1.3 Device manager

6.4.1.3.1 General

Figure 11 shows the WorldFIP representation of the DeviceManager object.

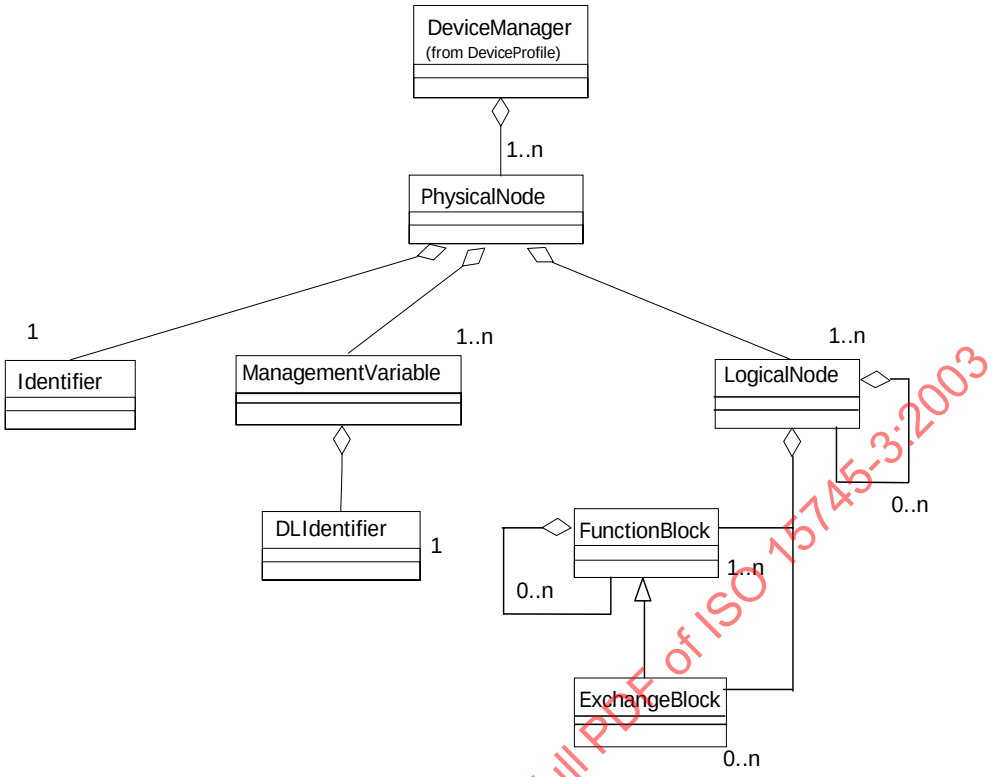

Figure 11 — WorldFIP DeviceManager class diagram

Any device is represented by a physical node which refers to a WorldFIP equipment, participating in application functions and supporting communication functions.

NOTE A device may comprise several physical nodes if it possesses several points to the same network or to different networks (e.g.: gateways, centralization equipment, etc.).

6.4.1.3.2 PhysicalNode

6.4.1.3.2.1 Identifier

Each physical node has a unique identifier.

6.4.1.3.2.2 ManagementVariable

Physical nodes are managed with SM_MPS and implement two or more management variables whose roles and identifiers are fixed, based on the value of DeviceConformityClass (see D.1.2).

6.4.1.3.2.3 LogicalNode

A user application in a physical node is decomposed in a series of logical nodes (at least one), representing an execution context of a processing operation performed in a physical node on behalf of the user application. A user application is described by one or more logical nodes.

A logical node may itself be composed of logical nodes.

Details for logical nodes are given in D.1.2.

6.4.1.3.2.4 FunctionBlock

The application functions of a logical node are composed of a set of user layer objects being represented by function blocks.

A function block allows identification and definition of a simple or complex elementary processing operation.

A function block may itself be composed of function blocks.

Details for function blocks are given in D.1.2.

6.4.1.3.2.5 ExchangeBlock

Exchange blocks are specialised function blocks. Their role is to model the mechanisms of the data exchanges which are necessary between the various remote user entities (i.e. logical nodes) located in different devices (i.e. physical nodes) in order to perform and, if necessary, to synchronize their processing functions.

6.4.1.4 Device function

Figure 12 shows the WorldFIP representation of the DeviceFunction object.

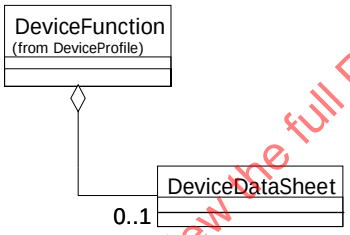

Figure 12 — WorldFIP DeviceFunction class diagram

NOTE The definition of the DeviceDataSheet of the DeviceFunction is outside the scope of this International Standard.

6.4.1.5 Application process

The ApplicationProcess object is composed of function blocks and optional exchange blocks presented in 6.4.1.3.2.4 and 6.4.1.3.2.5.

6.4.2 Communication network profile

6.4.2.1 Main class structure

The required format for WorldFIP communication network profiles is described in D.2. The XML schema representing the WorldFIP communication network profile template is defined in D.2.5. The file name of the XML schema shall be 'WFIPCOMP.XSD'.

NOTE 1 For better readability the WorldFIP communication network profile class diagram has been divided in five class diagrams.

NOTE 2 All these classes are mapped to the same XML schema defined in D.2.5.

NOTE 3 The WorldFIP communication network profile class diagrams shown in Figure 13 to Figure 18 define the main classes. Some classes are further decomposed; details are defined in D.2.

Figure 13 shows the class structure of the WorldFIP communication network profile.

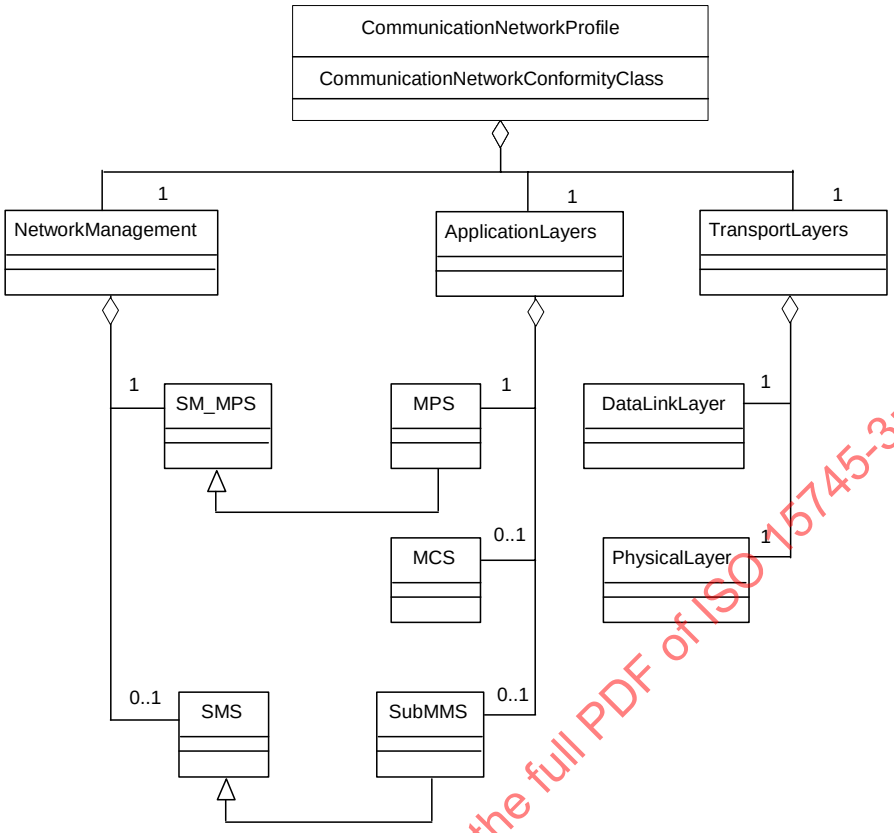

Figure 13 — WorldFIP communication network profile class diagram

The semantics of the CommunicationNetworkConformityClass attribute are detailed in D.2.

6.4.2.2 Application layers

6.4.2.2.1 General

An ApplicationLayers class shall represent the combined profiles for the upper 3 OSI layers of the communication network integration model. It states the supported application service elements and their associated services.

The ASEs (Application Service Elements) defined for WorldFIP can be stated by the following profile objects:

— MPS (Manufacturing Periodic/aperiodic Services);
MPS is a periodic/aperiodic broadcast update of a distributed database. These services are present in every WorldFIP communication network profile.

— MCS (Message Common Services);
MCS is an optional interface layer supporting messaging services.

— SubMMS;
SubMMS is an optional subset of MMS (Manufacturing Messages Specifications) services.

6.4.2.2.2 MPS

6.4.2.2.2.1 General

Figure 14 shows the representation of WorldFIP MPS application layer.

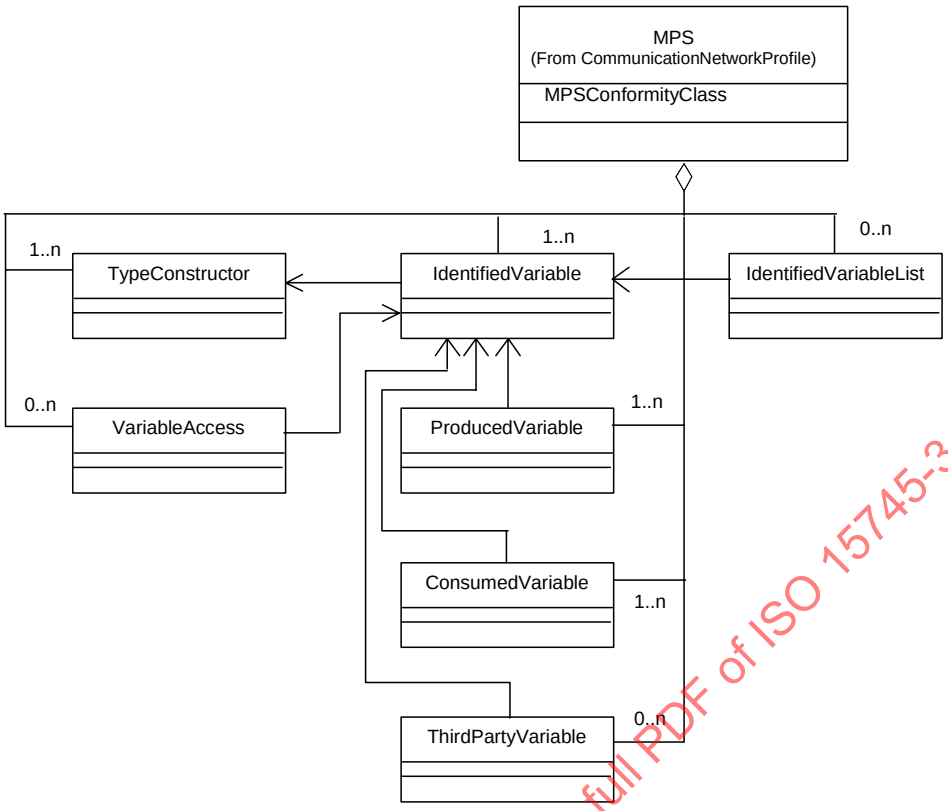

Figure 14 — WorldFIP communication network profile – MPS application layer

The semantics of the MPSConformityClass attribute are detailed in D.2.2.1.2.

6.4.2.2.2.2 IdentifiedVariable

MPS provides periodic and aperiodic exchange of variable between one producer and one or more application entities. All the common attributes of a variable are defined in the IdentifiedVariable class. Details for identified variables are given in the XML schema in D.2.5.

6.4.2.2.2.3 ProducedVariable

Within a distributed application, one unique application entity is declared as the producer of a variable's values. The ProducedVariable class gathers all the attributes relative to a variable within a producer application layer. Details for produced variables are given in the XML schema in D.2.5.

6.4.2.2.2.4 ConsumedVariable

One or several application entities are declared as consumers of a variable's values. The ConsumedVariable class gathers all the attributes relative to a variable within a consumer application layer. Details for consumed variables are given in the XML schema in D.2.5.

6.4.2.2.2.5 ThirdPartyVariable

Some application entities, while being neither producer, nor consumer of a variable's values, may have knowledge of it in order to invoke the updating of its value.

6.4.2.2.2.6 IdentifiedVariableList

The MPS application layer supports definition and handling of variable lists. A variable list is globally defined and instantiated at the application entity level. It is exclusively composed of consumed variables. Details for variable lists are given in the XML schema in D.2.5.

6.4.2.2.2.7 TypeConstructor

The variable type is defined by the TypeConstructor class which is referred by the variable. Figure 15 shows the class structure of the TypeConstructor object.

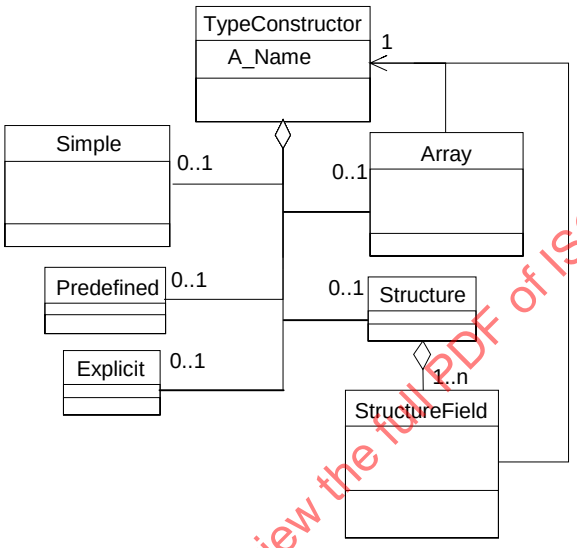

Figure 15 — WorldFIP MPS Variable TypeConstructor

Details for type constructors are given in D.2.2.1.1.

6.4.2.2.2.8 VariableAccess

The VariableAccess refers to a variable and indicates the access mode that is used. Details for variable access are given in the XML schema in D.2.5.

6.4.2.2.3 MCS

Figure 16 shows the representation of the WorldFIP MCS application layer.

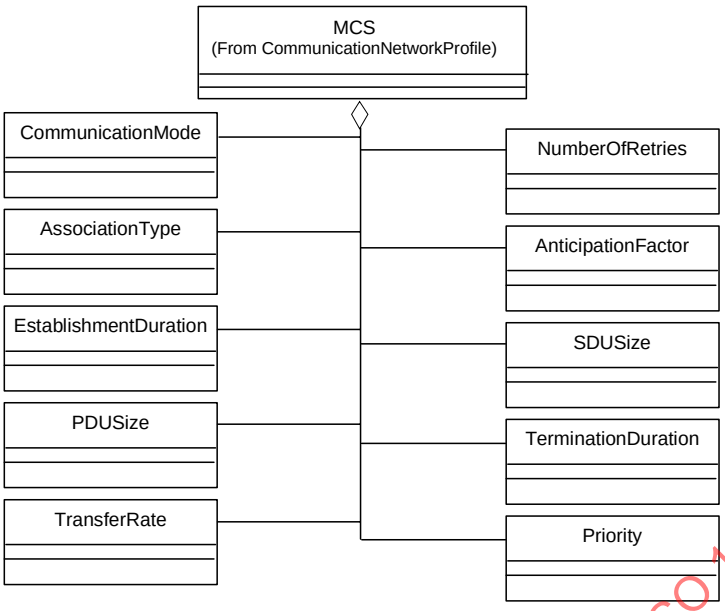

Figure 16 — WorldFIP communication network profile – MCS application layer

The MCS application layer provides services for monitoring application associations and to enable transfer of data units from the MMS application layer in the associated and non associated mode.

6.4.2.2.4 SubMMS

Figure 17 shows the representation of the WorldFIP SubMMS application layer.

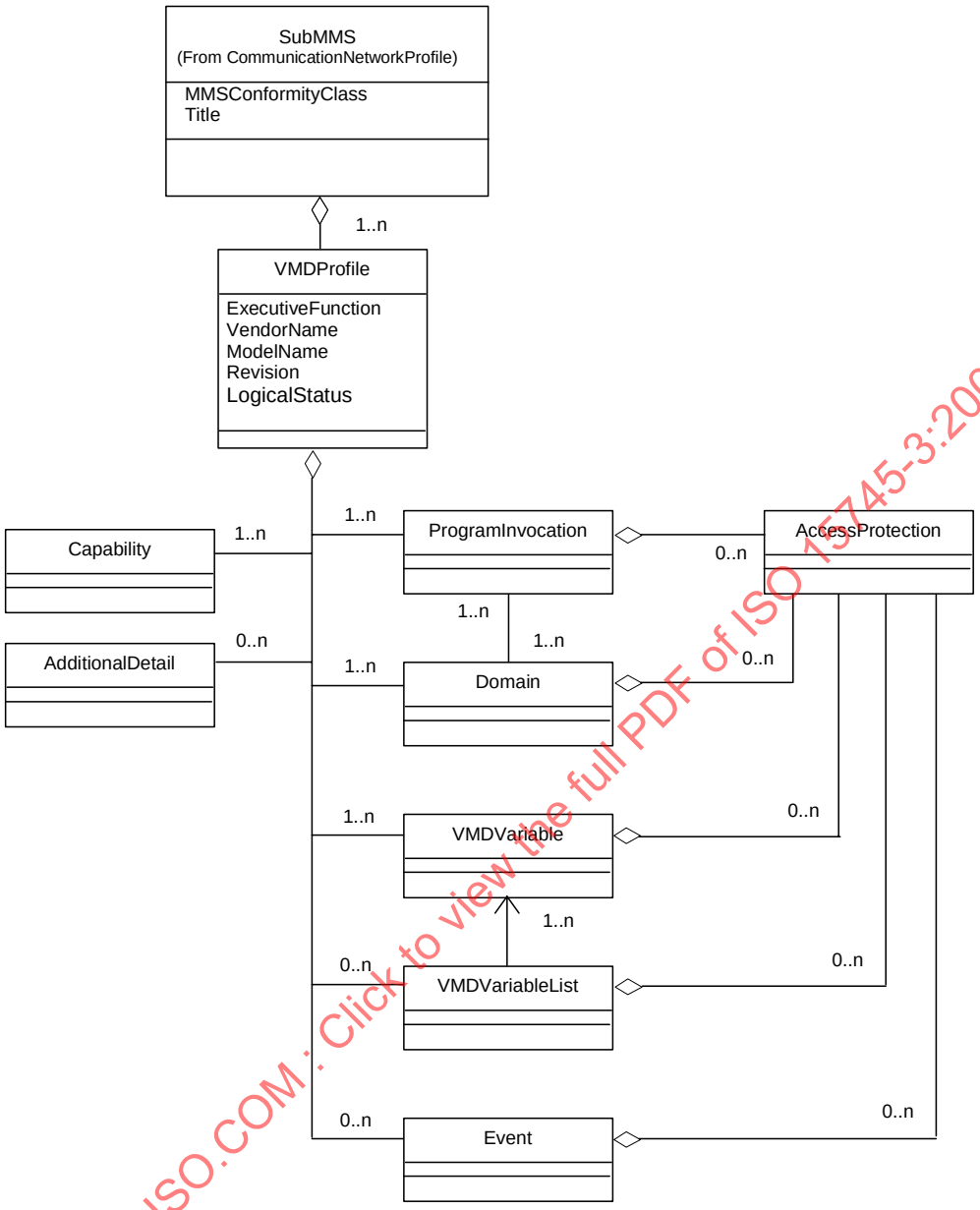

Figure 17 — WorldFIP communication network profile – SubMMS application layer

The semantics of the MMSConformityClass attribute are detailed in D.2.2.3.

6.4.2.3 Transport layers

6.4.2.3.1 General

A TransportLayers object shall represent the combined profiles for the lower 4 OSI layers of the communication network integration model. The TransportLayers object shall be divided into one or more PhysicalLayer objects and a DataLinkLayer object.

6.4.2.3.2 DataLinkLayer

6.4.2.3.2.1 General

Figure 18 shows the representation of the WorldFIP Data Link layer.

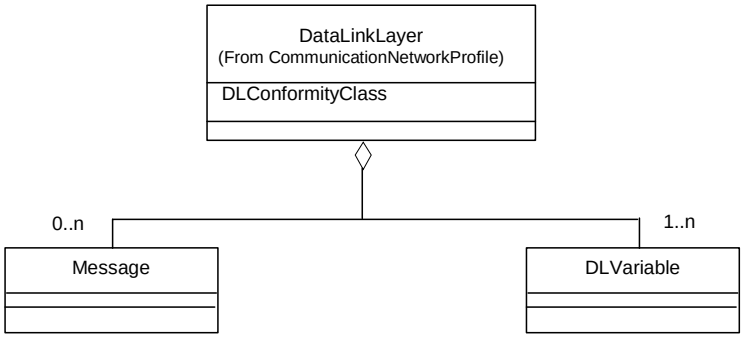

Figure 18 — WorldFIP communication network profile – Data Link layer

The semantics of the DLConformityClass attribute are detailed in D.2.3.

6.4.2.3.2.2 DLVariable

The WorldFIP Data Link layer supports cyclical variable exchange and explicit request for variable exchange.

Details for variables are given in the XML schema in D.2.5.

6.4.2.3.2.3 Message

The WorldFIP Data Link layer supports cyclical message transfer and aperiodic message transfer.

Details for messages are given in the XML schema in D.2.5.

6.4.2.3.3 PhysicalLayer

The PhysicalLayer object identifies the MAUType, the interface type and supported baud rates of the physical layer.

6.4.2.4 Network management

The NetworkManagement class shall identify the functionality for the configuration of a particular WorldFIP network. Network management for WorldFIP networks is defined in EN 50170:1996 Volume 3 Part 7-3. Depending on the network management conformity class documented in the NMConformityClass attribute, WorldFIP uses SM_MPS (based on MPS) and optionally SMS (based on MCS and SubMMS).

Network management functions are divided in three categories:

— installation and modification of the configuration,

— configuration consistency check,

— setting into service of the network.

The semantics of the NMConformityClass attribute are detailed in D.2.4.2.

6.5 INTERBUS

6.5.1 Device profile

6.5.1.1 Main class structure

Figure 19 shows the class structure of the INTERBUS device profile.

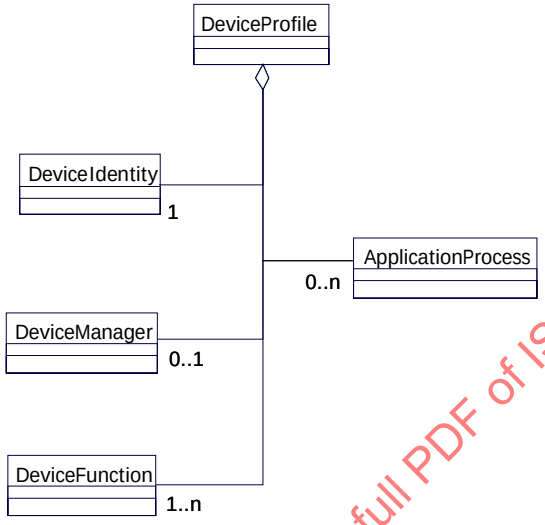

Figure 19 — INTERBUS device profile class diagram

The required format for INTERBUS device profiles is described in E.1. The XML schema representing the INTERBUS device profile template is defined in E.1.6.1. The file name of the XML schema shall be 'FDCML.XSD'.

NOTE 1 For better readability the INTERBUS device profile class diagram has been divided in six class diagrams.

NOTE 2 All these classes are mapped to the same XML schema defined in E.1.6.1.

NOTE 3 The INTERBUS device profile class diagrams shown in Figure 19 to Figure 25 define the main classes. Some classes are further decomposed; details are defined in Annex E.

6.5.1.2 Device identity

The DeviceIdentity class is defined in Figure 20.

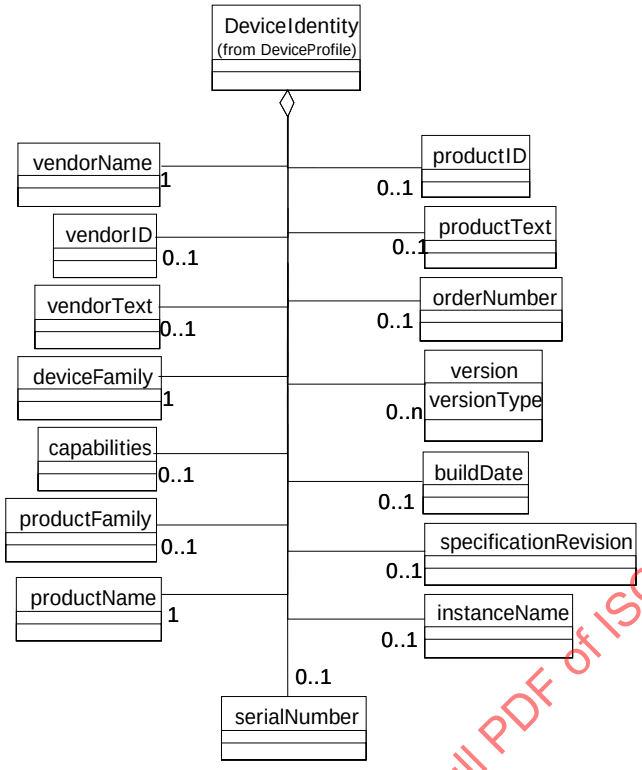

Figure 20 — INTERBUS DeviceIdentity class diagram

The DeviceIdentity class shall consist of the child classes shown in Figure 20 and specified in Table 3.

Table 3 — Decomposition of device identity object class

Class	Description	Profile	Type	Instance
vendorName	name of the manufacturer or vendor of the device	X	X	X
vendorID	IEEE OUI (Organizationally Unique Identifier) (see [3])		X	X
vendorText	can be used to provide further information on the vendor	X	X	X
deviceFamily	INTERBUS specific device type (i.e I/O). For a list of valid device types (see Table E.2)	X	X	X
capabilities	the definition of this class is not defined in ISO 15745-3		X	X
productFamily	vendor specific product family (brand name) of the device		X	X
productName	Vendor specific name of the product	X	X	X
productID	unique ID, identifying the device type, the format is at the vendor's discretion		X	X
productText	can be used to provide further information on the device	X	X	X
orderNumber	Vendor specific order number of the product		X	X
version	vendor specific product version, the versionType attribute allows the distinction of multiple versions (i.e. Hardware, Firmware)		X	X
buildDate	build date of the firmware of software constituting the major functionality of the device		X	X
specificationRevision	revision of the specification to which this device conforms	X	X	X
instanceName	name of device instance			X
serialNumber	serial number of device instance			X
NOTE The columns Profile, Type and Instance indicate whether a certain child class is suitable for usage in a device profile, device type description or device instance description.				

6.5.1.3 Device manager

6.5.1.3.1 General

Figure 21 shows the INTERBUS representation of the device manager object.

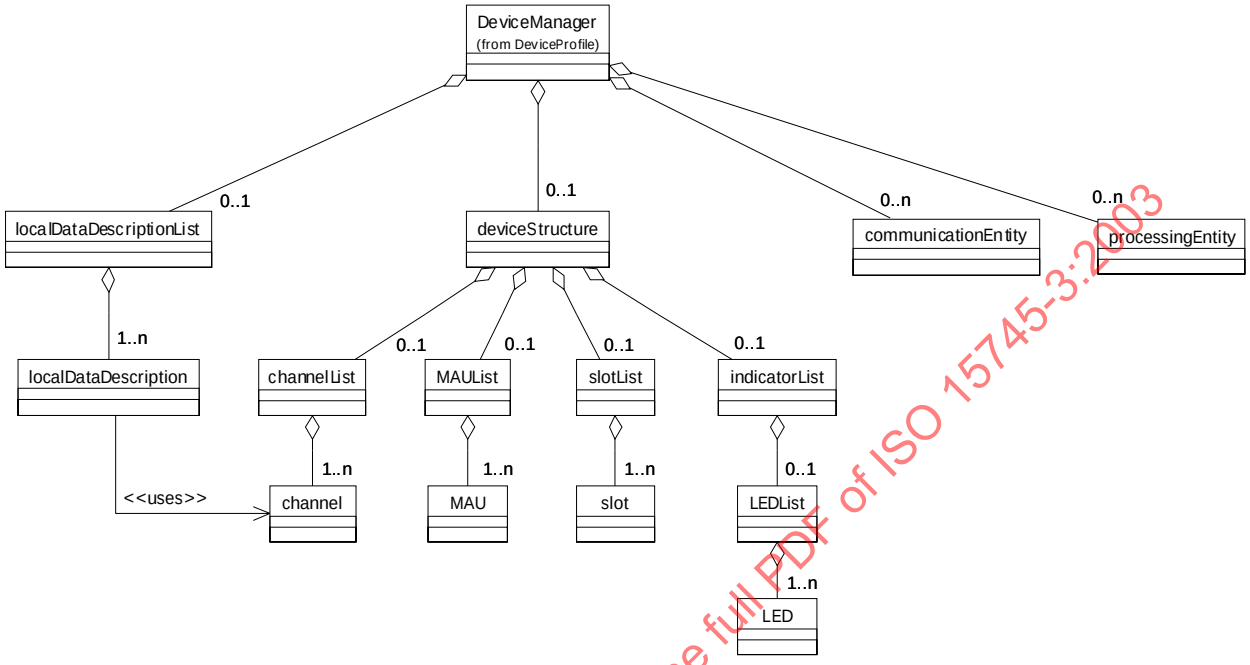

Figure 21 — INTERBUS DeviceManager class diagram

6.5.1.3.2 localDataDescriptionItemList, localDataItemlocalDataDescription

The localDataDescriptionItemList object shall be a collection of localDataItemDescription objects. A localDataItemDescription object shall describe data objects that are used only within the device context.

6.5.1.3.3 deviceStructure

6.5.1.3.3.1 Overview

The deviceStructure object shall be a container of all physical objects of the device. Such an object can be a channel (physical or logical I/O point), a MAU (Medium Attachment Unit), a slot for the connection of additional modules (as part of the device) or a LED (light-emitting diode).

6.5.1.3.3.2 channelList, channel

A channelList shall be a collection of channel objects. Channel objects shall describe physical or logical I/O points of a device.

6.5.1.3.3.3 MAUList, MAU

The MAUList shall be a collection of MAU objects. These objects shall describe the access points to network medias.

6.5.1.3.3.4 slotList, slot

A slotList object shall be a collection of slot objects. A slot object shall contain a reference to an external INTERBUS device profile exchange description.

NOTE Slots are used to describe modular devices or distinct combinations of devices.

6.5.1.3.3.5 indicatorList, LEDList, LED

A LEDList object shall be a collection of LED objects. A LED object shall describe a LED of a device.

NOTE The indicatorList class may be extended in future editions of this International Standard.

6.5.1.3.4 communicationEntity

6.5.1.3.4.1 Overview

Figure 22 shows the definition of the communicationEntity class.

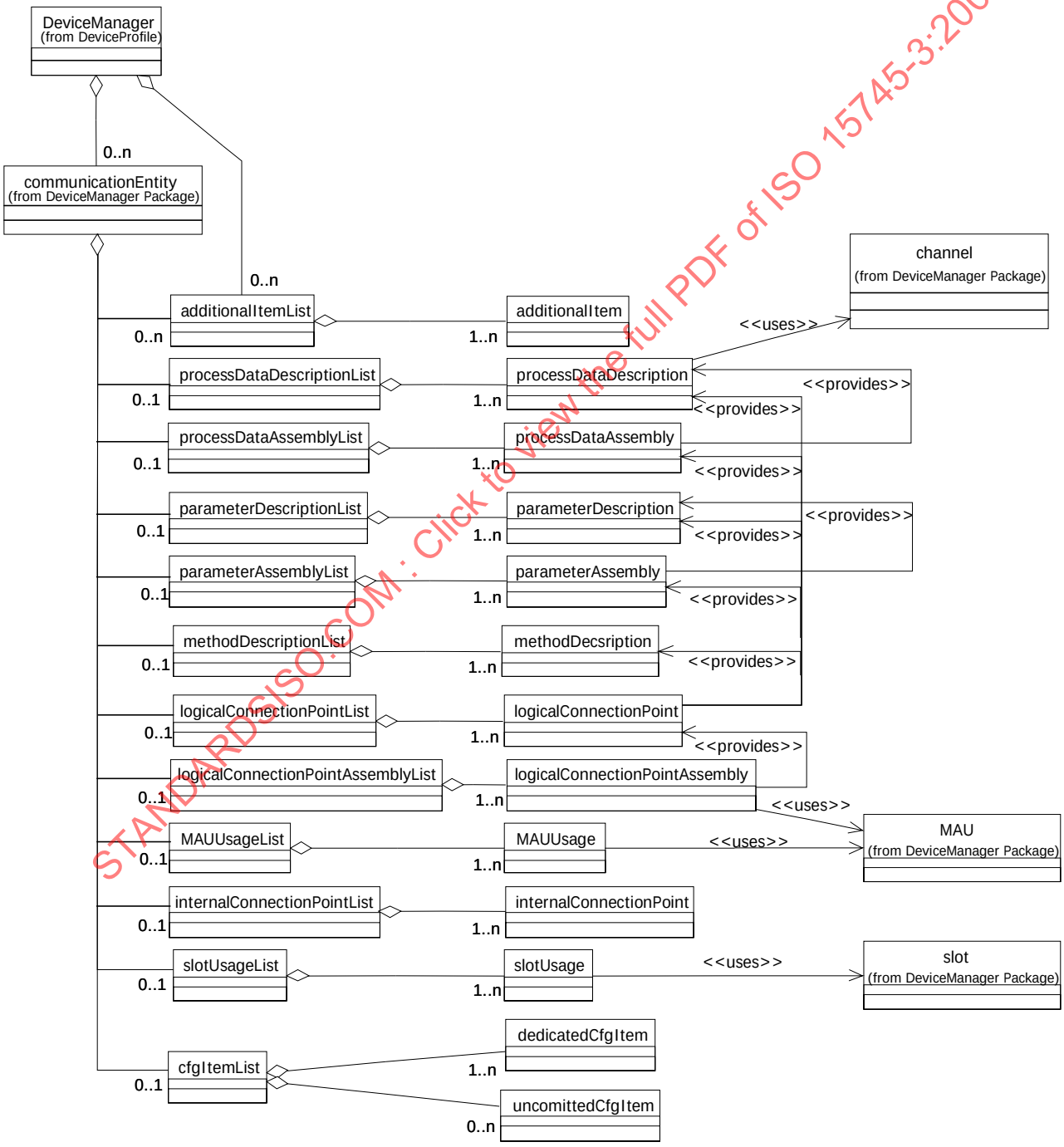

Figure 22 — INTERBUS communicationEntity class diagram

The communicationEntity shall describe an entity of a device, capable of communicating with entities of other devices and shall contain a complete set of predefined configuration items and communication object descriptions. There may be more than one communicationEntity in a device.

EXAMPLE An example for a device with two communication entities is a system coupler which 'combines' an INTERBUS slave and an INTERBUS master (see Figure 23).

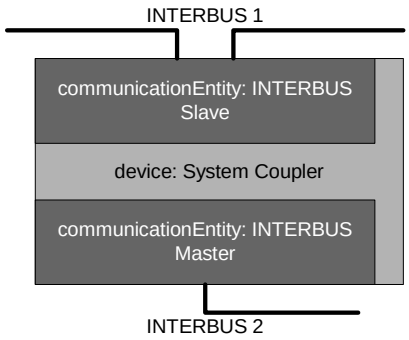

Figure 23 — Example of a device with two communication entities

6.5.1.3.4.2 additionalItemList, additionalItem

The additionalItemList shall be a collection of user defined additionalItem objects. An additionalItem object can be used to describe device properties other than configuration properties or communication objects.

NOTE The definition of the additionalItemType of additional items is outside the scope of this International Standard.

EXAMPLE Device documentation.

6.5.1.3.4.3 processDataDescriptionList, processDataDescription

The processDataDescriptionList shall be a collection of processDataDescription objects. The processDataDescription objects shall be descriptions of process data. A processDataDescription object may have a <<uses>> relation to a channel object.

EXAMPLE A process data item 'analog_out_1' uses a the terminal named '2.1'.

6.5.1.3.4.4 processDataAssemblyList, processDataAssembly

The processDataAssemblyList shall be a collection of processDataAssembly objects. The processDataAssembly objects shall be descriptions of groups of processDataDescription objects. A processDataAssembly object has <<provides>> relations to processDataDescription objects.

6.5.1.3.4.5 parameterDecsriptionList, parameterDecscription

The parameterDescriptionList shall be a collection of parameterDescription objects. The parameterDescription objects are descriptions of the variable and function invocation objects.

6.5.1.3.4.6 parameterAssemblyList, parameterAssembly

The parameterAssemblyList shall be a collection of parameterAssembly objects. The parameterAssembly objects shall be descriptions of groups of parameterDescription objects. A parameterAssembly object has <<provides>> relations to parameterDescription objects.

6.5.1.3.4.7 methodDescriptionList, methodDescription

The methodDescriptionList shall be a collection of methodDescription objects. The methodDescription objects are methods which can be invoked by a remote entity.

6.5.1.3.4.8 logicalConnectionPointList, logicalConnectionPoint

The logicalConnectionPointList shall be a collection of logicalConnectionPoint objects. A logicalConnectionPoint describes a connection endpoint (see E.1.4.6 for possible attributes).

NOTE It is assumed, that only connections between connection endpoints of the same type are used.

6.5.1.3.4.9 logicalConnectionPointAssemblyList, logicalConnectionPointAssembly

The logicalConnectionPointAssemblyList shall be a collection of logicalConnectionPointAssembly objects. A logicalConnectionPointAssembly shall be a description of a group of logicalConnectionPoint objects. A logicalConnectionPointAssembly object has <<provides>> relations to logicalConnectionPoint objects.

6.5.1.3.4.10 MAUUsageList, MAUUsage

The MAUUsageList shall be a collection of MAUUsage objects. The MAUUsage objects shall define which MAU objects are used by the communicationEntity.

6.5.1.3.4.11 internalConnectionPointList, internalConnectionPoint

The internalConnectionPointList shall be a collection of internalConnectionPoint objects, which define internal connections between multiple communicationEntity and/or resourceEntity objects in the same device.

6.5.1.3.4.12 slotUsageList, slotUsage

The slotUsageList object shall be a collection of slotUsage objects. The slotUsage objects shall define, which slots are associated with the communicationEntity.

6.5.1.3.4.13 cfgItemList (configuration item list)

The cfgItemList may consist of dedicatedCfgItem objects and uncomittedCfgItem objects.

6.5.1.3.4.14 dedicatedCfgItem (dedicated configuration item)

A dedicatedCfgItem shall be a configuration Item with a dedicatedCfgItemType attribute as defined in Table E.4 and Table E.5. A dedicatedCfgItem shall be used to specify the corresponding configuration properties.

6.5.1.3.4.15 uncommittedCfgItem (uncommitted configuration item)

An uncommittedCfgItem shall be a configuration item without a dedicatedCfgItemType attribute. An uncommittedCfgItem shall be used to specify configuration properties that cannot be described by a dedicatedCfgItem.

NOTE The definition of uncommitted configuration items is outside the scope of this International Standard.

EXAMPLE A description of a DIP-Switch which changes the ID code of a device.

6.5.1.3.5 processingEntity

Figure 24 shows the definition of the processingEntity class.

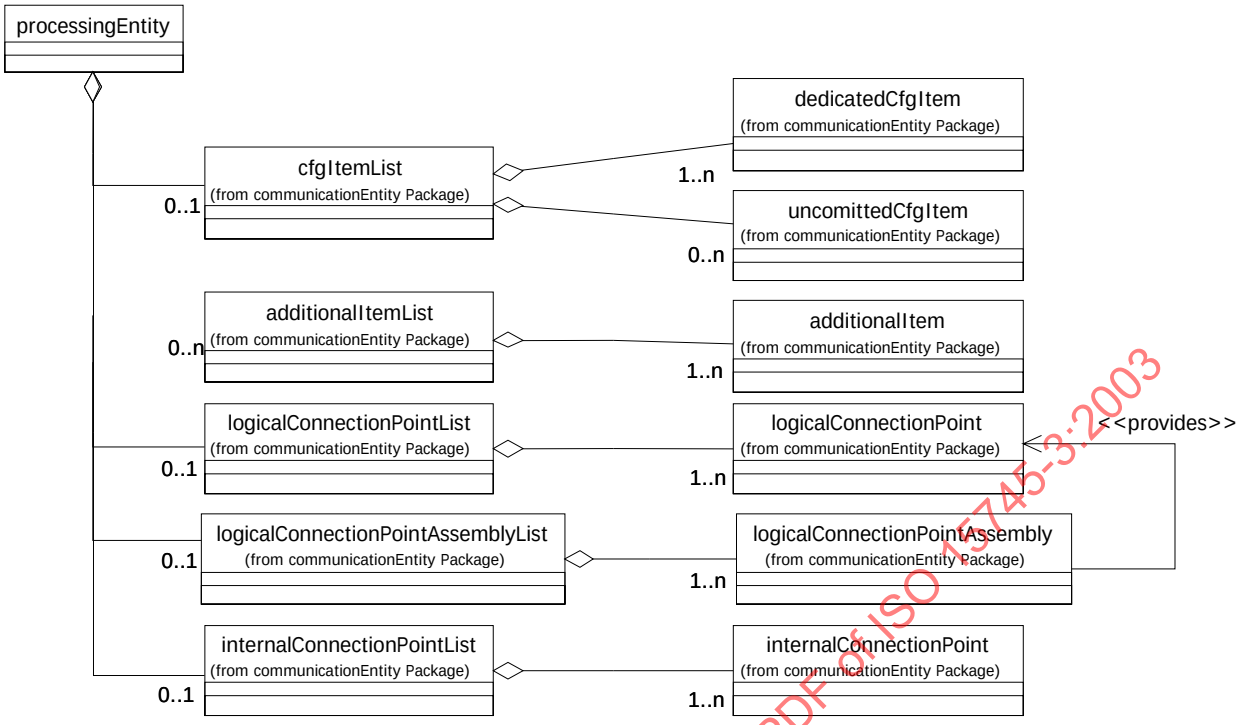

Figure 24 — INTERBUS processingEntity class diagram

A processingEntity shall describe any device entity that is not a communication entity.

NOTE For a description of the child objects see 6.5.1.3.4.

EXAMPLE A resource capable of executing programs.

6.5.1.4 Device function

6.5.1.4.1 Overview

The DeviceFunction is defined in Figure 25.

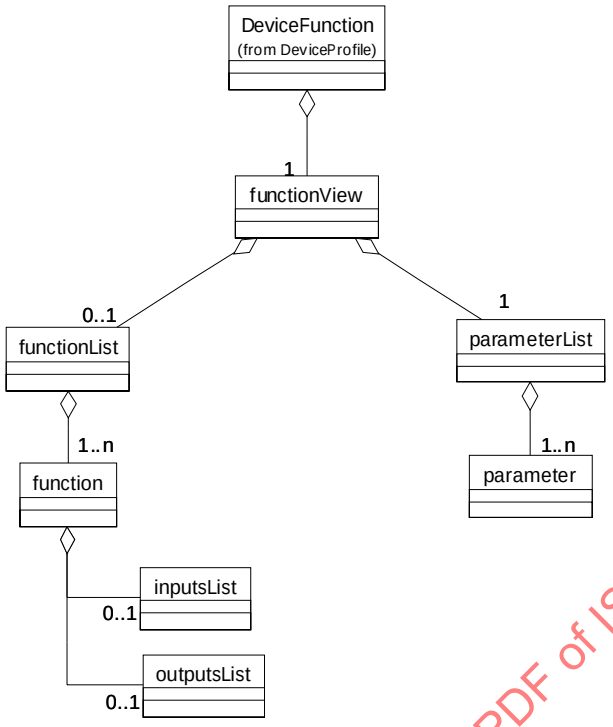

Figure 25 — INTERBUS DeviceFunction class diagram

To allow multiple representations of the device function, an additional XML schema is used to describe the DeviceFunction. The file name of this XML schema shall be "FDCMLISO15745DeviceFunction.XSD". The DeviceFunction XML schema is defined in E.1.6.3.

NOTE The definition of additional XML schemas describing the DeviceFunction is outside the scope of this International Standard.

6.5.1.4.2 parameterList, parameter

The parameterList shall be a collection of parameter objects. A parameter object describes a device parameter from a functional perspective. It is connected with a communication object in the communicationEntity.

6.5.1.4.3 functionList, function, inputsList, outputsList

The functionList shall be a collection of function objects. A function object shall consist of an inputList and an outputList. These lists shall contain a list of references to parameter objects.

6.5.1.5 Application process

The ApplicationProcess may be represented by one or more suitable XML schemas.

NOTE These XML schemas are not defined in ISO 15745-3.

6.5.2 Communication network profile

6.5.2.1 Class structure

Figure 26 shows the class structure of the INTERBUS communication network profile.

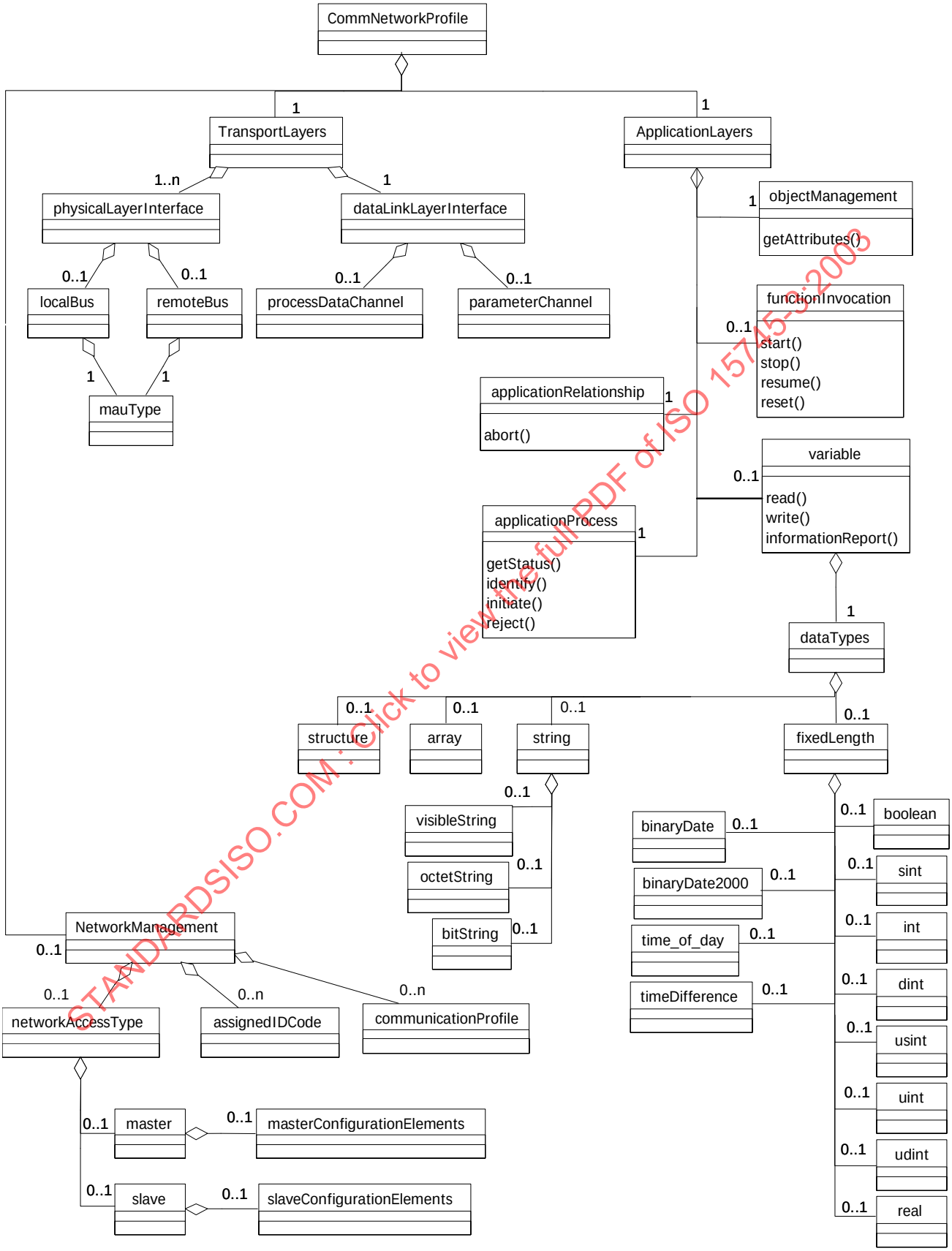

Figure 26 — INTERBUS communication network profile class diagram

The XML schema representing the INTERBUS communication network profile is defined in E.2. The file name of the XML schema shall be "INTERBUSCommNetworkProfile.XSD".

6.5.2.2 Transport layers

6.5.2.2.1 General

A TransportLayers object shall represent the combined profiles for the lower 4 OSI layers of the communication network integration model. The TransportLayers object shall be divided into one or more physicalLayerInterface objects and a dataLinkLayerInterface object.

6.5.2.2.2 physicalLayerInterface

The physicalLayerInterface profile object identifies the mauType, the interface type and supported baud rates of the physical layer interface. Valid interface types are localBus and remoteBus.

6.5.2.2.3 dataLinkLayerInterface

6.5.2.2.3.1 processDataChannel

The processDataChannel profile object shall state the length of the process data channel in bits.

6.5.2.2.3.2 parameterChannel

The parameterChannel profile class shall state the length of the parameter channel in octets.

6.5.2.3 Application layers

An ApplicationLayers class shall represent the combined profiles for the upper 3 OSI layers of the communication network integration model. It states the supported application service elements and their associated services.

The ASEs (Application Service Elements) defined in IEC 61158-5:2003, 13.2 can be stated by the following profile objects:

— applicationProcess

— applicationRelationship

— objectManagement

— functionInvocation

— variable

For the variable ASE, the selectable data types are shown in Figure 26. Data types are defined in IEC 61158-5:2003, clause 5.

6.5.2.4 Network management

6.5.2.4.1 networkAccessType, assignedIDCode

The NetworkManagement class shall identify the functionality for the configuration of a particular INTERBUS network. It contains the ID codes - least significant octet of an INTERBUS device code - assigned to a particular communication network profile, and the networkAccessType object, which contains:

— either the master profile for the INTERBUS master functionality in terms of elements (masterConfigurationElements object) necessary to configure an INTERBUS master;

— or the slave profile for the INTERBUS slave functionality in terms of elements (slaveConfigurationElements object) necessary to configure an INTERBUS slave.

NOTE An INTERBUS slave is a device that accesses the medium only after it has been initiated by the preceding slave or master. This includes remote bus devices, local bus devices and bus coupler.

6.5.2.4.2 communicationProfile

The communicationProfile shall state the usable communication profile identifiers (see 6.5.2.4.3).

6.5.2.4.3 Communication profile identifier

Communication profile identifiers are defined in IEC 61784-1:2003 subclause 10.1. An AIP Designer may specify additional communication profiles, the identifiers for such new communication profiles shall be a three digit number between 680 and 699.

Annex A
(normative)

ControlNet profile templates

A.1 General

The upper layers of the ControlNet network are based on the Common Industrial Protocol (CIP). This protocol models all communication and application entities as objects. CIP specific messaging requests services to be performed on corresponding object instances (or their attributes). This scheme provides an explicit access to all configuration, status, and runtime variables data in a node. At the same time, I/O connections allow direct exchange with the I/O database, without intermediate processing. In both cases, all data references within a device are specified using a CIP path, i.e. an octet string stream that defines the application object instance, attribute and/or connection end-point.

Multiple options are available for remote configuration of devices with a CIP communication interface, including:

— device information saved in printed or electronic format;

— dedicated Parameter Objects, which provide a known public interface to individual configuration/parameter data values, and may also embed additional configuration information such as descriptive text, data type, data limits and default;

— dedicated Configuration Assembly, which allows bulk upload and download of configuration data by grouping individual configuration/parameter data values;

— combinations of the above methods.

Configuration tools currently available for CIP-based devices use a specially formatted ASCII file, referred to as the Electronic Data Sheet (EDS), which provides:

— information needed to identify the connected device;

— a description of device data that can be accessed via the network (e.g. configurable parameters);

— a description of the communication capabilities supported by the device (e.g. connections);

— additional vendor-specific information.

The EDS allows a configuration tool to automate the device configuration process. The EDS requirements provide an open, consistent and compatible approach for performing device configuration in the CIP environment.

The EDS information is very similar to the information required in both communication network and device profiles, hence the following subclauses specify format for:

— communication network and device profile templates, as defined in ISO 15745-1;

— encapsulation of legacy EDS files in the ISO 15745 templates (“wrappers”);

— the legacy Electronic Data Sheet, including common semantics information.

NOTE The ControlNet EDS (Electronic Data Sheet) of a given device can be derived from the contents of the corresponding XML device and communication network profile files, using the appropriate style sheets.

A.2 Device profile template description

A.2.1 Device profile template description – XML based

A.2.1.1 General

The device profile XML files shall comply with the device profile XML schema as specified in A.2.1.3.3.

Contents of this XML schema are derived from the device profile class diagrams shown in 6.1.1, and extended with additional elements to allow full description of device requirements or capabilities.

A.2.1.2 Semantics of XML schema elements

A.2.1.2.1 ProfileBody

This main element is associated with a set of attributes which provide additional information about the profile file.

The semantics of these attributes are specified in A.4.1.4.2.

A.2.1.2.2 DeviceIdentity

This element specifies the supported instance attributes and operations of the Identity Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)), together with additional information for full device identification. When appropriate, it also indicates the actual values of the instance attributes.

The semantics of the DeviceIdentity_InstanceAttributes sub-elements of the DeviceIdentity element are specified in Table A.1.

Table A.1 — DeviceIdentity_InstanceAttributes elements

XML schema elements	Object Attributes	Semantics
SpecificationConformance	No	String specifying the reference version of the ControlNet specifications
VendCode, ProdType, ProdCode, ProdRevision	Yes	See A.4.1.4.3
VendName, ProdTypeStr, ProdName, Catalog, Icon, ExcludeFromAdapterRackConnection	No	See A.4.1.4.3
Status, SerialNumber	Yes	Not applicable
State, ConfigurationConsistencyValue, HeartbeatInterval	Yes	Not applicable
DeviceClassification	No	See A.4.1.4.4 and A.4.2.2.1

A.2.1.2.3 DeviceManager

This element specifies the supported class attributes and operations of the Identity Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)), together with additional information for device management. When appropriate, it also indicates the actual values of the instance attributes.

The semantics of the Modular sub-element of the DeviceManager element are specified in A.4.1.5.2.

A.2.1.2.4 DeviceFunction

The contents of this element are not detailed in this document.

A.2.1.2.5 ApplicationProcess

A.2.1.2.5.1 Assembly

This element specifies the supported class and instance attributes and operations of the Assembly Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)), together with a description of the individual instances.

The semantics of the Assem, ProxyAssem and ProxiedAssem sub-elements of the Assembly element are specified in A.4.1.4.8 and A.4.1.5.3.2.

A.2.1.2.5.2 Parameter

This element specifies the supported class and instance attributes and operations of the Parameter Object, together with a description of the individual instances.

The semantics of the Parameter_ClassAttributes sub-element of the Parameter element are specified in A.4.1.4.5.

The semantics of the Param, ProxyParam and ProxiedParam sub-elements of the Parameter element are specified in A.4.1.4.6 and A.4.1.5.3.1.

A.2.1.2.5.3 ParameterGroup

This element specifies groups of related parameters for a specific purpose.

The semantics of the Group sub-element of the ParameterGroup element is specified in A.4.1.4.7.

A.2.1.3 XML schemas

A.2.1.3.1 MasterTemplateTypes.xsd

NOTE This XML schema contains all the styles defined as part of the master template in ISO 15745-1:2003.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<!-- Target namespaces are not specified in this master template -->

<xsd:annotation>
   <xsd:documentation>* HEADER DATA TYPES *</xsd:documentation>
</xsd:annotation>

<xsd:simpleType name="ProfileClassID_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AIP" />
      <xsd:enumeration value="Process" />
      <xsd:enumeration value="InformationExchange" />
      <xsd:enumeration value="Resource" />
      <xsd:enumeration value="Device" />
      <xsd:enumeration value="CommunicationNetwork" />
      <xsd:enumeration value="Equipment" />
      <xsd:enumeration value="Human" />
      <xsd:enumeration value="Material" />
   </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="ISO15745Reference_DataType">
   <xsd:sequence>
      <xsd:element name="ISO15745Part" type="xsd:positiveInteger" />
      <xsd:element name="ISO15745Edition" type="xsd:positiveInteger" />
      <xsd:element name="ProfileTechnology" type="xsd:string" />
```

```
   </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="IASInterface_DataType">
   <xsd:union>
      <xsd:simpleType>
         <xsd:restriction base="xsd:string">
            <xsd:enumeration value="CSI" />
            <xsd:enumeration value="HCI" />
            <xsd:enumeration value="ISI" />
            <xsd:enumeration value="API" />
            <xsd:enumeration value="CMI" />
            <xsd:enumeration value="ESI" />
            <xsd:enumeration value="FSI" />
            <xsd:enumeration value="MTI" />
            <xsd:enumeration value="SEI" />
            <xsd:enumeration value="USI" />
         </xsd:restriction>
      </xsd:simpleType>
      <xsd:simpleType>
         <xsd:restriction base="xsd:string">
            <xsd:length value="4" />
         </xsd:restriction>
      </xsd:simpleType>
   </xsd:union>
</xsd:simpleType>

<xsd:annotation>
   <xsd:documentation>* ISO 15745 DEFINED DATA TYPES *</xsd:documentation>
</xsd:annotation>

<xsd:complexType name="ProfileHandle_DataType">
   <xsd:sequence>
      <xsd:element name="ProfileIdentification" type="xsd:string" />
      <xsd:element name="ProfileRevision" type="xsd:string" />
      <xsd:element name="ProfileLocation" type="xsd:anyURI" minOccurs="0" maxOccurs="1" />
   </xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

A.2.1.3.2 CIPDataTypes.xsd

NOTE This XML schema defines the XML schema items (e.g. data types, element types, attribute groups) used in the other XML schemas.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <!-- Target namespaces are not specified in this master template -->
   <xsd:annotation>
      <xsd:documentation>* CIP DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:simpleType name="dt_USINT">
      <xsd:restriction base="xsd:unsignedByte"/>
   </xsd:simpleType>
   <xsd:simpleType name="dt_UINT">
      <xsd:restriction base="xsd:unsignedShort"/>
   </xsd:simpleType>
   <xsd:simpleType name="dt_UDINT">
      <xsd:restriction base="xsd:unsignedInt"/>
   </xsd:simpleType>
   <xsd:simpleType name="dt_ULINT">
      <xsd:restriction base="xsd:unsignedLong"/>
   </xsd:simpleType>
   <xsd:simpleType name="dt_SINT">
      <xsd:restriction base="xsd:byte"/>
   </xsd:simpleType>
   <xsd:simpleType name="dt_INT">
      <xsd:restriction base="xsd:short"/>
   </xsd:simpleType>
   <xsd:simpleType name="dt_DINT">
      <xsd:restriction base="xsd:int"/>
   </xsd:simpleType>
```

```
<xsd:simpleType name="dt_LINT">
   <xsd:restriction base="xsd:long"/>
</xsd:simpleType>
<xsd:simpleType name="dt_BYTE">
   <xsd:restriction base="xsd:hexBinary">
      <xsd:maxLength value="1"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="dt_WORD">
   <xsd:restriction base="xsd:hexBinary">
      <xsd:maxLength value="2"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="dt_DWORD">
   <xsd:restriction base="xsd:hexBinary">
      <xsd:maxLength value="4"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="dt_LWORD">
   <xsd:restriction base="xsd:hexBinary">
      <xsd:maxLength value="8"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="dt_REAL">
   <xsd:restriction base="xsd:float"/>
</xsd:simpleType>
<xsd:simpleType name="dt_LREAL">
   <xsd:restriction base="xsd:double"/>
</xsd:simpleType>
<xsd:simpleType name="dt_EDS_Char_Array">
   <xsd:restriction base="xsd:string"/>
</xsd:simpleType>
<xsd:simpleType name="dt_EPATH">
   <xsd:list itemType="et_EPATH_item"/>
</xsd:simpleType>
<xsd:simpleType name="dt_STRINGI">
   <xsd:restriction base="xsd:string"/>
</xsd:simpleType>
<xsd:simpleType name="dt_EDS_Date">
   <xsd:restriction base="xsd:date"/>
</xsd:simpleType>
<xsd:simpleType name="dt_EDS_Time_Of_Day">
   <xsd:restriction base="xsd:time"/>
</xsd:simpleType>
<xsd:simpleType name="dt_EDS_Revision">
   <xsd:restriction base="xsd:string">
      <xsd:pattern value="[0-9]\.[1-9]|[1-9]\.[0-9]|[1-9]\.[1-9]"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="dt_EDS_URL">
   <xsd:restriction base="xsd:anyURI">
      <xsd:pattern value="http://.*"/>
      <xsd:pattern value="ftp://.*"/>
      <xsd:pattern value=".*"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="at_AccessType_OptionalGet">
   <xsd:restriction base="xsd:NMTOKEN">
      <xsd:enumeration value="None"/>
      <xsd:enumeration value="Get"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="at_AccessType_OptionalSet">
   <xsd:restriction base="xsd:NMTOKEN">
      <xsd:enumeration value="None"/>
      <xsd:enumeration value="Get"/>
      <xsd:enumeration value="Set"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="at_AccessType_Mandatory">
   <xsd:restriction base="xsd:NMTOKEN">
      <xsd:enumeration value="Get"/>
      <xsd:enumeration value="Set"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="et_VendorSpecificKeyword">
```

```
      <xsd:restriction base="xsd:string">
         <xsd:pattern value="[1-9][0-9]{0,4}_([A-Z]|[a-z]|[0-9])([A-Z]|[a-z]|[0-9]|[_])*"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:simpleType name="et_EPATH_item">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:pattern value="([0-9]|[a-f]|[A-F]){2}"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="SLOT"/>
               <xsd:enumeration value="SLOT_MINUS_ONE"/>
               <xsd:enumeration value="SYMBOL_ANSI"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:pattern value="Param[1-9][0-9]{0,4}"/>
               <xsd:pattern value="\[Param[1-9][0-9]{0,4}\]"/>
               <xsd:pattern value="ProxyParam[1-9][0-9]{0,4}"/>
               <xsd:pattern value="\[ProxyParam[1-9][0-9]{0,4}\]"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
   <xsd:simpleType name="et_ParamReference">
      <xsd:restriction base="xsd:NMTOKEN">
         <xsd:pattern value="Param[1-9][0-9]{0,4}([:][0-9]{1,2})*"/>
         <xsd:pattern value="ProxyParam[1-9][0-9]{0,4}([:][0-9]{1,2})*"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:simpleType name="et_AssemReference">
      <xsd:restriction base="xsd:NMTOKEN">
         <xsd:pattern value="Assem[1-9][0-9]{0,4}"/>
         <xsd:pattern value="ProxyAssem[1-9][0-9]{0,4}"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:attributeGroup name="ag_FileDescription">
      <xsd:attribute name="DescText" type="dt_EDS_Char_Array" use="required"/>
      <xsd:attribute name="CreateDate" type="dt_EDS_Date" use="required"/>
      <xsd:attribute name="CreateTime" type="dt_EDS_Time_Of_Day" use="required"/>
      <xsd:attribute name="ModDate" type="dt_EDS_Date" use="optional"/>
      <xsd:attribute name="ModTime" type="dt_EDS_Time_Of_Day" use="optional"/>
      <xsd:attribute name="Revision" type="dt_EDS_Revision" use="required"/>
      <xsd:attribute name="HomeURL" type="dt_EDS_URL" use="optional"/>
      <xsd:attribute name="SpecificationConformance" type="dt_EDS_Char_Array" use="required"/>
   </xsd:attributeGroup>
</xsd:schema>
```

A.2.1.3.3 CIP_Device_Profile.xsd

NOTE This XML schema includes the files “MasterTemplateTypes.xsd” (see A.2.1.3.1) and “CIPDataTypes.xsd” (see A.2.1.3.2).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <!-- Target namespaces are not specified in this master template -->
   <xsd:redefine schemaLocation="MasterTemplateTypes.xsd">
      <xsd:complexType name="ISO15745Reference_DataType">
         <xsd:complexContent>
            <xsd:restriction base="ISO15745Reference_DataType">
               <xsd:sequence>
                  <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
                  <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
                  <xsd:element name="ProfileTechnology" type="xsd:string" fixed="CIP"/>
               </xsd:sequence>
            </xsd:restriction>
         </xsd:complexContent>
      </xsd:complexType>
   </xsd:redefine>
```

```
   <xsd:include schemaLocation="CIPDataTypes.xsd"/>
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType" fixed="Device"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="DeviceIdentity"/>
            <xsd:element ref="DeviceManager" minOccurs="0"/>
            <xsd:element ref="DeviceFunction" maxOccurs="unbounded"/>
            <xsd:element ref="ApplicationProcess" minOccurs="0"/>
            <xsd:element name="ExternalProfileHandle" type="ProfileHandle_DataType" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attributeGroup ref="ag_FileDescription"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DeviceIdentity">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="DeviceIdentity_InstanceAttributes">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="VendCode">
                        <xsd:complexType>
                           <xsd:simpleContent>
                              <xsd:extension base="dt_UINT">
                                 <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                              </xsd:extension>
                           </xsd:simpleContent>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="VendName">
                        <xsd:complexType>
                           <xsd:simpleContent>
                              <xsd:extension base="dt_EDS_Char_Array"/>
                           </xsd:simpleContent>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="SpecificationConformance" type="dt_EDS_Char_Array"
minOccurs="0"/>
                     <xsd:element name="ProdType">
                        <xsd:complexType>
                           <xsd:simpleContent>
                              <xsd:extension base="dt_UINT">
                                 <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                              </xsd:extension>
                           </xsd:simpleContent>
```

```
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="ProdTypeStr">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="dt_EDS_Char_Array"/>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="ProdCode">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="dt_UINT">
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                                </xsd:extension>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="ProdRevision">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="MajRev" type="dt_USINT"/>
                                <xsd:element name="MinRev" type="dt_USINT"/>
                            </xsd:sequence>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Status" minOccurs="0">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="SerialNumber" minOccurs="0">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="ProdName">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="xsd:string">
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                                </xsd:extension>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="State" minOccurs="0">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="ConfigurationConsistencyValue" minOccurs="0">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="HeartbeatInterval" minOccurs="0">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Catalog" type="xsd:string" minOccurs="0"/>
                    <xsd:element name="Icon" type="xsd:string" minOccurs="0"/>
                    <xsd:element name="ExcludeFromAdapterRackConnection" type="xsd:string"
minOccurs="0"/>
                    <xsd:element name="DeviceClassification" minOccurs="0">
                        <xsd:complexType>
                            <xsd:sequence>
```

```
                              <xsd:element name="Class" maxOccurs="unbounded">
                                 <xsd:complexType>
                                    <xsd:sequence>
                                       <xsd:element name="MainClass">
                                          <xsd:simpleType>
                                             <xsd:union>
                                                <xsd:simpleType>
                                                   <xsd:restriction base="xsd:NMTOKEN">
                                                      <xsd:enumeration value="ControlNet"/>
                                                      <xsd:enumeration value="DeviceNet"/>
                                                      <xsd:enumeration value="EtherNetIP"/>
                                                   </xsd:restriction>
                                                </xsd:simpleType>
                                                <xsd:simpleType>
                                                   <xsd:restriction base="et_VendorSpecificKeyword"/>
                                                </xsd:simpleType>
                                             </xsd:union>
                                          </xsd:simpleType>
                                       </xsd:element>
                                       <xsd:element name="SubClass" type="xsd:NMTOKEN" minOccurs="0"
maxOccurs="unbounded"/>
                                    </xsd:sequence>
                                    <xsd:attribute name="id" use="required">
                                       <xsd:simpleType>
                                          <xsd:restriction base="xsd:ID">
                                             <xsd:pattern value="Class[1-9][0-9]{0,4}"/>
                                          </xsd:restriction>
                                       </xsd:simpleType>
                                    </xsd:attribute>
                                 </xsd:complexType>
                              </xsd:element>
                           </xsd:sequence>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="DeviceIdentity_InstanceOperations" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Get_Attribute_All">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Reset">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Get_Attribute_Single">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DeviceManager">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="DeviceIdentity_ClassAttributes" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="ObjectRevision">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
```

```
                        <xsd:element name="MaxInstance">
                           <xsd:complexType>
                              <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="MaxIDClassAttributes">
                           <xsd:complexType>
                              <xsd:simpleContent>
                                 <xsd:extension base="xsd:string">
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                                 </xsd:extension>
                              </xsd:simpleContent>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="MaxIDInstanceAttributes">
                           <xsd:complexType>
                              <xsd:simpleContent>
                                 <xsd:extension base="xsd:string">
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                                 </xsd:extension>
                              </xsd:simpleContent>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                     </xsd:sequence>
                  </xsd:complexType>
               </xsd:element>
               <xsd:element name="DeviceIdentity_ClassOperations" minOccurs="0">
                  <xsd:complexType>
                     <xsd:sequence>
                        <xsd:element name="Get_Attribute_All">
                           <xsd:complexType>
                              <xsd:attribute ref="SupportedService"/>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="Reset">
                           <xsd:complexType>
                              <xsd:attribute ref="SupportedService"/>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="Get_Attribute_Single">
                           <xsd:complexType>
                              <xsd:attribute ref="SupportedService"/>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="Find_Next_Object_Instance">
                           <xsd:complexType>
                              <xsd:attribute ref="SupportedService"/>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                     </xsd:sequence>
                  </xsd:complexType>
               </xsd:element>
               <xsd:element name="Modular" minOccurs="0">
                  <xsd:complexType>
                     <xsd:choice>
                        <xsd:element name="Chassis">
                           <xsd:complexType>
                              <xsd:sequence>
                                 <xsd:element name="DefineSlotsInRack" type="dt_UINT"/>
                                 <xsd:element name="SlotDisplayRule" type="et_ParamReference"
minOccurs="0"/>
                                 <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                              </xsd:sequence>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="Module">
                           <xsd:complexType>
                              <xsd:sequence>
                                 <xsd:element name="Width" type="dt_UINT"/>
                                 <xsd:element name="Rack" maxOccurs="unbounded">
                                    <xsd:complexType>
```

```
                                    <xsd:sequence>
                                       <xsd:element name="VendCode" type="dt_UINT"/>
                                       <xsd:element name="ProdType" type="dt_UINT"/>
                                       <xsd:element name="ProdCode" type="dt_UINT"/>
                                       <xsd:element name="MajRev" type="dt_USINT"/>
                                       <xsd:element name="MinRev" type="dt_USINT"/>
                                       <xsd:element name="LegalSlot" type="dt_UINT"
maxOccurs="unbounded"/>
                                       <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                                    </xsd:sequence>
                                    <xsd:attribute name="id" use="required">
                                       <xsd:simpleType>
                                          <xsd:restriction base="xsd:ID">
                                             <xsd:pattern value="Rack[1-9][0-9]{0,4}"/>
                                          </xsd:restriction>
                                       </xsd:simpleType>
                                    </xsd:attribute>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="ExternalID" type="dt_EPATH" minOccurs="0"/>
                              <xsd:element name="GenericID" type="dt_EPATH" minOccurs="0"/>
                              <xsd:element name="ExternIDExactMatch" minOccurs="0">
                                 <xsd:simpleType>
                                    <xsd:restriction base="xsd:NMTOKEN">
                                       <xsd:enumeration value="Yes"/>
                                       <xsd:enumeration value="No"/>
                                    </xsd:restriction>
                                 </xsd:simpleType>
                              </xsd:element>
                              <xsd:element name="Query" minOccurs="0">
                                 <xsd:complexType>
                                    <xsd:sequence>
                                       <xsd:element name="Path" type="dt_EPATH"/>
                                       <xsd:element name="Service" type="dt_USINT"/>
                                       <xsd:element name="Size">
                                          <xsd:simpleType>
                                             <xsd:restriction base="dt_USINT">
                                                <xsd:minInclusive value="1"/>
                                                <xsd:maxInclusive value="16"/>
                                             </xsd:restriction>
                                          </xsd:simpleType>
                                       </xsd:element>
                                       <xsd:element name="ExternalID" type="dt_EPATH"/>
                                    </xsd:sequence>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                           </xsd:sequence>
                        </xsd:complexType>
                     </xsd:element>
                  </xsd:choice>
               </xsd:complexType>
            </xsd:element>
            <xsd:any namespace="##any"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DeviceFunction">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ApplicationProcess">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Parameter" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Parameter_Class" minOccurs="0">
                        <xsd:complexType>
                           <xsd:sequence>
                              <xsd:element name="Parameter_ClassAttributes">
                                 <xsd:complexType>
                                    <xsd:sequence>
```

```
                                    <xsd:element name="ObjectRevision" minOccurs="0">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="MaxInstance">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="ParameterClassDescriptor">
                                       <xsd:complexType>
                                          <xsd:simpleContent>
                                             <xsd:extension base="xsd:string">
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                             </xsd:extension>
                                          </xsd:simpleContent>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="ConfigurationAssemblyInstance">
                                       <xsd:complexType>
                                          <xsd:simpleContent>
                                             <xsd:extension base="xsd:string">
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                             </xsd:extension>
                                          </xsd:simpleContent>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="NativeLanguage" minOccurs="0">
                                       <xsd:complexType>
                                          <xsd:simpleContent>
                                             <xsd:extension base="xsd:string">
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalSet" use="required"/>
                                             </xsd:extension>
                                          </xsd:simpleContent>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                                 </xsd:sequence>
                              </xsd:complexType>
                           </xsd:element>
                           <xsd:element name="Parameter_ClassOperations" minOccurs="0">
                              <xsd:complexType>
                                 <xsd:sequence>
                                    <xsd:element name="Get_Attribute_All">
                                       <xsd:complexType>
                                          <xsd:attribute ref="SupportedService"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="Reset">
                                       <xsd:complexType>
                                          <xsd:attribute ref="SupportedService"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="Get_Attribute_Single">
                                       <xsd:complexType>
                                          <xsd:attribute ref="SupportedService" fixed="true"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="Set_Attribute_Single">
                                       <xsd:complexType>
                                          <xsd:attribute ref="SupportedService"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="Restore">
                                       <xsd:complexType>
                                          <xsd:attribute ref="SupportedService"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="Save">
                                       <xsd:complexType>
```

```
                                        <xsd:attribute ref="SupportedService"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="Parameter_Instance" minOccurs="0">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Parameter_InstanceAttributes" minOccurs="0">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="ParameterValue">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Set"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="LinkPathSize">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Set"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="LinkPath">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Set"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="ParamDescriptor">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="DataType">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="DataSize">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="ParameterName">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="UnitsString">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="HelpString">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="MinimumValue">
                                    <xsd:complexType>
                                        <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                    </xsd:complexType>
                                </xsd:element>
```

```
                                    <xsd:element name="MaximumValue">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="DefaultValue">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="ScalingMultiplier">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="ScalingDivider">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="ScalingBase">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="ScalingOffset">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="MultiplierLink">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="DivisorLink">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="BaseLink">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="OffsetLink">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:element name="DecimalPrecision">
                                       <xsd:complexType>
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                       </xsd:complexType>
                                    </xsd:element>
                                    <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                                 </xsd:sequence>
                              </xsd:complexType>
                           </xsd:element>
                           <xsd:element name="Parameter_InstanceOperations" minOccurs="0">
                              <xsd:complexType>
                                 <xsd:sequence>
                                    <xsd:element name="Get_Attribute_All">
                                       <xsd:complexType>
                                          <xsd:attribute ref="SupportedService"/>
```

```
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Get_Attribute_Single">
                                    <xsd:complexType>
                                        <xsd:attribute ref="SupportedService" fixed="true"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Set_Attribute_Single">
                                    <xsd:complexType>
                                        <xsd:attribute ref="SupportedService" fixed="true"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Get_Enum_String">
                                    <xsd:complexType>
                                        <xsd:attribute ref="SupportedService"/>
                                    </xsd:complexType>
                                </xsd:element>
                                <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="Param" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:complexContent>
                    <xsd:extension base="et_ParamType">
                        <xsd:attribute name="id" use="required">
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:ID">
                                    <xsd:pattern value="Param[1-9][0-9]{0,4}"/>
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:attribute>
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ProxyParam" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:complexContent>
                    <xsd:extension base="et_ProxyParamType">
                        <xsd:attribute name="id" use="required">
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:ID">
                                    <xsd:pattern value="ProxyParam[1-9][0-9]{0,4}"/>
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:attribute>
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ProxiedParam" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:complexContent>
                    <xsd:extension base="et_ParamType">
                        <xsd:attribute name="id" use="required">
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:ID">
                                    <xsd:pattern value="ProxiedParam[1-9][0-9]{0,4}"/>
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:attribute>
                    </xsd:extension>
                </xsd:complexContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="Assembly" minOccurs="0">
    <xsd:complexType>
        <xsd:sequence>
```

```
                        <xsd:element name="Assembly_Class" minOccurs="0">
                           <xsd:complexType>
                              <xsd:sequence>
                                 <xsd:element name="Assembly_ClassAttributes" minOccurs="0">
                                    <xsd:complexType>
                                       <xsd:sequence>
                                          <xsd:element name="ObjectRevision">
                                             <xsd:complexType>
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:element name="MaxInstance">
                                             <xsd:complexType>
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                                       </xsd:sequence>
                                    </xsd:complexType>
                                 </xsd:element>
                                 <xsd:element name="Assembly_ClassOperations" minOccurs="0">
                                    <xsd:complexType>
                                       <xsd:sequence>
                                          <xsd:element name="Create">
                                             <xsd:complexType>
                                                <xsd:attribute ref="SupportedService"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:element name="Delete">
                                             <xsd:complexType>
                                                <xsd:attribute ref="SupportedService"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:element name="Get_Attribute_Single">
                                             <xsd:complexType>
                                                <xsd:attribute ref="SupportedService" fixed="true"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                                       </xsd:sequence>
                                    </xsd:complexType>
                                 </xsd:element>
                              </xsd:sequence>
                           </xsd:complexType>
                        </xsd:element>
                        <xsd:element name="Assembly_Instance" minOccurs="0">
                           <xsd:complexType>
                              <xsd:sequence>
                                 <xsd:element name="Assembly_InstanceAttributes" minOccurs="0">
                                    <xsd:complexType>
                                       <xsd:sequence>
                                          <xsd:element name="NumberOfMembers">
                                             <xsd:complexType>
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalGet" use="required"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:element name="AssemblyMemberList">
                                             <xsd:complexType>
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_OptionalSet" use="required"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:element name="AssemblyData">
                                             <xsd:complexType>
                                                <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Set"/>
                                             </xsd:complexType>
                                          </xsd:element>
                                          <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                                       </xsd:sequence>
                                    </xsd:complexType>
                                 </xsd:element>
                                 <xsd:element name="Assembly_InstanceOperations" minOccurs="0">
```

```
                        <xsd:complexType>
                           <xsd:sequence>
                              <xsd:element name="Delete">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Get_Attribute_Single">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Set_Attribute_Single">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Get_Member">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Set_Member">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Insert_Member">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Remove_Member">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                           </xsd:sequence>
                        </xsd:complexType>
                     </xsd:element>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Assem" minOccurs="0" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:complexContent>
                     <xsd:extension base="et_AssemType">
                        <xsd:attribute name="id" use="required">
                           <xsd:simpleType>
                              <xsd:restriction base="xsd:ID">
                                 <xsd:pattern value="Assem[1-9][0-9]{0,4}"/>
                              </xsd:restriction>
                           </xsd:simpleType>
                        </xsd:attribute>
                     </xsd:extension>
                  </xsd:complexContent>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="ProxyAssem" minOccurs="0" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:complexContent>
                     <xsd:extension base="et_AssemType">
                        <xsd:attribute name="id" use="required">
                           <xsd:simpleType>
                              <xsd:restriction base="xsd:ID">
                                 <xsd:pattern value="ProxyAssem[1-9][0-9]{0,4}"/>
                              </xsd:restriction>
                           </xsd:simpleType>
                        </xsd:attribute>
                     </xsd:extension>
                  </xsd:complexContent>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="ProxiedAssem" minOccurs="0" maxOccurs="unbounded">
               <xsd:complexType>
```

```
                        <xsd:complexContent>
                           <xsd:extension base="et_AssemType">
                              <xsd:attribute name="id" use="required">
                                 <xsd:simpleType>
                                    <xsd:restriction base="xsd:ID">
                                       <xsd:pattern value="ProxiedAssem[1-9][0-9]{0,4}"/>
                                    </xsd:restriction>
                                 </xsd:simpleType>
                              </xsd:attribute>
                           </xsd:extension>
                        </xsd:complexContent>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="ParameterGroup" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="Group" minOccurs="0" maxOccurs="unbounded">
                     <xsd:complexType>
                        <xsd:sequence>
                           <xsd:element name="NameString" type="dt_EDS_Char_Array"/>
                           <xsd:element name="NumberOfMembers" type="dt_UINT"/>
                           <xsd:choice maxOccurs="unbounded">
                              <xsd:element name="ParameterRef" type="dt_UINT"/>
                              <xsd:element name="VariantRef" type="xsd:NMTOKEN"/>
                              <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                           </xsd:choice>
                        </xsd:sequence>
                        <xsd:attribute name="id" use="required">
                           <xsd:simpleType>
                              <xsd:restriction base="xsd:ID">
                                 <xsd:pattern value="Group[1-9][0-9]{0,4}"/>
                              </xsd:restriction>
                           </xsd:simpleType>
                        </xsd:attribute>
                     </xsd:complexType>
                  </xsd:element>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:attribute name="SupportedService" use="required">
   <xsd:simpleType>
      <xsd:restriction base="xsd:boolean">
         <xsd:pattern value="true|false"/>
      </xsd:restriction>
   </xsd:simpleType>
</xsd:attribute>
<xsd:complexType name="et_ParamType">
   <xsd:sequence>
      <xsd:element name="LinkPathSize" type="dt_USINT" minOccurs="0"/>
      <xsd:element name="LinkPath" type="dt_EPATH" minOccurs="0"/>
      <xsd:element name="ParamDescriptor" type="dt_WORD"/>
      <xsd:element name="DataType">
         <xsd:simpleType>
            <xsd:union memberTypes="dt_USINT dt_EPATH"/>
         </xsd:simpleType>
      </xsd:element>
      <xsd:element name="DataSize" type="dt_USINT"/>
      <xsd:element name="ParameterName" type="dt_EDS_Char_Array"/>
      <xsd:element name="UnitsString" type="dt_EDS_Char_Array"/>
      <xsd:element name="HelpString" type="dt_EDS_Char_Array"/>
      <xsd:element name="MinimumValue" minOccurs="0"/>
      <xsd:element name="MaximumValue" minOccurs="0"/>
      <xsd:element name="DefaultValue" minOccurs="0"/>
      <xsd:element name="ScalingMultiplier" type="dt_UINT" minOccurs="0"/>
      <xsd:element name="ScalingDivider" type="dt_UINT" minOccurs="0"/>
      <xsd:element name="ScalingBase" type="dt_UINT" minOccurs="0"/>
      <xsd:element name="ScalingOffset" type="dt_INT" minOccurs="0"/>
      <xsd:element name="MultiplierLink" type="dt_UINT" minOccurs="0"/>
```

```
         <xsd:element name="DivisorLink" type="dt_UINT" minOccurs="0"/>
         <xsd:element name="BaseLink" type="dt_UINT" minOccurs="0"/>
         <xsd:element name="OffsetLink" type="dt_INT" minOccurs="0"/>
         <xsd:element name="DecimalPrecision" type="dt_USINT" minOccurs="0"/>
         <xsd:element name="InternationalParameterName" type="dt_EDS_Char_Array" minOccurs="0"/>
         <xsd:element name="InternationalEngineeringUnits" type="dt_EDS_Char_Array" minOccurs="0"/>
         <xsd:element name="InternationalHelpString" type="dt_EDS_Char_Array" minOccurs="0"/>
         <xsd:element name="Enum" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence maxOccurs="unbounded">
                  <xsd:element name="EnumValue" type="dt_LINT"/>
                  <xsd:element name="EnumName" type="dt_EDS_Char_Array"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="et_ProxyParamType">
      <xsd:complexContent>
         <xsd:extension base="et_ParamType">
            <xsd:sequence>
               <xsd:element name="ProxyParamSizeAdder" minOccurs="0">
                  <xsd:complexType>
                     <xsd:sequence>
                        <xsd:element name="MinimumValue"/>
                        <xsd:element name="MaximumValue"/>
                        <xsd:element name="DefaultValue"/>
                     </xsd:sequence>
                  </xsd:complexType>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="et_AssemType">
      <xsd:sequence>
         <xsd:element name="AssemblyName" type="dt_EDS_Char_Array" minOccurs="0"/>
         <xsd:element name="AssemblyPath" type="dt_EPATH" minOccurs="0"/>
         <xsd:element name="AssemblyDataSize" type="dt_UINT" minOccurs="0"/>
         <xsd:element name="AssemblyDescriptor" type="dt_WORD" minOccurs="0"/>
         <xsd:element name="AssemblyMember" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
               <xsd:choice>
                  <xsd:element name="MemberSize" type="dt_UINT"/>
                  <xsd:element name="MemberReference" type="et_MemberReferenceType"/>
                  <xsd:element name="VariantReference">
                     <xsd:complexType/>
                  </xsd:element>
                  <xsd:choice>
                     <xsd:sequence>
                        <xsd:element name="MemberSize" type="dt_UINT"/>
                        <xsd:element name="MemberReference" type="et_MemberReferenceType"/>
                     </xsd:sequence>
                     <xsd:sequence>
                        <xsd:element name="MemberSize" type="dt_UINT"/>
                        <xsd:element name="VariantReference">
                           <xsd:complexType/>
                        </xsd:element>
                     </xsd:sequence>
                  </xsd:choice>
               </xsd:choice>
            </xsd:complexType>
         </xsd:element>
         <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="et_MemberReferenceType">
      <xsd:union memberTypes="et_AssemReference et_ParamReference dt_UDINT dt_EPATH xsd:NMTOKEN"/>
   </xsd:simpleType>
</xsd:schema>
```

A.2.2 Device profile template description – XML encapsulation of EDS files

A.2.2.1 General

The device profile XML files used to encapsulate EDS files shall comply with the device profile XML schema as specified in A.2.2.2.

The semantics of the sub-elements of the ExternalProfileHandle element, used to reference an existing EDS file, are specified in Table A.2. Depending on the value of the attribute WrapperReference, the EDS file will be referenced using either identification elements from the EDS file itself, or from the product described by this EDS.

NOTE 1 Choice of relevant identification elements will depend upon the expected usage of the wrapper file.

Table A.2 — ExternalProfileHandle elements

XML schema elements	WrapperReference = FILEINFO	WrapperReference =DEVICEINFO
ProfileIdentification	EDS File description text[a]	VendorID, Device Type, Product Code [b]
ProfileRevision	EDS Revision[a]	Product Revision [b]
ProfileLocation	EDS HomeURL[a]	Icon File Name[b]
[a] See A.4.1.4.2 for more details [b] See A.4.1.4.3 for more details		

If present, the DeviceIdentity, DeviceManager, DeviceFunction and ApplicationProcess elements should be compatible with the formats specified in A.2.1.3.3.

NOTE 2 This may be used during a transition phase between the legacy EDS format and the full XML format.

A.2.2.2 XML schema : EDS_Device_Profile_wrapper.xsd

NOTE This XML schema includes the file “MasterTemplateTypes.xsd” (see A.2.1.3.1).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <!-- Target namespaces are not specified in this master template -->
   <xsd:redefine schemaLocation="MasterTemplateTypes.xsd">
      <xsd:complexType name="ISO15745Reference_DataType">
         <xsd:complexContent>
            <xsd:restriction base="ISO15745Reference_DataType">
               <xsd:sequence>
                  <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
                  <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
                  <xsd:element name="ProfileTechnology" type="xsd:string" fixed="EDS"/>
               </xsd:sequence>
            </xsd:restriction>
         </xsd:complexContent>
      </xsd:complexType>
   </xsd:redefine>
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
```

```
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType" fixed="Device"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="DeviceIdentity" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="DeviceManager" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="DeviceFunction" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="ApplicationProcess" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="ExternalProfileHandle">
               <xsd:complexType>
                  <xsd:complexContent>
                     <xsd:extension base="ProfileHandle_DataType">
                        <xsd:attribute name="WrapperReference" use="optional" default="FILEINFO">
                           <xsd:simpleType>
                              <xsd:restriction base="xsd:NMTOKEN">
                                 <xsd:enumeration value="FILEINFO"/>
                                 <xsd:enumeration value="DEVICEINFO"/>
                              </xsd:restriction>
                           </xsd:simpleType>
                        </xsd:attribute>
                     </xsd:extension>
                  </xsd:complexContent>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

A.3 Communication network profile template description

A.3.1 Communication network profile template description – XML based

A.3.1.1 General

The communication network profile XML files shall comply with the communication network profile XML schema as specified in A.3.1.3.

Contents of this XML schema are derived from the communication network profile class diagrams shown in 6.1.2, and extended with additional elements to allow full description of communication network requirements or capabilities.

A.3.1.2 Semantics of XML schema elements

A.3.1.2.1 ProfileBody

This main element is associated with a set of attributes which provide additional information about the profile file.

The semantics of these attributes is specified in A.4.1.4.2.

A.3.1.2.2 ApplicationLayers

A.3.1.2.2.1 ConnectionManager

This element specifies the supported instance attributes and operations of the Connection Manager Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)), together with a description of the individual connection instances.

The semantics of the Connection, ProxyConnect and ProxiedConnect sub-elements of the ConnectionDescriptions element are specified in A.4.1.4.9 and A.4.1.5.3.3.

A.3.1.2.2.2 MessageRouter

This element specifies the supported instance attributes and operations of the Message Router Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)).

A.3.1.2.3 TransportLayers

A.3.1.2.3.1 CNPhysicalLayer

This element identifies the physical layer characteristics (e.g. connectors, delays).

The semantics of its sub-elements are specified in Table A.3.

Table A.3 — CNPhysicalLayer elements

XML schema elements	XML schema attribute	Semantics
Connectors	Media	Specifies whether the device has a redundant media
	NetworkAccessPort	Specifies whether the device has a Network Access Port
Delay1	Not applicable	See A.4.2.4.1

A.3.1.2.3.2 CNLinkLayer

This element defines some properties associated with data link layer configuration.

The semantics of its sub-elements are specified in Table A.4.

Table A.4 — CNLinkLayer elements

XML schema elements	XML schema attribute	Semantics
Mac-IDSetting		Specifies the minimum, maximum and default MAC-ID
	SwitchType	Specifies avalaible hardware switches for setting of MAC-ID
	SoftwareSettable	Specifies whether MAC-ID may be set via software
Capacity	Not applicable	See A.4.2.4.4

A.3.1.2.3.3 ControlNetObject

This element specifies the supported instance attributes and operations of the ControlNet Object (see IEC 61158-4:2003 (Type 2)).

A.3.1.2.3.4 Ports

This element identifies the device ports which are able to route messages from one link to another link.

The semantics of the Port sub-element of the Ports element are specified in A.4.1.4.10 and A.4.2.2.2.

A.3.1.2.4 NetworkManagement

A.3.1.2.4.1 NM-ControlNetObject

This element specifies the supported class attributes and operations of the ControlNet Object (see IEC 61158-4:2003 (Type 2)).

A.3.1.2.4.2 NM-ConnectionManager

This element specifies the supported class attributes and operations of the Connection Manager Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)).

A.3.1.2.4.3 NM-MessageRouter

This element specifies the supported class attributes and operations of the Message Router Object (see IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2)).

A.3.1.2.4.4 Keeper

This element specifies the supported class and instance attributes and operations of the Keeper Object (see IEC 61158-4:2003 (Type 2)).

A.3.1.2.4.5 ControlNetScheduling

This element specifies the supported class and instance attributes and operations of the ControlNet Scheduling Object (see IEC 61158-4:2003 (Type 2)).

A.3.1.3 XML schema : CNet_CommNet_Profile.xsd

NOTE This XML schema includes the files “MasterTemplateTypes.xsd” (see A.2.1.3.1) and “CIPDataTypes.xsd” (see A.2.1.3.2).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <!-- Target namespaces are not specified in this master template -->
   <xsd:redefine schemaLocation="MasterTemplateTypes.xsd">
      <xsd:complexType name="ISO15745Reference_DataType">
         <xsd:complexContent>
            <xsd:restriction base="ISO15745Reference_DataType">
               <xsd:sequence>
                  <xsd:element name="ISO15745Part" type="xsd:positiveInteger" fixed="3"/>
                  <xsd:element name="ISO15745Edition" type="xsd:positiveInteger" fixed="1"/>
                  <xsd:element name="ProfileTechnology" type="xsd:string" fixed="ControlNet"/>
               </xsd:sequence>
            </xsd:restriction>
         </xsd:complexContent>
      </xsd:complexType>
   </xsd:redefine>
   <xsd:include schemaLocation="CIPDataTypes.xsd"/>
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"
fixed="CommunicationNetwork"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" fixed="CSI"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ApplicationLayers"/>
            <xsd:element ref="TransportLayers"/>
            <xsd:element ref="NetworkManagement" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attributeGroup ref="ag_FileDescription"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ApplicationLayers">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ConnectionManager"/>
            <xsd:element ref="MessageRouter"/>
            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransportLayers">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="CNPhysicalLayer"/>
            <xsd:element ref="CNLinkLayer"/>
```

```
            <xsd:element ref="ControlNetObject"/>
            <xsd:element ref="Ports" minOccurs="0"/>
            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="NetworkManagement">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="NM-ControlNetObject" minOccurs="0"/>
            <xsd:element ref="NM-ConnectionManager" minOccurs="0"/>
            <xsd:element ref="NM-MessageRouter" minOccurs="0"/>
            <xsd:element ref="Keeper" minOccurs="0"/>
            <xsd:element ref="ControlNetScheduling" minOccurs="0"/>
            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ConnectionManager">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ConnectionManager_InstanceAttributes" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="OpenReqs">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="OpenFormatRejects">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="OpenResourceRejects">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="OpenOtherRejects">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="CloseReqs">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="CloseFormatRejects">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="CloseOtherRejects">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="ConnTimeouts">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="NumConnEntries">
                        <xsd:complexType>
                           <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
```

```
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="ConnOpenBits">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="CpuUtilization">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="MaxBuffSize">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="BufSizeRemaining">
                        <xsd:complexType>
                            <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ConnectionManager_InstanceOperations" minOccurs="0">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Get_Attribute_All">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Set_Attribute_All">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Get_Attribute_List">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Set_Attribute_List">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Get_Attribute_Single">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Set_Attribute_Single">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Forward_Close">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Unconnected_Send">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Forward_Open">
                        <xsd:complexType>
                            <xsd:attribute ref="SupportedService" fixed="true"/>
```

```
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="Get_Connection_Data">
                    <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="Search_Connection_Data">
                    <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="Ex_Forward_Open">
                    <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="Get_Object_Owner">
                    <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                    </xsd:complexType>
                </xsd:element>
                <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element ref="ConnectionDescriptions" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="ConnectionDescriptions">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="Connection" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:complexContent>
      <xsd:extension base="et_ConnectionType">
       <xsd:attribute name="id" use="required">
        <xsd:simpleType>
         <xsd:restriction base="xsd:ID">
          <xsd:pattern value="Connection[1-9][0-9]{0,4}"/>
         </xsd:restriction>
        </xsd:simpleType>
       </xsd:attribute>
      </xsd:extension>
     </xsd:complexContent>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="ProxyConnect" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:complexContent>
      <xsd:extension base="et_ConnectionType">
       <xsd:attribute name="id" use="required">
        <xsd:simpleType>
         <xsd:restriction base="xsd:ID">
          <xsd:pattern value="ProxyConnect[1-9][0-9]{0,4}"/>
         </xsd:restriction>
        </xsd:simpleType>
       </xsd:attribute>
      </xsd:extension>
     </xsd:complexContent>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="ProxiedConnect" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:complexContent>
      <xsd:extension base="et_ConnectionType">
       <xsd:attribute name="id" use="required">
        <xsd:simpleType>
         <xsd:restriction base="xsd:ID">
          <xsd:pattern value="ProxiedConnect[1-9][0-9]{0,4}"/>
         </xsd:restriction>
        </xsd:simpleType>
       </xsd:attribute>
      </xsd:extension>
     </xsd:complexContent>
```

```
            </xsd:complexType>
         </xsd:element>
         <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:key name="ConnectionName">
      <xsd:selector xpath="Connection|ProxyConnect|ProxiedConnect"/>
      <xsd:field xpath="Name_String"/>
   </xsd:key>
</xsd:element>
<xsd:element name="MessageRouter">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="MessageRouter_InstanceAttributes" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="Object_List">
                     <xsd:complexType>
                        <xsd:simpleContent>
                           <xsd:extension base="xsd:string">
                              <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                           </xsd:extension>
                        </xsd:simpleContent>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="MaximumConnectionSupported">
                     <xsd:complexType>
                        <xsd:simpleContent>
                           <xsd:extension base="dt_UINT">
                              <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                           </xsd:extension>
                        </xsd:simpleContent>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="MessageRouter_InstanceOperations" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="Get_Attribute_All">
                     <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="Get_Attribute_List">
                     <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="Get_Attribute_Single">
                     <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="CNPhysicalLayer">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="Connectors" minOccurs="0">
            <xsd:complexType>
               <xsd:attribute name="Media">
                  <xsd:simpleType>
                     <xsd:restriction base="xsd:NMTOKEN">
                        <xsd:enumeration value="Redundant"/>
                        <xsd:enumeration value="NonRedundant"/>
                     </xsd:restriction>
                  </xsd:simpleType>
               </xsd:attribute>
```

```
                <xsd:attribute name="NetworkAccessPort">
                   <xsd:simpleType>
                      <xsd:restriction base="xsd:NMTOKEN">
                         <xsd:enumeration value="Present"/>
                         <xsd:enumeration value="Absent"/>
                      </xsd:restriction>
                   </xsd:simpleType>
                </xsd:attribute>
             </xsd:complexType>
          </xsd:element>
          <xsd:element name="Delay1" minOccurs="0">
             <xsd:complexType>
                <xsd:sequence>
                   <xsd:element name="Units">
                      <xsd:simpleType>
                         <xsd:restriction base="xsd:string">
                            <xsd:pattern value="Param[1-9][0-9]*"/>
                         </xsd:restriction>
                      </xsd:simpleType>
                   </xsd:element>
                   <xsd:element name="MinDelayPerUnit" type="dt_UDINT"/>
                   <xsd:element name="MaxDelayPerUnit" type="dt_UDINT"/>
                </xsd:sequence>
             </xsd:complexType>
          </xsd:element>
          <xsd:element name="Other" type="xsd:string" minOccurs="0"/>
          <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
       </xsd:sequence>
    </xsd:complexType>
 </xsd:element>
 <xsd:element name="CNLinkLayer">
    <xsd:complexType>
       <xsd:sequence>
          <xsd:element name="MAC-IDSetting" minOccurs="0">
             <xsd:complexType>
                <xsd:sequence>
                   <xsd:element name="MinimumMAC-ID" minOccurs="0">
                      <xsd:simpleType>
                         <xsd:restriction base="et_MAC-IDRange">
                            <xsd:maxInclusive value="98"/>
                         </xsd:restriction>
                      </xsd:simpleType>
                   </xsd:element>
                   <xsd:element name="MaximumMAC-ID" minOccurs="0">
                      <xsd:simpleType>
                         <xsd:restriction base="et_MAC-IDRange">
                            <xsd:minExclusive value="0"/>
                         </xsd:restriction>
                      </xsd:simpleType>
                   </xsd:element>
                   <xsd:element name="DefaultMAC-ID" type="et_MAC-IDRange" minOccurs="0"/>
                </xsd:sequence>
                <xsd:attribute name="SwitchType" use="required">
                   <xsd:simpleType>
                      <xsd:restriction base="xsd:NMTOKEN">
                         <xsd:enumeration value="DipSwitch"/>
                         <xsd:enumeration value="RotarySwitch"/>
                         <xsd:enumeration value="SoftwareOnly"/>
                      </xsd:restriction>
                   </xsd:simpleType>
                </xsd:attribute>
                <xsd:attribute name="SoftwareSettable" use="required">
                   <xsd:simpleType>
                      <xsd:restriction base="xsd:boolean">
                         <xsd:pattern value="true"/>
                         <xsd:pattern value="false"/>
                      </xsd:restriction>
                   </xsd:simpleType>
                </xsd:attribute>
             </xsd:complexType>
          </xsd:element>
          <xsd:element name="Capacity" minOccurs="0">
             <xsd:complexType>
                <xsd:sequence>
                   <xsd:element name="MaxReceiveLPacketsPerNUT" type="et_MaxPacketsPerNUT"/>
                   <xsd:element name="MaxTransmitLPacketsPerNUT" type="et_MaxPacketsPerNUT"/>
```

```
                </xsd:sequence>
             </xsd:complexType>
          </xsd:element>
          <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
       </xsd:sequence>
    </xsd:complexType>
 </xsd:element>
 <xsd:element name="ControlNetObject">
    <xsd:complexType>
       <xsd:sequence>
          <xsd:element name="ControlNet_InstanceAttributes" minOccurs="0">
             <xsd:complexType>
                <xsd:sequence>
                   <xsd:element name="Pending_Link_Config">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Current_Link_Config">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Diagnostic_Counters">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Station_Status">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="MAC_ID">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Scheduled_Max_Frame">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Error_Log">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Ext_Diagnostic_Counters">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="Active_node_table">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:element name="New_node_table">
                      <xsd:complexType>
                         <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"
use="required"/>
                      </xsd:complexType>
                   </xsd:element>
                   <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                </xsd:sequence>
             </xsd:complexType>
```

```
            </xsd:element>
            <xsd:element name="ControlNet_InstanceOperations" minOccurs="0">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Get_Attribute_All">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Set_Attribute_All">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Get_Attribute_List">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Set_Attribute_List">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Reset">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Get_Attribute_Single">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Set_Attribute_Single">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Get_and_Clear">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Enter_Listen_Only">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Where_am_I">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService" fixed="true"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Auto_Address">
                        <xsd:complexType>
                           <xsd:attribute ref="SupportedService"/>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Ports">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Port" minOccurs="0" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="PortTypeName">
                        <xsd:simpleType>
                           <xsd:union>
                              <xsd:simpleType>
```

```
                                <xsd:restriction base="xsd:NMTOKEN">
                                    <xsd:enumeration value="ControlNet"/>
                                    <xsd:enumeration value="ControlNet_Redundant"/>
                                    <xsd:enumeration value="TCP"/>
                                    <xsd:enumeration value="DeviceNet"/>
                                </xsd:restriction>
                            </xsd:simpleType>
                            <xsd:simpleType>
                                <xsd:restriction base="et_VendorSpecificKeyword"/>
                            </xsd:simpleType>
                        </xsd:union>
                    </xsd:simpleType>
                </xsd:element>
                <xsd:element name="PortName" type="dt_EDS_Char_Array" minOccurs="0"/>
                <xsd:element name="PortObject" type="dt_EPATH" minOccurs="0"/>
                <xsd:element name="PortNumber" type="dt_UINT"/>
                <xsd:element name="PortSpecific">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                        </xsd:sequence>
                    </xsd:complexType>
                </xsd:element>
                <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
            <xsd:attribute name="id" use="required">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:ID">
                        <xsd:pattern value="Port[1-9][0-9]{0,4}"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:attribute>
        </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="NM-ControlNetObject">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="ControlNetObject_ClassAttributes" minOccurs="0">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="ObjectRevision">
       <xsd:complexType>
        <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory" fixed="Get"/>
       </xsd:complexType>
      </xsd:element>
      <xsd:element name="MaxInstance">
       <xsd:complexType>
        <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory" fixed="Get"/>
       </xsd:complexType>
      </xsd:element>
      <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="ControlNetObject_ClassOperations" minOccurs="0">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="Get_Attribute_Single">
       <xsd:complexType>
        <xsd:attribute ref="SupportedService" fixed="true"/>
       </xsd:complexType>
      </xsd:element>
      <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
<xsd:element name="NM-ConnectionManager">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="ConnectionManager_ClassAttributes" minOccurs="0">
```

```
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="ObjectRevision">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="MaxInstance">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="OptionalAttributeList">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="ConnectionManager_ClassOperations" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="Get_Attribute_All">
                     <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="Get_Attribute_List">
                     <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="Get_Attribute_Single">
                     <xsd:complexType>
                        <xsd:attribute ref="SupportedService"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="NM-MessageRouter">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="MessageRouter_ClassAttributes" minOccurs="0">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="ObjectRevision">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="OptionalAttributeList">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="OptionalServiceList">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="MaxIDClassAttributes">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="MaxIDInstanceAttributes">
                     <xsd:complexType>
                        <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalGet"/>
                     </xsd:complexType>
```

```
                    </xsd:element>
                    <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                 </xsd:sequence>
              </xsd:complexType>
           </xsd:element>
           <xsd:element name="MessageRouter_ClassOperations" minOccurs="0">
              <xsd:complexType>
                 <xsd:sequence>
                    <xsd:element name="Get_Attribute_All">
                       <xsd:complexType>
                          <xsd:attribute ref="SupportedService"/>
                       </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Get_Attribute_List">
                       <xsd:complexType>
                          <xsd:attribute ref="SupportedService"/>
                       </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Get_Attribute_Single">
                       <xsd:complexType>
                          <xsd:attribute ref="SupportedService"/>
                       </xsd:complexType>
                    </xsd:element>
                    <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                 </xsd:sequence>
              </xsd:complexType>
           </xsd:element>
        </xsd:sequence>
     </xsd:complexType>
  </xsd:element>
  <xsd:element name="Keeper">
     <xsd:complexType>
        <xsd:sequence>
           <xsd:element name="Keeper_Class">
              <xsd:complexType>
                 <xsd:sequence>
                    <xsd:element name="Keeper_ClassAttributes" minOccurs="0">
                       <xsd:complexType>
                          <xsd:sequence>
                             <xsd:element name="ObjectRevision">
                                <xsd:complexType>
                                   <xsd:simpleContent>
                                      <xsd:extension base="dt_UINT">
                                         <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                      </xsd:extension>
                                   </xsd:simpleContent>
                                </xsd:complexType>
                             </xsd:element>
                             <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                          </xsd:sequence>
                       </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="Keeper_ClassOperations" minOccurs="0">
                       <xsd:complexType>
                          <xsd:sequence>
                             <xsd:element name="Get_Attribute_Single">
                                <xsd:complexType>
                                   <xsd:attribute ref="SupportedService" fixed="true"/>
                                </xsd:complexType>
                             </xsd:element>
                             <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                          </xsd:sequence>
                       </xsd:complexType>
                    </xsd:element>
                 </xsd:sequence>
              </xsd:complexType>
           </xsd:element>
           <xsd:element name="Keeper_Instance">
              <xsd:complexType>
                 <xsd:sequence>
                    <xsd:element name="Keeper_InstanceAttributes" minOccurs="0">
                       <xsd:complexType>
                          <xsd:sequence>
                             <xsd:element name="KeeperStatus">
                                <xsd:complexType>
```

```
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="PortStatusNodes1-99">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="CurrentNetworkParameters">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="CurrentLinkName">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="TableUniqueIdentifier">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="LinkTableUniqueIdentifier">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Get"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="CurrentCableConfiguration">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="CoSummary">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="ConnectionOriginatorInformation">
                                 <xsd:complexType>
                                    <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                           </xsd:sequence>
                        </xsd:complexType>
                     </xsd:element>
                     <xsd:element name="Keeper_InstanceOperations" minOccurs="0">
                        <xsd:complexType>
                           <xsd:sequence>
                              <xsd:element name="Get_Attribute_List">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Set_Attribute_List">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Get_Attribute_Single">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Set_Attribute_Single">
                                 <xsd:complexType>
```

```
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Obtain_Network_Resource">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Hold_Network_Resource">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Release_Network_Resource">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Change_Start">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Change_Complete">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Change_Abort">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Get_Signature">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Get_Attribute_Fragment">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:element name="Set_Attribute_Fragment">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                           </xsd:sequence>
                        </xsd:complexType>
                     </xsd:element>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ControlNetScheduling">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ControlNetScheduling_Class">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="ControlNetScheduling_ClassAttributes" minOccurs="0">
                        <xsd:complexType>
                           <xsd:sequence>
                              <xsd:element name="ObjectRevision">
                                 <xsd:complexType>
                                    <xsd:simpleContent>
                                       <xsd:extension base="dt_UINT">
                                          <xsd:attribute name="Access_Rule"
type="at_AccessType_Mandatory" use="required" fixed="Get"/>
                                       </xsd:extension>
                                    </xsd:simpleContent>
                                 </xsd:complexType>
```

```
                                </xsd:element>
                                <xsd:element name="MaxInstance">
                                   <xsd:complexType>
                                      <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
fixed="Get"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="NumberOfInstances">
                                   <xsd:complexType>
                                      <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
fixed="Get"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                             </xsd:sequence>
                          </xsd:complexType>
                       </xsd:element>
                       <xsd:element name="ControlNetScheduling_ClassOperations" minOccurs="0">
                          <xsd:complexType>
                             <xsd:sequence>
                                <xsd:element name="Get_Attribute_All">
                                   <xsd:complexType>
                                      <xsd:attribute ref="SupportedService" fixed="true"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Get_Attribute_List">
                                   <xsd:complexType>
                                      <xsd:attribute ref="SupportedService"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Create">
                                   <xsd:complexType>
                                      <xsd:attribute ref="SupportedService" fixed="true"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Get_Attribute_Single">
                                   <xsd:complexType>
                                      <xsd:attribute ref="SupportedService"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="Restart-Connections">
                                   <xsd:complexType>
                                      <xsd:attribute ref="SupportedService" fixed="true"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                             </xsd:sequence>
                          </xsd:complexType>
                       </xsd:element>
                    </xsd:sequence>
                 </xsd:complexType>
              </xsd:element>
              <xsd:element name="ControlNetScheduling_Instance">
                 <xsd:complexType>
                    <xsd:sequence>
                       <xsd:element name="ControlNetScheduling_InstanceAttributes" minOccurs="0">
                          <xsd:complexType>
                             <xsd:sequence>
                                <xsd:element name="Route">
                                   <xsd:complexType>
                                      <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="TimeOut">
                                   <xsd:complexType>
                                      <xsd:attribute name="Access_Rule" type="at_AccessType_Mandatory"
use="required" fixed="Set"/>
                                   </xsd:complexType>
                                </xsd:element>
                                <xsd:element name="ControllerState">
                                   <xsd:complexType>
                                      <xsd:attribute name="Access_Rule" type="at_AccessType_OptionalSet"
use="required"/>
                                   </xsd:complexType>
                                </xsd:element>
```

```
            <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ControlNetScheduling_InstanceOperations" minOccurs="0">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Get_Attribute_All">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Set_Attribute_All">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Get_Attribute_List">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Set_Attribute_List">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Create">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Delete">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService" fixed="true"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Get_Attribute_Single">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Set_Attribute_Single">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Kick_Timer">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService" fixed="true"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Read">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService" fixed="true"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Conditional_Write">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService" fixed="true"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Forced_Write">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Change_Start">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService" fixed="true"/>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Break_Connections">
               <xsd:complexType>
                  <xsd:attribute ref="SupportedService" fixed="true"/>
               </xsd:complexType>
```

```
                              </xsd:element>
                              <xsd:element name="Change_Complete">
                                 <xsd:complexType>
                                    <xsd:attribute ref="SupportedService" fixed="true"/>
                                 </xsd:complexType>
                              </xsd:element>
                              <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
                           </xsd:sequence>
                        </xsd:complexType>
                     </xsd:element>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:attribute name="SupportedService" use="required">
      <xsd:simpleType>
         <xsd:restriction base="xsd:boolean">
            <xsd:pattern value="true|false"/>
         </xsd:restriction>
      </xsd:simpleType>
   </xsd:attribute>
   <xsd:simpleType name="et_MAC-IDRange">
      <xsd:restriction base="xsd:nonNegativeInteger">
         <xsd:maxInclusive value="99"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="et_MaxPacketsPerNUT">
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">
         <xsd:element name="NUT">
            <xsd:simpleType>
               <xsd:restriction base="xsd:decimal">
                  <xsd:minInclusive value="2"/>
                  <xsd:maxInclusive value="100"/>
                  <xsd:fractionDigits value="2"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="Lpackets" type="dt_UINT"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="et_ConnectionType">
      <xsd:sequence>
         <xsd:element name="Trigger_Transport" type="dt_DWORD"/>
         <xsd:element name="Connection_Parameters" type="dt_DWORD"/>
         <xsd:element name="O-T_RPI" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="dt_UDINT et_ParamReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="O-T_Size" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="dt_UINT et_ParamReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="O-T_Format" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="et_ParamReference et_AssemReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="T-O_RPI" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="dt_UDINT et_ParamReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="T-O_Size" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="dt_UINT et_ParamReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="T-O_Format" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="et_ParamReference et_AssemReference"/>
            </xsd:simpleType>
         </xsd:element>
```

```
         <xsd:element name="Config1_Size" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="dt_UINT et_ParamReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="Config1_Format" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="et_ParamReference et_AssemReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="Config2_Size" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="dt_UINT et_ParamReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="Config2_Format" minOccurs="0">
            <xsd:simpleType>
               <xsd:union memberTypes="et_ParamReference et_AssemReference"/>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="Name_String" type="dt_EDS_Char_Array"/>
         <xsd:element name="Help_String" type="dt_EDS_Char_Array"/>
         <xsd:element name="Path"/>
         <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

A.3.2 Communication network profile template description – XML encapsulation of EDS files

A.3.2.1 General

The communication network profile XML files used to encapsulate EDS files shall comply with the communication network profile XML schema as specified in A.3.2.2.

The semantics of the sub-elements of the ExternalProfileHandle element, used to reference an existing EDS file, are specified in Table A.2. Depending on the value of the attribute WrapperReference, the EDS file will be referenced using either identification elements from the EDS file itself, or from the product described by this EDS.

NOTE Choice of relevant identification elements will depend upon the expected usage of the wrapper file.

A.3.2.2 XML schema : EDS_CommNet_Profile_wrapper.xsd

NOTE This XML schema includes the file “MasterTemplateTypes.xsd” (see A.2.1.3.1).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <!-- Target namespaces are not specified in this master template -->
   <xsd:redefine schemaLocation="MasterTemplateTypes.xsd">
      <xsd:complexType name="ISO15745Reference_DataType">
         <xsd:complexContent>
            <xsd:restriction base="ISO15745Reference_DataType">
               <xsd:sequence>
                  <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
                  <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
                  <xsd:element name="ProfileTechnology" type="xsd:string" fixed="EDS"/>
               </xsd:sequence>
            </xsd:restriction>
         </xsd:complexContent>
      </xsd:complexType>
   </xsd:redefine>
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
```

```
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"
fixed="CommunicationNetwork"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" fixed="CSI"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ExternalProfileHandle">
               <xsd:complexType>
                  <xsd:complexContent>
                     <xsd:extension base="ProfileHandle_DataType">
                        <xsd:attribute name="WrapperReference" use="optional" default="FILEINFO">
                           <xsd:simpleType>
                              <xsd:restriction base="xsd:NMTOKEN">
                                 <xsd:enumeration value="FILEINFO"/>
                                 <xsd:enumeration value="DEVICEINFO"/>
                              </xsd:restriction>
                           </xsd:simpleType>
                        </xsd:attribute>
                     </xsd:extension>
                  </xsd:complexContent>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

A.4 Electronic Data Sheet (EDS)

A.4.1 Common CIP EDS requirements

A.4.1.1 General

This subclause specifies the file encoding requirements of the Electronic Data Sheet (EDS) which are common to all CIP-based networks. The EDS encoding requirements define the standard file encoding format to use for CIP products without regard to the configuration tool host platform or file system.

The term “file” as used in this chapter refers to any recognized file format associated with a configuration tool’s file system without regard to the file storage media.

An EDS file is defined as an ASCII file, which includes an ASCII representation of objects in the device that can be accessed from the network (e.g. Parameter and Assembly), and some additional information required to support object addressing.

A.4.1.2 EDS content

A.4.1.2.1 EDS structure

A single file shall contain the entire EDS. An EDS shall consist of sections. Table A.5 summarizes the structure of the sections which are common to several CIP-based networks, the corresponding legal section delimiters, and the order of these sections in an EDS.

Table A.5 — CIP EDS file structure

Section Name	Legal Delimiter	Placement	Required/Optional
File Description	[File]	1	Required
Device Description	[Device]	2	Required
Device Classification	[Device Classification]	[a]	Optional
Parameter Class	[ParamClass]	[a]	Optional
Parameters	[Params]	[a]	Optional
Parameter Groups	[Groups]	[a]	Optional
Assembly	[Assembly]	[a]	Optional
Connection Characteristics	[Connection Manager]	[a]	Optional
Port	[Port]	[a]	Optional
Modular	[Modular]	[a]	Optional
Vendor Specific	[VendorID_vendorspecifickeyword]	Last	Optional
[a] Placement of these optional groups only needs to follow the required groups			

The Electronic Data Sheet (EDS) contents shall be organised as follows:

— all EDS files shall contain the File Description section, which shall be the first section in the EDS file and shall use the legal delimiter [File];

— all EDS files shall contain the Device Description section, which shall immediately follow the File Description section and shall use the legal delimiter [Device];

— the optional sections described in this specification may be present in any order provided that no forward references exist within the EDS file;

— the optional Vendor Specific section(s) shall uses the legal delimiter(s) [VendorID_vendorspecifickeyword] as specified in A.4.1.2.2.11 and shall be placed after all the sections defined in this specification.

A.4.1.2.2 EDS formatting rules

A.4.1.2.2.1 General

An EDS file shall consist of sections, entries, fields, comments and white space. This subclause defines the rules that shall be observed when defining an EDS.

A.4.1.2.2.2 EDS White Space

White space may be used in the EDS file, but shall be ignored by all EDS interpreters when it appears outside of fields and double quoted character arrays.

The EDS interpreter shall treat the following characters as white space characters. These characters, read by the interpreter but not encoded as human-readable characters, designate the presence of blank space in a file.

— Space character

— New line

— Carriage Return

— Linefeed

— Tabs, vertical and horizontal

— Form Feed

— End of File marker

— Comments

A.4.1.2.2.3 Keyword characters

All keywords within an EDS file shall be composed of ASCII characters from the following list:

— upper case letters A through Z;

— lower case letters a through z;

— numerals 0 through 9;

— the special character underscore "_";

— the space character.

The space shall only be used in a section keyword. The space shall only appear internal to a section name, and multiple sequential spaces are invalid.

A.4.1.2.2.4 Sections

The EDS file shall be partitioned into required and optional sections.

A.4.1.2.2.5 Section delimiters

Each section in the EDS shall be properly delimited by a section keyword in square brackets (the Legal Delimiter). The valid section legal delimiters shall be those specified in Table A.5.

A.4.1.2.2.6 Section keywords

Each section keyword is defined to be the text between the beginning of section keyword delimiter "[" and the terminating delimiter "]". The characters valid for use in section keywords are defined in A.4.1.2.2.3. There are two types of section keywords, public and vendor specific.

A.4.1.2.2.7 Section order

Each required section shall be placed in the required order as specified in A.4.1.2. Optional sections may be omitted entirely or included with empty data place holders. Except for the Vendor Specific section(s), optional sections may be placed in any order. The Vendor Specific section(s) shall be placed last in the EDS file.

A.4.1.2.2.8 Entry

Each section in the EDS shall contain one or more entries beginning with an entry keyword followed by an equal sign. The entry keyword meaning shall be global in scope, allowing keywords defined in one section to be used in another. Each entry shall terminate with a semicolon. An entry may extend over multiple lines, as long as commas properly delimit the fields.

A.4.1.2.2.9 Entry keywords

An entry keyword shall consist of a unique sequence of keyword characters, as defined in A.4.1.2.2.3. There are two types of entry keywords, public and vendor specific.

A.4.1.2.2.10 Public keyword

A Public Keyword shall always be defined within the CIP specification by the responsible vendors associations. A public keyword shall never begin with any numeric digit.

A.4.1.2.2.11 Vendor-specific keywords

Keywords may be vendor-specific. These keywords shall begin with the Vendor ID of the company making the addition followed by an underscore (VendorID_VendorSpecificKeyword). The VendorID shall be displayed in decimal and shall not contain leading zeroes. Each vendor is responsible for maintaining and documenting their vendor-specific keywords.

A.4.1.2.2.12 Entry fields

Each entry shall contain one or more fields. Comma delimiters shall separate all fields. The meaning of the field(s) shall depend on the context of the section. Entry fields are either required or optional, as defined by this specification. A white space or nothing between commas shall be used for optional fields not provided. A semicolon may be used to designate the absence of trailing optional fields. The term “Field Number” shall indicate field position within the entry. Fields shall be numbered from left-to-right (or up-to-down) starting with the number 1.

A.4.1.2.2.13 Field keywords

A field keyword shall consist of a unique sequence of keyword characters, as defined in A.4.1.2.2.3. There are two types of field keywords, public and vendor specific.

A.4.1.2.2.14 Complex data fields

Certain entry fields shall be specified with data that cannot be specified by a single value between the comma delimiters. The ability to further delimit an entry field is defined via the use of one or more sets of matching brace characters "{" and "}". The content between brace characters shall be considered a single item or entry. Content may be grouped in multiple braces.

A.4.1.2.2.15 Comments

Comments shall be delimited with the dollar sign character ($) and the new line character. The EDS interpreter shall treat all characters between the comment delimiters as white space. The $ comment delimiter appearing inside of a field or double quoted character array shall not be treated as a comment delimiter.

EXAMPLE

Some example comments are:

```
$ This is a valid comment line<NL>
1, 2, 3;                          $ This is a valid comment <NL>
$ Comments cannot span <NL>
more than one line <NL>           <= This is an error - no $
```

A.4.1.2.2.16 EDS formatting structure example

Figure A.1 examples highlight the structure of the Electronic Data Sheet.

```
[section name]
$ Comment - extends to end of line
Entry1=Field1, Field2, Field3;       $ Entire entry on one line
Entry2=Field1, Field2, Field3, Field4;  $ Entire entry on one line

Entry3=                              $ Multiple line entry
       Field1,                       $ Field1
       Field2,                       $ Field2
       Field3;                       $ Field3

Entry4=                              $ Combination
       Field1, Field2,               $ Fields 1 and 2 on one line
       Field3,                       $ Field3
       Field4;                       $ Field4

Entry5=   1,                         $ Field 1 specifies the value 1
       {1,2,3};                      $ Field 2 specifies an array or
                                     $ structure with three values

Entry6=   { 44, {22,33,11} };        $ Entry 6 specifies a single field.
                                     $ The field contains two sets of data.
                                     $ The first set is the single value 44
                                     $ The second set contains three values

65535_Entry=                         $ Vendor Specific entry for
          Field1, Field2;            $ Vendor_ID 65535 with two fields
```

Figure A.1 — EDS formatting structure example (informative)

A.4.1.2.3 File naming requirements

No file naming conventions are specified for disk-based EDS files, except for files in a DOS/Windows environment : these files shall have the suffix ".EDS" appended to the file name.

A.4.1.3 EDS data encoding requirements

A.4.1.3.1 General

This section specifies the data encoding requirements for the EDS file.

The information contained in the EDS file may represent attributes of object instances in the device to be configured. All data in the EDS file shall be ASCII text whereas the object class and instance attributes need not be ASCII (available data types are defined in the CIP specification). Therefore, translation between data contained in an EDS file and the object attributes may be needed : this translation is specified in the following subclauses.

The elementary data types specified in the CIP specification are also used for other elements of the EDS, however, the meaning is transformed as described in the following subclauses (see A.4.1.3.3 to A.4.1.3.10).

Some data types are used solely in EDS files (see A.4.1.3.11 to A.4.1.3.14).

A.4.1.3.2 ASCII character file convention

All data in the EDS shall be encoded using 8-bit ASCII characters, where all references to "ASCII characters" mean an 8-bit ASCII character format (as defined by Tables 1 and 2, Row 00 of ISO/IEC 10646-1:2000). Characters that

cannot be displayed on an ANSI terminal shall not be used in identifier names or in data representations. The valid ASCII character values shall include newline, tab, and those from 32 to 126 decimal.

A.4.1.3.3 Character string convention – EDS_Char_Array

A.4.1.3.3.1 General

All string data in the EDS file shall be character strings of fixed length, without null terminators, and shall be enclosed by double quotes (EDS_Char_Array data type).

There are two forms of string data conversions. Characters contained between double quotes shall be converted into 8-bit ASCII characters. Characters contained between double quotes that are preceded by a capital L shall be converted into UNICODE (16-bit) characters.

EXAMPLE 1 “This results in a string composed by 8-bit characters”

EXAMPLE 2 L“A string of UNICODE characters, including the Greek character Pi \u03C0”

NOTE The text \u03C0 specifies a single 16-bit character whose value is 03C0. In the UNICODE character set, this is Table 9, Row 3, Basic Greek – the character for lower case "Pi". Descriptions of character escape sequences is described in A.4.1.3.3.5.

A.4.1.3.3.2 Handling insufficient characters in a string field

An EDS interpreter shall use right-justification of characters in a field and fill any unspecified characters with leading blanks (ASCII 0x20) for the remaining length of the string.

EXAMPLE If a parameter has a maximum string length of 8 and receives the string ”123AB”, the string is interpreted as ”~~~123AB”, where the tilde characters (~) represent spaces.

A.4.1.3.3.3 Handling excess characters in a string field

If a given string field contains too many characters, the EDS interpreter shall truncate characters from left to right.

EXAMPLE If a parameter has a maximum string length of 8 and receives the string ”I23ABCDEFG”, the string is truncated and interpreted as ”I23ABCDE”.

A.4.1.3.3.4 String concatenation

Multiple strings with no intervening commas shall be concatenated.

EXAMPLE 1
The line : ”ABC” ”123” ”XYZ”
is interpreted as : ”ABC123XYZ”

The strings may also be on separate lines.

EXAMPLE 2
The following lines :
”ABC” $this is a comment
”123”
”XYZ”
are also interpreted as : ”ABC123XYZ”

For a UNICODE string (long string), only the first double quotes mark shall be preceded by a capital L.

EXAMPLE 3 L”ABC” ”123” ”XYZ” is the same as L”ABC123XYZ”.

A.4.1.3.3.5 String escape sequences

The EDS interpreter shall recognize all escape sequences listed in Table A.6. Interpretation is application specific.

Table A.6 — String escape sequences

Escape sequence	Translation
\\	\
\n	newline
\t	tab
\v	vertical tab
\b	backspace
\r	carriage return
\f	form feed
\a	the BELL character (0x07)
\”	”
\’	’
\xnn	single byte containing the value of “nn” as expressed in hexadecimal
\unnnn	pair of bytes containing the value of “nnnn” as expressed in hexadecimal. This form of string escape is only valid where the resultant string data is 16-bits in length, e.g. the L" form of string specification.

If a sequence not listed above is encountered, the interpreting device shall reject the entire string and indicate an error. EDS files shall only contain escape sequences defined in Table A.6.

A.4.1.3.4 ASCII string convention (STRING, SHORT_STRING, STRING2)

All string data types (STRING, SHORT_STRING, STRING2) used in the object attributes shall be converted into EDS_Char_Array in the EDS file.

A.4.1.3.5 STRINGI

The CIP International String (STRINGI) data type is encoded in an EDS file as a complex data representation. The entire content of a STRINGI entry shall be enclosed in a pair of braces. The number of language members, specified as a USINT, shall be followed by the language members definitions, each being enclosed in a pair of braces, and separated by a comma. Each language member of a STRINGI entry shall be specified as four fields. The first field (the language selection) shall be expressed as exactly a three character fixed length string enclosed in double quotation marks – language code as defined in ISO 639-2/T. The string data type shall be expressed using the data type code as defined in the CIP specification for STRING, STRING2, STRINGN or SHORT_STRING. The character set selection shall be expressed as a UINT – as defined in IANA MIB Printer Codes (RFC 1759). The string content portion of the language member shall be expressed as a string or long string.

EXAMPLE

The following represents a STRINGI entry with three languages:

```
Field1 =        { 3,
                {"eng",0xD0,4,"This is an ASCII English language string"},
                {"spa",0xD5,1000,L"Españoles palabras"},        $ “Spanish words”
                                                               $ using UNICODE
                {"deu",0xD0,4,"Spanische Wörter auf Deutsch"}  $“Spanish words in German”
                };
```

A.4.1.3.6 CIP path (EPATH)

The CIP EPATH data type, used in particular to define CIP path strings, shall be encoded in EDS files using the base format defined in ISO 15745-3 for EDS_Char_Array. In addition, the string contents for a CIP path or other EPATH data shall consist of groups of two adjacent hexadecimal characters separated by spaces. Both upper and lower case may be used.

EXAMPLE 1 “20 04 24 01”

EXAMPLE 2 “20 05 24 02 30 04”

A.4.1.3.7 ASCII unsigned integer convention - (USINT, UINT, UDINT, ULINT)

The unsigned integer data types represent positive integer values. Unsigned integer data shall be entered either in decimal or in hexadecimal notation with no whitespace or commas between characters. If hexadecimal notation is used to represent the unsigned integer characters, the two character sequence 0x, with no white space, shall precede the unsigned integer characters.

The range of legal USINT data is:

Decimal notation:	0	to	255
Hexadecimal notation:	0x0	to	0xFF

The range of legal UINT data is:

Decimal notation:	0	to	65535
Hexadecimal notation:	0x0	to	0xFFFF

The range of legal UDINT data is:

Decimal notation:	0	to	4294967295
Hexadecimal notation:	0x0	to	0xFFFFFFFF

The range of legal ULINT data is:

Decimal notation:	0	to	18446744073709551615
Hexadecimal notation:	0x0	to	0xFFFFFFFFFFFFFFFF

Leading zeros shall not be used for the decimal notation, but may be used for the hexadecimal notation. For hexadecimal notation, both upper and lower case may be used, and the total number of characters shall be limited to 10 (0x plus 8 more), or 18 (0x plus 16 more) for the ULINT type.

EXAMPLE The decimal UINT value 254 may be represented as 254 (decimal), or as 0xFE (hexadecimal), or as 0x000000FE (hexadecimal) but 0254 (decimal) and 0x0000000FE (hexadecimal) are illegal.

A.4.1.3.8 ASCII signed integer convention - (SINT, INT, DINT, LINT)

The SINT, INT, DINT and LINT data types represent signed integer data values. Signed integer data shall be entered either in decimal or in hexadecimal notation with no whitespace or commas between characters. If hexadecimal notation is used to represent the signed integer characters, the two character sequence 0x, with no white space, shall precede the integer value characters.

The range of legal SINT data is:

Decimal notation:	-128	to	127
Hexadecimal notation:	0x80	to	0x7F

The range of legal INT data is:

Decimal notation:	-32768	to	32767
Hexadecimal notation:	0x8000	to	0x7FFF

The range of legal DINT data is:

Decimal notation:	-2147483648	to	2147483647
Hexadecimal notation:	0x80000000	to	0x7FFFFFFF

The range of legal LINT data is:

Decimal notation:	-9223372036854775808 to 9223372036854775807
Hexadecimal notation:	0x8000000000000000 to 0x7FFFFFFFFFFFFFFF

Leading zeros shall not be used for the decimal notation, but may be used for the hexadecimal notation. For hexadecimal notation, both upper and lower case may be used, and the total number of characters shall be limited to 10 (0x plus 8 more), or 18 (0x plus 16 more) for the LINT type.

EXAMPLE The decimal INT value 254 may be represented as 254 (decimal), or as 0xFE (hexadecimal), or as 0x000000FE (hexadecimal) but 0254 (decimal) and 0x0000000FE (hexadecimal) are illegal.

A.4.1.3.9 ASCII word convention - (BYTE, WORD, DWORD, LWORD)

The BYTE, WORD, DWORD and LWORD data types represent bit-addressable values. These values are considered discrete bit position values and are not intended to represent either signed or unsigned integer values. However, these values, for convenience, shall be entered either in decimal, hexadecimal or binary notation with no whitespace or commas between characters. If hexadecimal (respectively binary) notation is used to represent value characters, the two character sequence 0x (respectively 0b), with no white space, shall precede the value characters.

The range of legal BYTE data is:

Decimal notation:	0 to 255
Hexadecimal notation:	0x0 to 0xFF
Binary notation:	0b00000000 to 0b11111111

The range of legal WORD data is:

Decimal notation:	0 to 65535
Hexadecimal notation:	0x0 to 0xFFFF
Binary notation:	0b0000000000000000 to 0b1111111111111111

The range of legal DWORD data is:

Decimal notation:	0 to 4294967295
Hexadecimal notation:	0x0 to 0xFFFFFFFF
Binary notation:	0b00000000000000000000000000000000 to 0b11111111111111111111111111111111

The range of legal LWORD data is:

Decimal notation:	0 to 18446744073709551615
Hexadecimal notation:	0x0 to 0xFFFFFFFFFFFFFFFF
Binary notation:	0b0000000000000000000000000000000000000000000000000000000000000000 to 0b1111111111111111111111111111111111111111111111111111111111111111

Leading zeros shall not be used for the decimal notation, but may be used for the hexadecimal and binary notations. For hexadecimal notation, both upper and lower case may be used, and the total number of characters shall be limited to 10 (0x plus 8 more), or 18 (0x plus 16 more) for the LWORD type.

EXAMPLE The decimal WORD value 254 may be represented as 254 (decimal), or as 0xFE (hexadecimal), or as 0x000000FE (hexadecimal) but 0254 (decimal) and 0x0000000FE (hexadecimal) are illegal.

A.4.1.3.10 ASCII floating point convention (REAL, LREAL)

The REAL and LREAL data types represent binary floating point values. The internal representation of these data formats are described by the IEEE Standard 754. This standard describes both numeric values and bit sequences which are interpreted as “Not a Number” (NaN) symbolic values and positive and negative infinity. The floating point values may be entered as either integer values, values based upon decimal floating point representation, or values entered in “scientific” notation using a base value and an offset in exponential form. Integer values are the same as those shown for the INT, DINT or LINT data types. These values cannot be use to represent fractional values. Decimal floating point values are those which have both a whole integer and fractional component. The

whole value and fractional components are separated by a decimal point "." or period character. The exponential (scientific) notation form of the value is the same as the fractional value representation with the addition of an exponential component. This exponent is always a signed integer power to ten applied to the base value.

NOTE The maximum precision of a floating point value is determined by the capabilities of the internal binary format, i.e. the number of binary digits available to encode the mantissa. Therefore, using a large number of decimal digits within the decimal notation (or the mantissa part of the scientific notation) of a floating point value is more for presentation convenience than precision. EDS defines arbitrary limits for the number of decimal digits.

The range of legal REAL data (single IEEE, 32 bit format) is based upon the formula:

value = $(-1)^{s}$•$(2)^{e-127}$•(m)

Where:

— "s" is the value of the sign bit;

— "e" is the eight bit exponent. This exponent allows an exponent range between -126 and +127;

— "m" is the normalized 24 bit mantissa (23 internal to the storage plus one hidden bit). This allows a range of mantissa values to range between 0 and 16777215.

The combination of "e" and "m" allows an approximate absolute value range of 0 to 3,4028e^{38}.

EDS uses for REAL data the following floating point values notations:

Integer (Fixed) notation: -16777215 to 16777215

Decimal (Floating Point) notation: 0.0 to ±9999999999999999

Where the total number of digits shall not exceed 16, in addition to the decimal point and sign characters. Both the decimal point character and the sign character may be omitted (+ sign is implied if the sign character is omitted).

Scientific notation: 0.0 to ±nn.nnnnnnnnnE±xxxx

Where the total number of digits in the mantissa shall not exceed 11 (in addition to the decimal point character and sign character), and the number of digits in the exponent shall not exceed 4 (in addition to the "E" character and sign character). The decimal point may be placed anywhere in the mantissa. Both the decimal point character and the sign character may be omitted in the mantissa (+ sign is implied if the sign character is omitted).

The range of legal LREAL data (double IEEE, 64 bit format) is based upon the formula:

value = $(-1)^{s}$•$(2)^{e-1023}$•(m)

Where:

— "s" is the value of the sign bit;

— "e" is the eleven bit exponent. This exponent allows an exponent range between -1022 and +1023;

— "m" is the normalized 53 bit mantissa (52 internal to the storage plus one hidden bit). This allows a range of mantissa values to range between 0 and 9007199254740991.

The combination of "e" and "m" allows an approximate absolute value range of 0 to 1,7976e^{308}.

EDS uses for LREAL data the following floating point values notations:

Integer (Fixed) notation: -9007199254740991 to 9007199254740991

Decimal (Floating Point) notation: 0.0 to ±9999999999999999

Where the total number of digits shall not exceed 16, in addition to the decimal point and sign characters. Both the decimal point character and the sign character may be omitted (+ sign is implied if the sign character is omitted).

Scientific notation: 0.0 to ±nnnn.nnnnnnnnnnnnE±xxxx

Where the total number of digits in the mantissa shall not exceed 16 (in addition to the decimal point character and sign character), and the number of digits in the exponent shall not exceed 4 (in addition to the “E” character and sign character). The decimal point may be placed anywhere in the mantissa. Both the decimal point character and the sign character may be omitted in the mantissa (+ sign is implied if the sign character is omitted).

In addition to the above value entries, the floating point representation allows for two styles of “Not a Number” or NaN symbolic entries, and two forms of infinity. There are two types of NaN; a Signaling NaN and a Quiet NaN. Also, the format allows for the representation of the values positive and negative infinity. For these cases, the following special words are reserved and shall be used to represent the entry of the associated floating point symbol:

— Quiet Not a Number: QUIET-NAN

— Signaling Not a Number: SIGNAL-NAN

— Positive Infinity: INFINITY (or +INFINITY)

— Negative Infinity: -INFINITY

A.4.1.3.11 EDS_Date

The EDS_Date data type shall be of the format mm-dd-yyyy, where mm is the month, dd is the day of the month and yyyy is the year. Valid values for the month, day and year portions of the mm-dd-yyyy shall be:

— mm 01 through 12;

— dd 01 through 31 (depending upon the month and year);

— yyyy1996 through 9999.

Two character years representations may be used in which case the EDS_Date data type shall be of the format; mm-dd-yy, where mm is the month, dd is the day of the month and yy is the year. In this case, the two digits for the year have an implied leading 19, such that yy=96 shall represent the year 1996. Valid values for the month, day and year portions of the mm-dd-yy parameters shall be:

— mm 01 through 12;

— dd 01 through 31 (depending upon the month and year);

— yy 96 through 99 (a leading 19 is implied).

NOTE Two character year representations are not recommended.

A.4.1.3.12 EDS_Time_Of_Day

The EDS_Time_Of_Day data type shall be of the format hh:mm:ss, where hh is hours, mm is minutes and ss is seconds. Valid values for the hours, minutes and seconds shall be:

— hh 00 through 23;

— mm 00 through 59;

— ss 00 through 59.

A.4.1.3.13 EDS_Revision

The EDS_Revision data type shall be of format Major_Revision.Minor_Revision with valid values of:

— Major_Revision 0 to 9;

— Minor_Revision 0 to 9.

An EDS_Revision of 0.0 shall be invalid.

EXAMPLE An EDS_Revision of 1.4 corresponds to a major revision of 1 and a minor revision of 4.

A.4.1.3.14 EDS_URL Uniform Resource Locator

All references to EDS_URL within the EDS requirements are for the formalized information necessary to locate and access resources via an Internet capable mechanism. An EDS_URL shall be encoded in EDS files using the base format defined in ISO 15745-3 for EDS_Char_Array. In addition, the string contents for an EDS_URL shall follow the format defined by the Internet's Network Working Group RFC 1738 "Uniform Resource Locator (URL)". For specifications made within an EDS file, the EDS_URL shall be limited to any of the forms:

— http;

— ftp;

— file.

A.4.1.4 Basic EDS file requirements

A.4.1.4.1 Overview

This subclause describes the basic sections of an EDS which are common to several CIP-based networks, and specifies the corresponding usage requirements. Table A.7 gives the subclause location of these section definitions.

Table A.7 — Basic sections definition

EDS sections	Defined in
File description section	A.4.1.4.2
Device description section	A.4.1.4.3
Device classification section	A.4.1.4.4
Parameter class section	A.4.1.4.5
Parameters section	A.4.1.4.6
Parameter groups section	A.4.1.4.7
Assembly section	A.4.1.4.8
Connection manager section	A.4.1.4.9
Port section	A.4.1.4.10
Modular section	A.4.1.5.2

A.4.1.4.2 File Description section

The file description section shall contain administrative information about the EDS file. A configuration tool shall read this information, format it, and display it to the user. The user can also access this section with a text file viewer and display the unformatted information. This section shall not require modification unless the user manually modifies the file. The file description section shall contain the entries shown in Table A.8.

Table A.8 — File description format

Entry Name	Entry Keyword	Field Number	Data Type	Required/Optional
File Description Text	DescText	1	EDS_Char_Array	Required
File Creation Date	CreateDate	1	EDS_Date	Required
File Creation Time	CreateTime	1	EDS_Time_Of_Day	Required
Last Modification Date	ModDate	1	EDS_Date	Conditional
Last Modification Time	ModTime	1	EDS_Time_Of_Day	Conditional
EDS Revision	Revision	1	EDS_Revision	Required
Home URL	HomeURL	1	EDS_URL	Optional

The entries in the file description section shall provide the information as shown in Table A.9.

Table A.9 — File description entries

Entries	Description
File Description Text	A single line of text displayed by the configuration tool. The EDS developer shall assign a meaningful line of text for this entry. Double quotes shall enclose all character arrays.
File Creation Date	The creation date of the EDS, assigned by the EDS developer. Provided only for convenience, this date can be used to get version information about the file. A configuration tool shall not use this information to perform any type of version control, but it may display the contents.
File Creation Time	The creation time of the EDS, assigned by the EDS developer. Provided only for convenience, this time can be used to get version information about the file. A configuration tool shall not use this information to perform any type of version control, but it may display the contents.
Last Modification Date	The date of the last modification to the EDS. A configuration tool that allows modification of the EDS file shall update this field as needed. Provided only for convenience, the configuration tool shall display the contents of this entry if it exists. If a configuration tool changes the EDS, the configuration tool shall update this field. However, if the EDS is modified manually or with a text editor, this field shall also be updated. This entry is required if either: – the EDS file is modified by a software tool; – the Last Modification Time entry is present.
Last Modification Time	The time of the last modification to the EDS. A configuration tool that allows modification of the EDS file shall update this entry as needed. Provided for convenience, the configuration tool shall display the contents of this entry if it exists. If a configuration tool changes the EDS, the configuration tool shall update this field. However, if the EDS is modified manually or with a text editor, this field shall also be updated.
EDS Revision	The revision of the EDS. The EDS revision need not have any relationship to the product's revision, it is simply the revision of the EDS file itself.
Home URL	Uniform Resource Locator of the master EDS file, the Icon file and other files related to this EDS. The HomeURL shall specify a complete qualified URL for referencing a master version of the EDS file. In addition, the referenced area (without the file name specification) is used to specify an area where other related file(s) relating to the device described by this EDS are contained.

Figure A.2 is an example that shows a typical [File] section.

```
   [File]
      DescText = "Smart Widget EDS File";
      CreateDate = 04-03-94;                $ created
      CreateTime = 17:51:44;
      ModDate = 04-06-94;                   $ last changed
      ModTime = 22:07:30;
      Revision = 2.1;                       $ Revision of EDS
      HomeURL = “http://www.odva.org/EDS/example.eds”;
```

Figure A.2 — [File] section example (informative)

A.4.1.4.3 Device Description section

The Device Description section shall contain manufacturer's information about the device, including some of the same values as in a device Identity Object. The device description section shall contain the entries specified in Table A.10

Table A.10 — Device description format

Entry Name	Entry Keyword	Field Number	Data Type	Required/Optional
Vendor Id[a,b]	VendCode	1	UINT	Required
Vendor Name	VendName	1	EDS_Char_Array	Required
Device Type[a,b]	ProdType	1	UINT	Required
Device Type String	ProdTypeStr	1	EDS_Char_Array	Required
Product Code[a,b]	ProdCode	1	UINT	Required
Major Revision[a,b]	MajRev	1	USINT	Required
Minor Revision[a]	MinRev	1	USINT	Required
Product Name[c]	ProdName	1	EDS_Char_Array	Required
Catalog Number	Catalog	1	EDS_Char_Array	Optional
Exclude from Adapter Rack Connection	ExcludeFromAdapterRackConnection	1	EDS_Char_Array	Optional
Icon File Name	Icon	1	EDS_Char_Array	Optional

[a] This entry represents an attribute of the Identity Object

[b] This entry is used to match an EDS with a specific product/revision

[c] This entry represents an attribute of the Identity Object, although the data type may be slightly different.

The entry name for the device description field describes the unique data entry line number.

A configuration tool shall use the required entries in the device description section to match the EDS to the device being configured. The entries in the device description section shall provide the information as shown in Table A.11.

Table A.11 — Device description entries

Entries	Description
Vendor ID	Numeric vendor identifier as defined by the Identity Object, Attribute 1.
Vendor Name	Textual vendor name. When displayed, truncation may occur to meet the display capabilities.
Device Type	Numeric device identifier as defined by the Identity Object, Attribute 2.
DeviceType String	Textual description of device type exactly as defined in the corresponding CIP device profile. The individual vendors may choose the strings for vendor specific device types.
Product Code	Vendor assigned numeric product code identifier as defined by the Identity Object, Attribute 3. Each product code shall have its own EDS.
Major Revision	Vendor-assigned major revision number as defined by the Identity Object, Attribute 4. The major revision of a product may be typically incremented when there is a change to the form, fit, or function of the device. Changes to major revisions shall be used by a configuration tool to match a device to an EDS.
Minor Revision	Vendor-assigned minor revision number as defined by the Identity Object, Attribute 4. The minor revision number shall be used to identify changes in a product that do not effect user configuration choices (e.g. firmware bug fixes, an additional LED, internal hardware changes). Changes in minor revisions shall not be used by a configuration tool to match a device with an EDS.
Product Name	Textual product name as defined by the Identity Object, Attribute 7. When displayed, truncation may occur to meet the display capabilities.
Catalog Number	Textual catalog or model number. One or more catalog numbers may be associated with a particular product code. NOTE In the case of multiple catalog numbers, it is still useful to provide as much of the catalog number as is practical. For example, 1438-BAC7xx where 'xx' represents variants in the catalog number supported by this product code/EDS.
ExcludeFromAdapterRackConnection	This field is used to describe if a rack based device is required to be excluded from an adapter rack connection. If the field value is the string "Yes" this module shall be excluded from adapter rack connections by resetting the associated slot mask bits (input, output and configuration). If the field value is the string "No" or this optional field is omitted the associated slot mask bits may be set.
Icon File Name	File name of an icon file. Identifies a file that contain a graphical representation of the device. The file shall have the *.ICO MSWindows format, and shall minimally contain a 16x16 icon. The file may also contain 32x32, 48x48, and 64x64 icons. The location of the icon file is the combination of the location specified by the HomeURL keyword (without the HomeURL file name component) and the file name specified by this keyword. This keyword shall only be present when a HomeURL keyword exists.

Figure A.3 is an example that shows a typical Device Section.

```
[Device]
   VendCode = 65535;
   VendName = "Widget-Works, Inc.";
   ProdType = 0;
   ProdTypeStr = “Generic”;
   ProdCode = 42;
   MajRev = 1;            $ Device Major Revision
   MinRev = 1;            $ Device Minor Revision
   ProdName = "Smart-Widget";
   Catalog = “1499-DVG”;
   Icon = “example.ico”;
```

Figure A.3 — [Device] section example (informative)

A.4.1.4.4 Device Classification section

The Device Classification section shall classify the device described by the EDS into one or more categories of devices. The entry keyword for all classifications shall consist of the character array, "Class", combined with a decimal number. The numbers shall start at 1 for the first class, and shall be incremented for each additional class.

The number of fields for each classification entry shall be variable to allow a tree classification structure similar to a file systems directory structure. Sub-classification of the public classifications shall be reserved. Vendor-specific classifications may be sub-classified at the discretion of the vendor. The first field shall represent the highest level in the tree structure and shall be one of the following field keywords:

— ControlNet;

— DeviceNet;

— EtherNetIP;

— a vendor-specific field keyword.

The vendor-specific field keyword shall begin with the Vendor ID of the company making the addition followed by an underscore (VendorID_VendorSpecificField). The VendorID shall be displayed in decimal and shall not contain leading zeroes. Each vendor is responsible for maintaining and documenting their vendor-specific field keyword.

A.4.1.4.5 Parameter Class section

The parameter class section shall identify general attributes of the configuration parameters described by the EDS, which correspond to a subset of the Parameter Object class attributes as defined in the CIP Object Library.

The parameter class section shall contain the entries specified in Table A.12.

Table A.12 — Parameter class format

Entry Name	Entry Keyword	Field Number	Data Type	Required/Optional
Max Instances	MaxInst	1	UINT	Required
Parameter Class Descriptor	Descriptor	1	WORD	Required
Configuration Assembly Instance	CfgAssembly	1	UINT	Required

The entries in the parameter class section shall provide the information as shown in Table A.13.

Table A.13 — Parameter class entries

Entries	Description
Max Instances	Identifies the total number of configuration parameters contained in the device associated with the EDS
Parameter Class Descriptor	Contains bit flags that describe the behavior of the device's parameter objects
Configuration Assembly Instance	Specifies the instance number of the Assembly Object that contains the device configuration data.

The Parameter Class Descriptor entry shall contain bits to describe parameter characteristics as defined in Table A.14. Bits not defined in Table A.14 shall not be used and shall be set to zero (0).

Table A.14 — Parameter class descriptor bit values

Bit	Name	Bit value and meaning
0	Supports individual parameter access	0 = NO parameter can be individually accessed. Only the Configuration assembly is used. 1 = Parameters can be individually accessed.
1	Supports full attributes	0 = Only the current value of a parameter is available within the device. 1 = All configuration data for a parameter is available within the device itself.
2	Non-volatile storage save command	0 = Parameters saved automatically. 1 = Parameters not saved automatically. Need to execute non-volatile storage save command when desired parameters to be saved in non-volatile storage.
3	Params are stored in non-volatile storage	0 = Parameters are not stored in non-volatile storage. 1 = All full parameters are stored in non-volatile storage.

Figure A.4 is an example that shows a typical Parameter Class Section.

```
[ParamClass]
   MaxInst = 3;
   Descriptor = 0x0E;
   CfgAssembly = 3;
```

Figure A.4 — [ParamClass] section example (informative)

A.4.1.4.6 Parameters section

The parameter section shall identify the configuration parameters in a device. The entry keyword shall be one of the following character arrays, “Param”, “ProxyParam”, “ProxiedParam”, combined with a parameter instance number (decimal) for the device, e.g. “Param1”. The actual parameter object instance may, but need not be, implemented in the device. Conversely, it is not required that ALL parameter object instances have a corresponding “ParamN” entry in an EDS. However, when a parameter object instance exists within a node, and if this parameter is also described within an EDS, then the value of “N” in “ParamN” shall be equal to the parameter object instance.

Each entry shall contain the formatted fields shown in Table A.15. The “ProxyParam”and “ProxiedParam” keywords are defined further in A.4.1.5.3.1, as part of the modular EDS requirements.

Table A.15 — Parameter format

Field Name	Field Number	Data Type	Required/Optional
Reserved	1	USINT	Required
Link Path Size	2	USINT	Optional
Link Path	3	EPATH	Optional
Descriptor	4	WORD	Required
Data Type	5	USINT/EPATH	Required
Data Size	6	USINT	Required
Parameter Name	7	EDS_Char_Array	Required
Units String	8	EDS_Char_Array	Required
Help String	9	EDS_Char_Array	Required
Minimum Value	10	data type	Conditional[a]
Maximum Value	11	data type	Conditional[a]
Default Value	12	data type	Required
Scaling Multiplier	13	UINT	Optional
Scaling Divider	14	UINT	Optional
Scaling Base	15	UINT	Optional
Scaling Offset	16	INT	Optional
Multiplier Link	17	UINT	Optional
Divisor Link	18	UINT	Optional
Base Link	19	UINT	Optional
Offset Link	20	UINT	Optional
Decimal Precision	21	USINT	Optional
International Parameter Name	22	STRINGI	Optional
International Engineering Units	23	STRINGI	Optional
International Help String	24	STRINGI	Optional
[a]These are further specified in Table A.19			

The entries in the parameter section shall provide the information as shown in Table A.16 and Table A.20.

Parameter fields listed in Table A.16 are common to all parameters.

Table A.16 — Common parameter fields

Fields	Description
Reserved	This first field shall contain a zero.
Link Path Size	The number of bytes used to represent path. If the link size does not agree with the number bytes in the "Link Path" field, then the "Link Size" shall be ignored. If this parameter is not addressable from the link, this field shall be empty. If this field is empty and the "Link Path" field is not, the "Link Size" shall be equal to the number of bytes in the "Link Path" field.
Link Path	CIP path to the object attribute from where the parameter value is retrieved. The path shall be entered as a character array, using the path notation described in IEC 61158-6:2003 (Type 2) and with the format as specified in A.4.1.3.6. If the parameter described by this ParamN entry is not directly addressable from the network, this field shall be empty. If this field contains a null string, "", the parameter described by this ParamN entry shall be addressable as the data attribute (instance attribute 1) of the Nth instance of the Parameter object (i.e. using path "20 0F 24 N 30 01")
Descriptor	The parameter descriptor. Contains bit flags that describe the behaviour of the individual parameters (see Table A.17)
Data Type	The data type identifier, as defined in IEC 61158-6:2003 (Type 2: Data Type Reporting). This identifier shall be encoded either as a USINT, or an EPATH. NOTE Old versions of EDS files may use USINT data type identifiers as specified in Table A.18, but these are now obsolete. They are provided here for compatibility reasons.
Data Size	The numeric data size value. For string and EPATH data types, this field specifies the number of bytes per character or entry. Therefore, for the STRING and EPATH data types, this value shall be specified as 1. For the STRING2 data type, this shall be specified as a 2. For the STRINGN data type, this shall be specified as the value of "N".
Parameter Name	The textual parameter name. If necessary, truncation of the retrieved text shall occur to meet the maximum character array length allowed.
Units String	The textual display units character array. If necessary, truncation of the retrieved text shall occur to meet the maximum character array length allowed.
Help String	The textual help character array. If necessary, truncation of the retrieved text occurs to meet the maximum character array length allowed.
Minimum Value	See Table A.19 for meaning and requirement based on parameter data type.
Maximum Value	See Table A.19 for meaning and requirement based on parameter data type.
Default Value	The default numeric value assigned to the parameter data value.
International Parameter Name	The parameter name expressed in STRINGI notation.
International Engineering Name	The engineering units expressed in STRINGI notation.
International Help String	The help string expressed in STRINGI notation.

The bits of the Descriptor field shall be as defined in Table A.17.

Table A.17 — Bit definitions of descriptor field

Bit	Definition	Bit value and meaning
0	Supports settable path	0 = Link path cannot be set. 1 = Link path can be set.
1	Supports enumerated strings	0 = Enumerated strings are not supported.µ 1 = Enumerated strings are supported and may be read
2	Supports scaling	0 = Scaling not supported. 1 = Scaling is supported. The scaling attributes are implemented and the value presented to the user in engineering units.
3	Supports scaling Links	0 = Scaling links not supported. 1 = The values for the scaling attributes may be retrieved from other parameters
4	Read only parameter	0 = Parameter value can be written (set) and read (get). 1 = The parameter value can only be read (get), and not set.
5	Monitor parameter	0 = Parameter value is not updated in real time by the device. 1 = The parameter value is updated in real time by the device.
6	Supports extended precision scaling	0= Extended precision scaling not supported. 1 = Extended precision scaling should be implemented and the value presented to the user in engineering units.
7	Support non-consecutive enumerated strings	0 = Non-consecutive enumerated strings not supported 1 = Non-consecutive enumerated strings are supported
8	Allow both enumeration and individual values	0 = Both enumeration and individual values are not supported 1 = Both enumeration and individual values are supported
9-15	Reserved	These bits are reserved and shall be set to 0.

Old versions of EDS files may use data type identifiers as specified in Table A.18.

Table A.18 — Data types identifiers (obsolete)

Data type identifier	Definition	Data type description
1	WORD	16-bit word
2	UINT	16-bit unsigned integer
3	INT	16-bit signed integer
4	BOOL	Boolean
5	SINT	Short integer
6	DINT	Double integer
7	LINT	Long integer
8	USINT	Unsigned short integer
9	UDINT	Unsigned double integer
10	ULINT	Unsigned long integer
11	REAL	Single floating point format (IEEE 754)
12	LREAL	Double floating point format (IEEE 754)
13	ITIME	Duration (short)
14	TIME	Duration
15	FTIME	Duration (high resolution)
16	LTIME	Duration (long)
17	DATE	Date
18	TIME_OF_DAY	Time of day
19	DATE_AND_TIME	Date and time
20	STRING	8-bit per character string
21	STRING2	16-bit per character string
22	STRINGN	N-byte per character string
23	SHORT_STRING	Short N-byte character string
24	BYTE	8-bit string
25	DWORD	32-bit string
26	LWORD	64-bit string

Table A.19 specifies the meaning and specific requirements for the minimum and maximum value entries, based on the parameter data type.

Table A.19 — Semantics for minimum and maximum value entries

Data Type	Description and Semantics	Minimum Value Semantics	Maximum Value Semantics	Required/ Optional/ Not Allowed
BYTE	Bit String – 8 bit length	The minimum and maximum values for these data types are not defined and shall not be specified in an EDS file.		Not allowed
WORD	Bit String – 16 bit length			
DWORD	Bit String – 32 bit length			
LWORD	Bit String – 64 bit length			
STRING[a]	String (2 byte length indicator, 1 byte per character)	Minimum string length	Maximum string length	Required
STRING2[a]	String (2 byte length indicator, 2 bytes per character)	Minimum string length	Maximum string length	Required
STRINGN[a]	String (2 byte length indicator, N bytes per character)	Minimum string length	Maximum string length	Required
SHORT_STRING[a]	Character string (1 byte length indicator, 1 byte characters)	Minimum string length	Maximum string length	Required
EPATH[a]	Enumerated Path	Minimum string length	Maximum string length	Optional
All Other Data Types		The minimum numeric value that may be assigned to the data value.	The maximum numeric value that may be assigned to the data value.	Optional[b]

[a]The STRING, STRING2, STRINGN, SHORT_STRING and EPATH data types do not have a minimum or maximum value specification. The minimum and maximum value fields are used to present the minimum and maximum string or path lengths. In these cases, the Data Size parameter is used to represent the number of bytes required per character or encoding entry.

[b]If the Minimum Value and/or Maximum Value is not specified then the minimum and/or maximum value for the parameter data value are as defined in IEC 61158-5:2003 (Type 2), based on the parameter data type.

Parameter fields listed in Table A.20 are optional and are only meaningful when used with the following data types : SINT, INT, DINT, LINT, USINT, UINT, UDINT, ULINT, REAL and LREAL. Specification of these fields with any other data type is prohibited.

Table A.20 — Parameter fields reserved for numeric data types

Fields	Description
Scaling Multiplier	The numeric multiplier value applied to the current parameter data value.
Scaling Divider	The numeric divisor value applied to the current parameter data value.
Scaling Base	The numeric base value applied to the current parameter data value.
Scaling Offset	The numeric offset value applied to the current parameter data value.
Multiplier Link	The parameter number pointing to a Parameter Object instance or other object attribute that contains the numeric multiplier value to apply to the current parameter data value.
Divisor Link	The parameter number pointing to a Parameter Object instance or other object attribute that contains the numeric divisor value to apply to the current parameter data value.
Base Link	The parameter number pointing to a Parameter Object instance or other object attribute that contains the numeric base value to apply to the current parameter data value.
Offset Link	The parameter number pointing to a Parameter Object instance or other object attribute that contains the numeric offset value to apply to the current parameter data value.
Decimal Precision	The numeric precision value applied to the current parameter data value.

Scaling shall not be performed by the device containing the parameter, but by the display tool. If scaling is supported, the display tool shall use the equation shown in Figure A.5 to determine the parameter engineering value (i.e. the value to display) from the parameter actual value. If scaling is not supported then the parameter value shall be displayed as is.

$$EngValue = \frac{(ActualValue + Offset) * Mult * Base}{Div * 10^{Decimal\ Precision}}$$

[a] If extended scaling is not supported, this formula shall be used with DecimalPrecision=0

Figure A.5 — Parameter scaling formula

A second keyword may also exist in the [Params] section. This keyword shall be used to provide an enumeration list of parameter choices to present to the user. The entry keyword for all enumerated parameters shall consist of the character array, “Enum”, combined with the decimal number from the corresponding Param entry. Each Enum entry shall consist of pairs of integers and strings.

Figure A.6 is an example that shows a typical Parameter section.

```
[Params]
   Param1 = 0, 1,"20 02", 0x0E94, 1, 1,"Preset","V","User Manual p33",
          0, 5, 1, 1, 1, 1, 0, 0, 0, 0, 0, 2;

   Param2 =            $ parameter instance
   0,                     $ First field shall equal 0
   6, "20 04 24 01 30 03", $ path size, path
   0x0A94,                $ descriptor - in hex format
   1,                     $ data type
   1,                     $ data size
   "Trigger",          $ name
   "Hz",                  $ units
   "User Manual p49",     $ help string
   0, 2, 0,               $ min, max, default data values
   1, 1, 1, 0,            $ mult, div, base, offset scaling
    ,  ,  ,  ,            $ mult, div, base, offset links not used
   2;                     $ decimal places

   Param3 =               $ not addressable from link
   0, , , 0x0082, 8, 1, "speed control", "", "", 3, 12, 3, ,,,,,,,,;

   Enum3  = 3, "stop", 8, "slow", 12, "fast";
```

Figure A.6 — [Params] section example

A.4.1.4.7 Parameter Groups section

The parameter groups section shall identify all of the parameter groups in a device. Each parameter group shall contain a list of the parameters in the group. The entry keyword for each group shall consist of the combination of the character array, “Group”, and a parameter group number (decimal), for example “Group1”. The decimal numbers shall start at one and increment by one.

The actual Parameter Group object instance may, but need not be, implemented in the device. Conversely, it is not required that ALL Parameter Group object instances have a corresponding “GroupN” entry in an EDS. However, when a Parameter Group object instance exists within a node, and if this Parameter Group is also described within an EDS, then the value of “N” in “GroupN” shall be equal to the Parameter Group object instance.

The fields in each entry shall contain the group name, the number of members in the group, and the instance numbers of the parameters in the group. The parameter group section shall contain the fields shown in Table A.21.

Table A.21 — Parameter group format

Field Name	Field Number	Data Type	Required/Optional
Group Name String	1	EDS_Char_Array	Required
Number of Members	2	UINT	Required
Parameter	3 through (number of members + 2)	UINT	Required

Figure A.7 is an example that shows a typical Parameter Groups section.

```
[Groups]
   Group1 = "Setup", 2, 1, 2;              $ group 1
   Group2 = "Monitor", 2, 2, 3;            $ group 2
   Group3 = "Maintenance", 2, 1, 3;        $ group 3
```

Figure A.7 — [Groups] section example

A.4.1.4.8 Assembly section

The Assembly section describes the structure of a data block. Often this block is the data attribute of an Assembly object; however, this section of the EDS can be used to describe any complex structure. The description of this data block parallels the mechanism that the Assembly object uses to describe its member list.

The "Revision" entry keyword shall have one 16-bit integer field that shall be the revision (class attribute 1) of the Assembly object within the device. If this optional entry is missing, the revision of the Assembly object shall be 2.

The entry keyword for all assemblies shall consist of one of the following character arrays, “Assem”, “ProxyAssem”, “ProxiedAssem”combined with the Assembly object instance number (decimal) for the device, e.g. “Assem1”. If a particular instance of the Assembly object is addressable from the link, there shall be a one-for-one pairing between the Assem number in the EDS file and the Assembly instance number in the device. The “ProxyAssem”and “ProxiedAssem” keywords are defined further in A.4.1.5.3.2, as part of the modular EDS requirements.

Each entry shall contain the formatted fields shown in Table A.22.

Table A.22 — AssemN keyword format

Field name	Field number	Data type	Required/ Optional
Name	1	EDS_Char_Array	Optional
Path	2	EDS_Char_Array	Optional
Size	3	UINT	Conditional
Descriptor	4	WORD	Optional
Reserved	5, 6	empty	
Member Size	7, 9, 11 …	UINT	Conditional
Member Reference	8, 10, 12 …	AssemN, ProxyAssemN, ParamN, ProxyParamN, UDINT, or EPATH	Conditional

The first field, called "Name", shall be a string giving a name to the data block. This optional field may be used by a user interface.

The second field, called "Path", shall be a string that specifies a logical path. This path shall identify the address of the data block within the device. If the block described by this AssemN entry is not directly addressable from the link, this field shall be empty. If this field is a null string, “”, the data block shall be addressable as the data attribute (instance attribute 3) of the Nth instance of the Assembly object.

The third field, called "Size", shall be the size of the data block in bytes. If neither this field nor the "Member Size"/"Member Reference" fields are present the size of the data block shall be 0. Both of these fields may be present; however, since they both specify the size of the block, the sizes specified by both means shall agree.

The fourth field, “Descriptor”, shall be a bit-field that describes certain properties of the Assembly. The bits of this field shall be interpreted as specified in Table A.23.

Table A.23 — Bit definition of Assembly descriptor field

Bit	Name	Meaning
0	Allow Value Edit	If this bit is set (1), the contents of the Assembly’s member references fields defined as values may be edited. If reset (0), the contents of these member references fields may not be edited. If this field is empty, the meaning shall default to reset (0). The member references considered to be values are those specifying either a UDINT constant, or a path composed of Data Segments.
1-15		Reserved

Fields 5 and 6 shall be reserved and shall be empty.

The remaining fields shall be paired such that a "Member Size" field is paired with a "Member Reference" field making the total number of fields even. The number of fields pairs in each entry shall be variable. The pairs shall correspond to an Assembly object member list.

The allowed values for the "Member Reference" field shall be one of the following:

— a ParamN or ProxyParamN reference from the [Params] section;

— an AssemN or ProxyAssemN reference from the [Assembly] section;

— a string representing a path (EPATH);

— a UDINT constant;

— an empty field;

— additional values as defined for modular EDS in A.4.1.5.3.2.

If the "Member Reference" field is empty, the number of bits specified by the "Member Size" field shall be used as a pad within the Assembly object. A "Member Reference" field containing a null string shall be treated as if the field was empty. A "Member Reference" field and its corresponding "Member Size" shall not both be empty. If the "Member Reference" field specifies an EPATH, this path shall consist of either Logical Segments (path to an object within the device) or Data Segments.

The "Member Size" field shall have units of bits. If a "Member Size" field is empty, the defined size of the corresponding "Member Reference" field shall be used. The defined size of a Param entry shall be as given in its 6th field (size). The defined size of an Assem entry shall be as given in its 3rd field (size).

The members shall be placed into the data block least significant bit first just as they are in the Assembly object. If a "Member Size" field is smaller than the defined size of the corresponding "Member Reference" field, the least significant bits of the corresponding "Member Reference" field shall be used. If a "Member Size" field is larger than the defined size of the corresponding "Member Reference" field, the entire member shall be followed by zero pads to extend the member to the "Member Size". The data block represented shall be an integer number of bytes. The total of all member sizes shall equal the AssemN Size field (when expressed as bits).

Figure A.8 is an example that shows a typical Assembly section. In this example, Assem5 is 1 byte long and has a default value of 0x21.

```
[Params]
   Param1 =
          0,                      $ first field shall equal 0
          6, "20 0F 24 01 30 01", $ path size, path
          0x0000,                 $ descriptor
          2,                      $ data type : 16-bit WORD
          2,                      $ data size in bytes
          "Idle state",           $ name
          "",                     $ units
          "User Manual p48",      $ help string
          0, 2, 1,                $ min, max, default data values
          0, 0, 0, 0,             $ mult, dev, base, offset scaling not used
          0, 0, 0, 0,             $ mult, dev, base, offset link not used
          0;                      $ decimal places not used

   Param2 =
          0, 6, "20 0F 24 02 30 01", $ path size, path
          0x0000, 2, 2,
          "Fault state", "", "User Manual p49",
          0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0;

[Assembly]
   Revision = 2;

   Assem5 = "configuration", "20 04 24 05 30 03",1,,,,
            4, Param1,
            3, Param2,
            1, ;
```

Figure A.8 — [Assembly] section example

NOTE The keyword Variant, combined with a decimal number (e.g. “Variant1”) is reserved for the future definition of new entry types in the Assembly section.

A.4.1.4.9 Connection Manager section

A.4.1.4.9.1 Contents

The Connection Manager section shall contain information concerning the number of types of application connections a device supports. This section is modeled after the Connection Manager Object. Many of the terms used here are described in IEC 61158-5:2003 and IEC 61158-6:2003 (Type 2). Each entry keyword shall be one of the following character arrays, “Connection”, “ProxyConnect”, “ProxiedConnect”, combined with a number (decimal), for example, “Connection1”, “ProxyConnect1”, or “ProxiedConnect1”. The decimal numbers shall start at 1 and increment for each additional “Connection” entry. The decimal number shall not be required to start at 1 or increment for each additional “ProxyConnect” or “ProxiedConnect” entry. The “ProxyConnect”and “ProxiedConnect” keywords are defined further in A.4.1.5.3.3, as part of the modular EDS requirements.

Each entry shall contain the formatted fields shown in Table A.24.

Table A.24 — Connection Manager format

Field name	Field number	Data type	Required/Optional
Trigger and transport	1	DWORD	Required
Connection parameters	2	DWORD	Required
O=>T RPI	3	UDINT or ParamN or ProxyParamN	Optional
O=>T size	4	UINT or ParamN or ProxyParamN	Conditional
O=>T format	5	ParamN or ProxyParamN or AssemN or ProxyAssemN	Conditional
T=>O RPI	6	UDINT or ParamN or ProxyParamN	Optional
T=>O size	7	UINT or ParamN or ProxyParamN	Conditional
T=>O format	8	ParamN or ProxyParamN or AssemN or ProxyAssemN	Conditional
config #1 size	9	UINT or ParamN or ProxyParamN	Optional
config #1 format	10	ParamN or ProxyParamN or AssemN or ProxyAssemN	Optional
config #2 size	11	UINT or ParamN or ProxyParamN	Optional
config #2 format	12	ParamN or ProxyParamN or AssemN or ProxyAssemN	Optional
Connection name string	13	EDS_Char_Array	Required
Help string	14	EDS_Char_Array	Required
Path	15	EDS_Char_Array	Required

A.4.1.4.9.2 Trigger and transport mask

The bit assignments for the trigger and transport mask shall be as shown in Table A.25. A bit shall be set to a 1 (on) for each trigger mode the connection supports. All other bits shall be set to a 0 (off). For the client/server bit : 0=client, 1=server. Only one of the transport types shall be set to a 1 (on).

Table A.25 — Trigger and transport mask bit assignments

Bit	Bit definition
0	class 0: null
1	class 1: duplicate detect
2	class 2: acknowledged
3	class 3: verified
4	class 4: non-blocking
5	class 5: non-blocking, fragmenting
6	class 6: multicast, fragmenting
7-15	class: reserved
16	trigger: cyclic
17	trigger: change of state
18	trigger: application
19-23	trigger: reserved
24	transport type: listen-only
25	transport type: input-only
26	transport type: exclusive-owner
27	transport type: redundant-owner
28-30	reserved
31	client = 0 / server = 1

A.4.1.4.9.3 Connection parameters

The bit assignments for the connection type and priority mask shall be as shown in Table A.26. A bit shall be set to a 1 (on) for each connection type and priority the connection supports. All other bits shall be set to a 0 (off).

Table A.26 — Connection parameters bit assignments

Bit	Bit definition
0	O=>T fixed size supported
1	O=>T variable size supported
2	T=>O fixed size supported
3	T=>O variable size supported
4 – 5	O=>T number of bytes per slot in the O=>T real time data packet for adapter rack connections: 0 = 1 byte 1 = 2 bytes 2 = 4 bytes 3 = 8 bytes
6 – 7	T=>O number of bytes per slot in the T=>O real time data packet for adapter rack connections: 0 = 1 byte 1 = 2 bytes 2 = 4 bytes 3 = 8 bytes
8 – 10	O=>T Real time transfer format. 0 = connection is pure data and is modeless 1 = use zero data length packet to indicate idle mode 2 = reserved 3 = heartbeat 4 = 32-bit run/idle header 5 thru 7 are reserved
11	Reserved
12 – 14	T=>O Real time transfer format 0 = connection is pure data and is modeless 1 = use zero data length packet to indicate idle mode 2 = reserved 3 = heartbeat 4 = 32-bit run/idle header 5 thru 7 are reserved
15	reserved
16	O=>T connection type: NULL
17	O=>T connection type: MULTICAST
18	O=>T connection type: POINT2POINT
19	O=>T connection type: reserved
20	T=>O connection type: NULL
21	T=>O connection type: MULTICAST
22	T=>O connection type: POINT2POINT
23	T=>O connection type: reserved
24	O=>T priority: LOW
25	O=>T priority: HIGH
26	O=>T priority: SCHEDULED
27	O=>T priority: reserved
28	T=>O priority: LOW
29	T=>O priority: HIGH
30	T=>O priority: SCHEDULED
31	T=>O priority: reserved

A.4.1.4.9.4 O=>T RPI

The O=>T RPI shall be the number of microseconds of the requested packet interval. The O=>T RPI shall be a UDINT or a Param or ProxyParam entry from the [Params] section that evaluates to a UDINT. If this field is empty, no constraints are placed on the O=>T RPI.

A.4.1.4.9.5 O=>T size

The O=>T size shall be the number of bytes delivered to the target transport. It shall not include the transport sequence count. The O=>T size shall be a UINT or a Param or ProxyParam entry from the [Params] section that evaluates to a UINT. If this field is empty, the defined size of the O=>T format shall be used after the optional run/idle header size is added.

A.4.1.4.9.6 O=>T format

The O=>T format entry shall define the structure of the consumer buffer for this connection. Valid format descriptors shall be identifiers within the EDS file including

— a Param or ProxyParam entry from the [Params] section;

— an Assem or ProxyAssem entry from the [Assembly] section.

This field may be empty indicating that the consuming format is not specified. This field shall not be empty if the O=>T size field is empty. The O=>T format shall not include the 32-bit run/idle header if it is present.

A.4.1.4.9.7 T=>O RPI

The T=>O RPI shall be the number of microseconds of the requested packet interval. The T=>O RPI shall be a UDINT or a Param or ProxyParam entry from the [Params] section that evaluates to a UDINT. If this field is empty, no constraints are placed on the T=>O RPI.

A.4.1.4.9.8 T=>O size

The T=>O size shall be the number of bytes produced by the target transport. It shall not include the transport sequence count. The T=>O size shall be a UINT or a Param or ProxyParam entry from the [Params] section that evaluates to a UINT. If this field is empty, the defined size of the T=>O format shall be used after the optional run/idle header is added.

A.4.1.4.9.9 T=>O format

The T=>O format entry shall define the structure of the producer buffer for this connection. Valid format descriptors shall be identifiers within the EDS file including

— a Param or ProxyParam entry from the [Params] section;

— an Assem or ProxyAssem entry from the [Assembly] section.

This field may be empty indicating that the producing format is not specified. This field shall not be empty if the T=>O size field is empty. The format shall include the status header if it is present.

A.4.1.4.9.10 Configuration

The config #1 size and config #2 size shall specify the size of the optional data segment that is appended to the path in the Forward_Open. The data segment shall be the concatenation of the two buffers described by the config #1 format and config #2 format. The sizes shall be the number of bytes and shall be a UINT or a Param or ProxyParam entry from the [Params] section that evaluates to a UINT. If one of the config size fields is empty, the natural size of the corresponding config format field shall be used.

Valid config format fields shall be identifiers within the EDS file including

— a Param or ProxyParam entry from the [Params] section;

— an Assem or ProxyAssem entry from the [Assembly] section.

The config format fields may be empty indicating that the config format is not specified. If both the config size and config format fields are empty, no data segment shall be appended to the path of the Forward_Open.

A.4.1.4.9.11 Connection name string

A tool may display the connection name string (character array). The connection name string shall be unique among all Connection entries within the EDS.

A.4.1.4.9.12 Help string

A tool may display the textual help character array. If no help string is to be provided a "null" string shall be used where a null string is defined as two double quotations "" with no characters between the quotation marks.

A.4.1.4.9.13 Path

A path referencing the target object. The path shall be entered as a CIP Path (EPATH), using the padded path notation described in IEC 61158-6:2003 (Type 2) and with the format as specified in A.4.1.3.6. In addition to the format specified in A.4.1.3.6, the path field may also contain the other following references:

— Param or ProxyParam entries from the [Params] section;

— the keyword SLOT;

— the keyword SYMBOL_ANSI;

— the keyword SLOT_MINUS_ONE.

The Param/ProxyParam entries shall evaluate to a USINT, UINT or UDINT. The value of the Param/ProxyParam shall be used in a little endian order for insertion into the path. The Param/ProxyParam references within the path may be enclosed in brackets as shown in Figure A.9. When enclosed in brackets, the value of the Param/ProxyParam shall be local to the path – the same Param/ProxyParam entry may have a different value elsewhere in the EDS. If the Param/ProxyParam is not enclosed in brackets, the value shall be the same everywhere within the EDS.

The keyword SLOT shall always evaluate to a USINT. The values substituted for the SLOT keyword shall correspond to the position of a module in a backplane.

The keyword SLOT_MINUS_ONE shall always evaluate to a USINT. The values substituted for the SLOT_MINUS_ONE keyword shall correspond to the position of a module in the backplane minus 1.

The keyword SYMBOL_ANSI shall evaluate to an extended symbolic segment (see IEC 61158-6:2003 (Type 2)) entered through the user interface. The extended symbol segment shall be an ANSI extended symbol (CIP path type = 0x91). For example, the string "CAB" shall evaluate to the following extended symbol segment (padded) : 0x91 0x03 0x43 0x41 0x42 0x00.

A.4.1.4.9.14 Example Connection Manager section (informative)

Figure A.9 is an example that shows a typical Connection Manager section.

```
[Params]
   Param1 =              $ specifies read buffer
      0, , ,             $ no path means not directly accessible
      0x0004,            $ descriptor : support scaling
      8, 1,              $ USINT, 1 byte
      "Read",            $ name
      "", "",            $ units & help string
      64, 95, 64,        $ min, max, default data values
      1, 1, 1, -63,      $ mult, div, base, offset scaling
      0, 0, 0, 0, 0;     $ mult, div, base, offset link & decimal
                         $(not used)

   Param2 =              $ specifies write buffer
      0, , ,             $ no path means not directly accessible
      0x0004,            $ descriptor : support scaling
      8, 1,              $ USINT, 1 byte
      "Write",           $ name
      "", "",            $ units & help string
      160, 191, 160,     $ min, max, default data values
      1, 1, 1, -159,     $ mult, div, base, offset scaling
      0, 0, 0, 0, 0;     $ mult, div, base, offset link & decimal
                         $(not used)

[Connection Manager]
   Connection1 =
      0x04010002,        $ trigger & transport
                         $   class 1, cyclic, exclusive-owner
      0x44244401,        $ point/multicast & priority & realtime format
                         $   fixed, 32-bit headers, scheduled,
                         $   O=>T point-to-point, T=>O multicast
      , 16, ,            $ O=>T RPI, size, format
      , 12, ,            $ T=>O RPI, size, format
      , ,                $ config part 1 (not used)
      , ,                $ config part 2 (not used)
      "read/write",      $ connection name
      "",                $ Help string
      "20 04 24 01 2C [Param2] 2C [Param1]";
```

Figure A.9 — [Connection Manager] section example

A.4.1.4.10 Port section

The Port section shall describe the CIP routable ports available within a device. Every CIP routable port shall have a corresponding entry in this section. The entry keyword for all ports shall consist of the character array “Port”, combined with a decimal number corresponding to an instance of the port object. For example, Port1 is instance 1 of the Port Object.

NOTE A CIP routable port is a port that is able to exchange CIP messages with another CIP port connected to another CIP link

Each entry shall contain the formatted fields shown in Table A.27.

Table A.27 — Port entry format

Field Name	Field number	Data type	Required/Optional
Port Type Name	1	Field Keyword	Required
Port Name	2	EDS_Char_Array	Optional
Port Object	3	EDS_Char_Array	Optional
Port Number	4	UINT	Required
Reserved	5, 6	empty	Not Used
Port Specific	7, 8, ...	Port Specific	Port Specific

The first field, called “Port Type Name”, shall be one of the following field keywords:

— ControlNet;

— ControlNet_Redundant;

— TCP (to indicate an EtherNet/IP capable TCP port);

— DeviceNet;

— a vendor-specific field keyword beginning with the device’s Vendor ID and an underscore character (‘65535_’).

The optional “Port Name” field shall be a string giving a name to the port, and may be used by a user interface. The “Port Object” field shall be a path (EPATH) that identifies the network specific link object associated with the port.

The port number 1 shall correspond to the backplane “port”. Devices with a backplane that cannot route CIP messages shall not have a port number 1.

Figure A.10 is an example that shows a typical Port section.

```
[Port]
   Port1 =   DeviceNet,
             “Port A”,          $ name of port
             “20 03 24 01”,     $ instance one of the DeviceNet object
             2;                 $ port number 2

   Port2 =   65535_Chassis,
             “Chassis”,    $ name of port
             “20 9A 24 01”,     $ vendor specific backplane object
             1;                 $ port number 1
```

Figure A.10 — [Port] section example

A.4.1.5 Modular EDS file requirements

A.4.1.5.1 General

This subclause describes the concept and contents of a Modular EDS and specifies the usage requirements.

A.4.1.5.2 Modular section

A.4.1.5.2.1 Contents

The [Modular] section shall describe a chassis based system. The two types of modular devices shall be:

— chassis;

— module.

A.4.1.5.2.2 Chassis device

A [Modular] section that describes a chassis shall contain a required keyword “DefineSlotsInRack”. The single field on this entry shall be a 16-bit unsigned integer (UINT) indicating the number of slots in the chassis. Even though an electronic key is defined for the chassis, it need not be addressable from the link. The SLOT keyword used in path definitions in the [Connection Manager] section shall range from 0 to the number of slots minus 1.

The keyword “SlotDisplayRule” is optional. The single field on this entry shall be a parameter from the [Params] section (ParamN only) which defines the translation between internal and external slot number.

Figure A.11 is an example that shows an EDS for a chassis device, including a Modular section.

```
[File]
   DescText = "Wonder Chassis EDS file";
   CreateDate = 09-01-1997;
   CreateTime = 17:23:00;
   Revision = 1.1;

[Device]
   VendCode = 65535;
   VendName = "Widget Works, Inc.";
   ProdType = 101;
   ProdTypeStr = "Widget Works Generic";
   ProdCode = 1;
   MajRev = 1;
   MinRev = 1;
   ProdName = "Widget Chassis";
   Catalog = "1234-chassis";

[Params]
   Param1 =
      0,                          $ first field shall equal 0
      ,,                          $ path size,path
      0x0004,                     $ descriptor
      8,                          $ data type: 32-bit Unsigned Long Integer
      1,                          $ data size in bytes
      "Slot Naming Convention",   $ name
      "",                         $ units
      "",                         $ help string
      0,4,0,                      $ min,max,default data values
      0,0,0,0,                    $ mult,dev,base,offset scaling
      0,0,0,0,                    $ mult,dev,base,offset link not used
      0;                          $ decimal places not used

   Enum1 =   0,"n/a",1,"0",2,"1",3,"2",4,"3";

[Modular]
   DefineSlotsInRack    = 5;
   SlotDisplayRule      = Param1;
```

Figure A.11 — [Modular] section describing a chassis

A.4.1.5.2.3 Module device (basic entries)

A [Modular] section that describes a module shall contain the "Width" and "Rack" entries.

The required entry with the keyword "Width" shall have a single field that indicates how many slots of the chassis are consumed by the module. The field shall be a 16-bit unsigned integer (UINT).

The entry keyword for all chassises, into which the module can be placed, shall consist of the character array, "Rack", combined with a decimal number. The numbers shall start at 1 for the first chassis, and shall be incremented for each additional chassis. The fields for the "Rack" entries shall be as shown in Table A.28.

Table A.28 — Rack entry format

Field name	Field number	Data type	Required/Optional
Vendor ID	1	UINT	Required
Product Type	2	UINT	Required
Product Code	3	UINT	Required
Major Revision	4	USINT	Required
Minor Revision	5	USINT	Required
reserved	6, 7, 8	empty	not used
Legal Slot	9, 10, 11 ...	UINT	Required

The "Vendor ID", "Product Type", "Product Code", "Major Revision" and "Minor Revision" field shall identify the electronic key of the chassis into which the module may be placed. The reserved field shall be empty. The "Legal Slot" fields shall identify the slots into which the module may be placed. The EDS for the module shall contain one "Rack" entry for each chassis into which the module may be placed.

Figure A.12 is an example that shows a typical [Modular] section.

```
[Modular]
   Width = 1;

   Rack1 =                    $ this module can plug into
      65535, 101, 1, 1, 1,,,  $ slots 1, 2, 3 and 4 of
      1, 2, 3, 4;             $ this five slot chassis
```

Figure A.12 — [Modular] section example

A.4.1.5.2.4 Module device (additional entries)

Overview

Additional entries are defined in the EDS to allow device identification and device keying for modules in chassis-based systems which do not support CIP.

For that purpose, modular devices are typically divided into two categories:

— modules that have a CIP link connection, a corresponding Identity object addressable from the link, and are placed in slot 0 (e.g. communication adapters);

— modules that do not have a CIP link connection or an addressable Identity object, and therefore may not be placed in slot 0 (e.g. I/O modules).

NOTE CIP provides other mechanisms for device identification and device keying for modules supporting a CIP link addressable Identity object.

Entries for a module that does not have a link addressable Identity object

A [Modular] section that describes a module that does not have a link addressable Identity object may contain the entry keyword "ExternalID". The keyword shall have a single field. This field shall be a byte string that identifies the module. This byte string shall be encoded using the same format as specified for an EPATH.

Figure A.13 is an example that shows a typical [Modular] section describing a module without a link addressable Identity object.

```
[Modular]
   Width = 1;

   Rack1 =                          $ this module can plug into
      65535, 101, 1, 1, 1,,,,       $ slots 1, 2, 3 and 4 of
      1, 2, 3, 4;                   $ this five slot chassis

   Rack2 =
      65535, 101, 2, 1, 1,,,,
      1, 2, 3, 4, 5, 6, 7;

   ExternalID = “12 34”;
```

Figure A.13 — [Modular] section example (module without a link addressable Identity object)

Entries for a module that has a link connection and is placed in slot 0

A [Modular] section that describes a module that has a link connection and is placed in slot 0 may contain any of the following entry keywords, or a combination of them.

The keyword “GenericID” shall have a single field. This field shall be a byte string that shall be included in the data segment for a module connection in place of the ExternalID when no module keying is desired. This byte string shall be encoded using the same format as specified for an EPATH.

The keyword “ExternIDExactMatch” shall have a single field, with a value of Yes or No. Yes shall indicate that the ExternalID specifies one specific device. No shall indicate that the ExternalID specifies one of a set of compatible devices. If the “ExternIDExactMatch” keyword is omitted, the default condition shall be that the ExternalID specifies one specific device.

The keyword “Query” shall have 4 fields. The first field shall be a path that identifies a link addressable attribute that contains an array of external identifiers, one for each slot in the chassis except slot 0. The second field shall be the service to use with the query path (i.e. 1 – get attribute all or 14 – get attribute single). The third field shall be an integer that determines the number of bytes used to identify each module and shall be in the range 1 to 16. If a double slot module is in the chassis, the external identifier for the module shall appear twice in the array returned from a query. A query shall only be addressed to a module in slot 0. The fourth field shall be the ExternalID returned when an empty slot exists, encoded using the same format as specified for an EPATH.

Figure A.14 is an example that shows a typical [Modular] section describing a module with a link connection placed in slot 0.

```
[Modular]
  Width = 1;

  Rack1 =                        $ this module can only plug into
     65535, 101, 1, 1, 1,,,,    $ slot 0 of this five slot chassis
     0;

  Rack2 = 65535, 101, 2, 1, 1,,,,      0;

  Query = “20 04 24 07 30 03”,1,2,”FF FF”;

  GenericID = “00 00”;

  ExternalIDExactMatch = No;
```

Figure A.14 — [Modular] section example (module with a link connection in slot 0)

A.4.1.5.3 Modular additions to basic EDS sections

A.4.1.5.3.1 Additions to the Parameter section

The “ProxyParam” and “ProxiedParam” keywords shall be used to describe parameters that are proxied by a ControlNet adapter device to another device that does not support the CIP protocol. An example of this is a ControlNet adapter module (the device proxying the connection) in a multiple slot I/O rack with an analog I/O module (the device the connection is proxied for).

The “ProxyParam” shall exist in the EDS for the device that performs the proxy.

The “ProxiedParam” keyword shall exist in the EDS for the device that the proxy is performed for.

The information in the [Modular] section shall be used to associate EDS files containing “ProxyParam” keywords to EDS files containing “ProxiedParam” keywords. This association shall exist when both EDS files specify a matching Rack entry.

The decimal number (that is combined with “ProxyParam” and “ProxiedParam”) shall be used to match a “ProxyParam” to a “ProxiedParam”. The field values of a matched “ProxyParam” and “ProxiedParam” pair shall be combined to constitute the same field value information that exists in a single “Param” entry. This combination shall be done by using the field value from the “ProxyParam” unless that field value is the keyword “Module”. When the field value specified in the “ProxyParam” is “Module” the field value specified in the “ProxiedParam” shall be used. It shall be legal to specify field values for “ProxiedParam” entries whose corresponding field value in the “ProxyParam” is not “Module”, however, these field value shall not by used, they shall exist only for documentation.

Another keyword may also exist in the [Params] section. This keyword shall be used to provide minimum, maximum and default values to be added to the “ProxyParam” minimum, maximum and default values. This entry keyword shall be “ProxyParamSizeAdder”, combined with the decimal number from the corresponding “ProxyParam” entry. Each “ProxyParam” entry shall consist of a Minimum Value, Maximum Value and Default Value fields. The definition of these fields matches the “Param” definitions. The “ProxyParamSizeAdder” keyword provides a means for an adapter on a module connection (e.g. “ProxyConnect”) to add adapter data to the module data and return the combined data on the connection.

Another keyword may also exist in the [Param] section that corresponds to the “ProxyParam”, “ProxyEnum”. “ProxyEnum” has the same definition as “Enum” except it is associated with “ProxyParam” instead of “Param”. A second keyword may also exist in the [Param] section that corresponds to the “ProxiedParam”, “ProxiedEnum”. “ProxiedEnum” has the same definition as “Enum” except it is associated with “ProxiedParam” instead of “Param”.

A.4.1.5.3.2 Additions to the Assembly section

Additional entry keywords

The “ProxyAssem” and “ProxiedAssem” keywords shall be used to describe assemblies that are proxied by a CIP adapter device to another device that does not support the CIP protocol. An example of this is a ControlNet adapter module (the device proxying the connection) in a multiple slot I/O rack with an analog I/O module (the device the connection is proxied for).

The “ProxyAssem” keyword shall exist in the EDS for the device that performs the proxy; the “ProxiedAssem” keyword shall exist in the EDS for the device that the proxy is performed for.

The information in the [Modular] section shall be used to associate EDS files containing “ProxyAssem” keywords to EDS files containing “ProxiedAssem” keywords. This association shall exist when both EDS files specify a matching Rack entry.

The decimal number (that is combined with “ProxyAssem” and “ProxiedAssem”) shall be used to match a “ProxyAssem” to a “ProxiedAssem”. The field values of a matched “ProxyAssem” and “ProxiedAssem” pair shall be combined to constitute the same field value information that exists in a single “Assem” entry. This combination shall be done by using the field value from the “ProxyAssem” unless that field value is one of the keywords “Module” or “ModuleMemberList”. When the field value specified in the “ProxyAssem” is “Module” the field value specified in the “ProxiedAssem” shall be used. The field value “Module” shall not be used for “Member Size” or “Member Reference” fields. “ModuleMemberList” shall only be used in place of a “Member Size” and “Member Reference” field pair. When the field value specified in the “ProxyAssem” is “ModuleMemberList” all “Member Size” and “Member Reference” fields specified in the “ProxiedAssem” shall be used. It shall be legal to specify field values for “ProxiedAssem” entries whose corresponding field value in the “ProxyAssem” is not “Module”, however, these field values shall not by used, they shall exist only for documentation.

Additional field keywords

An adapter rack connection is a connection to a rack based adapter device that includes data from modules in the rack. Such a connection may also be used to send configuration and keying data for modules in the rack (e.g. at connection establishment).

The following keywords are additional values allowed for the "Member Reference" field within the Assembly section, that indicate the special purpose intended for the use of the data defined by an assembly member:

— ExternalID;

— InputSlotMask0 or InputSlotMask1;

— OutputSlotMask0 or OutputSlotMask1;

— ConfigSlotMask0 or ConfigSlotMask1.

The “ExternalID” keyword specifies that this assembly member shall contain either a module device “ExternalID” value if device keying is desired, or the “GenericID” value defined in the adapter EDS if module keying is not desired.

The “ExternalID” keyword combined with a decimal number (e.g. ExternalID2) shall be used to allow individual device keying for adapter rack connections. The decimal (positive) number N in “ExternalIDN” specifies slot N in the rack. The “ExternalIDN” keyword specifies that this assembly member shall contain either a module device “ExternalID” value for slot N if device keying is desired for this slot, or the “GenericID” value defined in the adapter EDS if module keying is not desired for this slot.

NOTE Keying is not available for slot 0.

The “InputSlotMask0” or “InputSlotMask1” keyword shall indicate the location of the input slot mask in the assembly. An input slot mask is an array of bits which represents inclusion or exclusion of a module’s target to

originator data in an adapter rack connection. If “InputSlotMask0” keyword is used, bit 0 in this array represents slot 0, bit 1 represents slot 1, and so on. If “InputSlotMask1” keyword is used, bit 0 in this array represents slot 1, bit 1 represents slot 2, and so on. “InputSlotMask0” and “InputSlotMask1” shall not be used both in the same assembly. The preceding “Member size” field shall be required.

The “OutputSlotMask0” or “OutputSlotMask1” keyword shall indicate the location of the output slot mask in the assembly. An output slot mask is an array of bits which represents inclusion or exclusion of a module’s originator to target data in an adapter rack connection. If “OutputSlotMask0” keyword is used, bit 0 in this array represents slot 0, bit 1 represents slot 1, and so on. If “OutputSlotMask1” keyword is used, bit 0 in this array represents slot 1, bit 1 represents slot 2, and so on. “OutputSlotMask0” and “OutputSlotMask1” shall not be used both in the same assembly. The preceding “Member size” field shall be required.

The “ConfigSlotMask0” or “ConfigSlotMask1” keyword shall indicate the location of the configuration slot mask in the assembly. A configuration slot mask is an array of bits which represents inclusion or exclusion of a module’s configuration data in the establishement service of an adapter rack connection. If “ConfigSlotMask0” keyword is used, bit 0 in this array represents slot 0, bit 1 represents slot 1, and so on. If “ConfigSlotMask1” keyword is used, bit 0 in this array represents slot 1, bit 1 represents slot 2, and so on. “ConfigSlotMask0” and “ConfigSlotMask1” shall not be used both in the same assembly. The preceding “Member size” field shall be required.

A.4.1.5.3.3 Additions to the Connection Manager section

The “ProxyConnect” and “ProxiedConnect” keywords shall be used to describe connections that are proxied by a CIP adapter device to another device that does not support the CIP protocol. An example of this is a ControlNet adapter module (the device proxying the connection) in a multiple slot I/O rack with an analog I/O module (the device the connection is proxied for).

The “ProxyConnect” keyword entry shall exist in the EDS for the device that performs the proxy. In the example above, this would be the ControlNet adapter module.

The “ProxiedConnect” keyword entry shall exist in the EDS for the device that the proxy is performed for. In the example above, this would be the analog I/O module.

The information in the [Modular] section shall be used to associate EDS files containing “ProxyConnect” keywords to EDS files containing “ProxiedConnect” keywords. This association shall exist when both EDS files specify a matching Rack entry.

The decimal number (that is combined with “ProxyConnect” and “ProxiedConnect”) shall be used to match a “ProxyConnect” to a “ProxiedConnect”. The field values of a matched “ProxyConnect” and “ProxiedConnect” pair shall be combined to constitute the same field value information that exists in a single “Connection” entry. This combination shall be done by using the field values from the “ProxyConnect” except for those fields where the value is the keyword “Module”. In those cases, the field value specified in the associated “ProxiedConnect” shall be used. It shall be legal to specify field values for “ProxiedConnect” entries whose corresponding field value in the “ProxyConnect” entry is not “Module”, however, these field values shall not be used, they shall exist only for documentation. The field value for the “ProxyConnect” “connection name string” field shall not be “Module”, the “ProxyConnect” shall always specify the “connection name string”.

A.4.1.5.3.4 Examples of extended EDS section (informative)

Figure A.15 and Figure A.16 are examples that show the usage of the modular EDS extensions for the Parameter, Assembly and Connection Manager sections.

```
[Params]
   Param1 =          0,,,0x0010,2,2,” Target Error Codes",
                           "","",0,0xFFFF,0,0,0,0,0,0,0,0,0,0;
   ProxyParam1 =     0,,,0x0000,2,2,"input size",
                     "","",Module,Module,Module,0,0,0,0,,,,,0;
   ProxyParamSizeAdder1 = 4,4,4;

[Assembly]
   Assem1 = "connection input format",,,,,,
      32,Param1,
      ,ProxyAssem1,
      ,ProxyAssem2;
   ProxyAssem1 = "real time input format","20 7D 24 SLOT 30 0A",,,,,
      ModuleMemberList;
   ProxyAssem2 = "real time status format","20 7D 24 SLOT 30 0B",,,,,
      ModuleMemberList;

[Connection Manager]
   ProxyConnect1 = 0x010100002, 0x44244401,
         2, 0, ,  2, ProxyParam1, Assem1, , , , , "Listen Only", "",
         "01 SLOT_MINUS_ONE 20 04 24 03 2C 04 2C 02";
```

Figure A.15 — Example of ProxyParam and ProxyAssem entries

```
[Params]
   ProxiedParam1 = ,,,,,,"input size","","",0,2,2,,,,,,,,;

[Assembly]
   ProxiedAssem1 = "real time input format",,,,;
   ProxiedAssem2 = "real time status format",,,,,,16,;

[Connection Manager]
   ProxiedConnect1 = ,,,0,,,,,,,,,,;
```

Figure A.16 — Example of matching ProxiedParam and ProxiedAssem entries

A.4.2 ControlNet specific EDS requirements

A.4.2.1 ControlNet EDS content

This subclause specifies the file encoding requirements of the Electronic Data Sheet (EDS) which are specific for ControlNet-based networks.

Table A.29 summarizes the structure of the sections which may be present in a ControlNet EDS, the corresponding legal section delimiters, and the order of these sections in an EDS. Some of these sections are common to several CIP-based networks, their specific use for ControlNet is further specified in A.4.2.2 if needed. Other sections are specific to ControlNet and are specified in A.4.2.4.

Table A.29 — ControlNet EDS file structure

Section Name	Legal Delimiter	Placement	Required/Optional
File Description	[File]	1	Required
Device Description	[Device]	2	Required
Device Classification	[Device Classification]	[a]	Required
Parameter Class	[ParamClass]	[a]	Optional
Parameters	[Params]	[a]	Optional
Parameter Groups	[Groups]	[a]	Optional
Assembly	[Assembly]	[a]	Optional
Connection Characteristics	[Connection Manager]	[a]	Conditional[b]
Port	[Port]	[a]	Optional
Modular	[Modular]	[a]	Optional
Physical Layer	[ControlNet Physical Layer]	*	Conditional[b]
Keeper	[Keeper]	*	Conditional[b]
Scheduling	[Scheduling]	*	Conditional[b]
Capacity	[Capacity]	*	Optional
Vendor Specific	[VendorID_vendorspecifickeyword]	Last	Optional

[a] Placement of these groups only needs to follow the Device Description and Device Classification sections

[b] These sections are required if the corresponding functionality is implemented, else they may be omitted

The ControlNet EDS contents shall be further organised as follows:

— all ControlNet EDS files shall contain the Device Classification section, which shall use the legal delimiter [Device Classification], and may be placed anywhere after the File Description section;

— the optional and conditional sections described in this specification may be present in any order provided that no forward references exist within the EDS file.

A.4.2.2 Implementation of common CIP requirements

A.4.2.2.1 Device Classification section

For any ControlNet compliant device, the Device Classification section of its EDS shall contain at least one ClassN keyword entry with its first field set to ControlNet. Further sub-classification of the ControlNet classification shall be reserved.

A.4.2.2.2 Port section

In the Port section of the EDS, the PortN entry corresponding to a ControlNet compliant port shall be set as follows:

— the “Port Type Name” field shall have a value of “ControlNet”;

— the optional “Port Object” field shall be set to the path of the ControlNet object for this port;

— no additional requirements, beyond those specified in the CIP common subclause (see A.4.1.4.10), are placed on the “Port Name” and “Port Number” fields.

A.4.2.3 Additional data encoding requirements

There are no additional data encoding requirements for ControlNet EDS files.

A.4.2.4 Additional file requirements

A.4.2.4.1 ControlNet Physical Layer section

The ControlNet Physical Layer section shall describe the time delay through a component of the ControlNet Physical Layer such as a repeater or a RG-6 coaxial medium. This section shall only be included in the EDS if the ProdType entry from the Device Description section equals 50. The single entry, "Delay1", shall contain the formatted fields shown in Table A.30.

Table A.30 — ControlNet Physical Layer format

Field name	Field number	Data type	Required/Optional
Units	1	ParamN	Required
Min Delay Per Unit	2	UDINT	Required
Max Delay Per Unit	3	UDINT	Required

The first field, called "Units", shall be a reference to an entry from the [Params] section that describes a USINT parameter. The "Min Delay Per Unit" and "Max Delay Per Unit" fields, when multiplied by the scaled value of the USINT parameter, shall specify the range of possible delay given to each frame passing through the device. The resulting delay shall be in units of picoseconds.

NOTE A link wide propagation time calculation is used to compute the slot time parameter of the moderator packet (see IEC 61158-4:2003 (Type 2)). The USINT parameter is stored in instance attribute 0x0103 of the Keeper object (see IEC 61158-4:2003 (Type 2)).

Figure A.17 is an example that shows a typical EDS for a Physical Layer device.

```
[File]
   DescText  = "RG6 coax EDS file";
   CreateDate= 09-01-1997;
   CreateTime= 17:23:00;
   Revision  = 1.1;

[Device]
   VendCode  = 65535;
   VendName  = "Widget Works, Inc.";
   ProdType  = 50;
   ProdTypeStr  = "ControlNet Physical Layer";
   ProdCode  = 1;
   MajRev    = 1;
   MinRev    = 1;
   ProdName  = "RG6 coax";
   Catalog   = "1234-RG6";

[Params]
   Param1 =
      0, ,,           $ not accessible from link
      0x0004,         $ supports scaling
      8, 1,           $ USINT, 1 byte
      "RG-6 length",  $ name
      "m",            $ units
      "",             $ help string
      1, 255, 10,     $ min, max, default data values
      100, 1, 1, 0,   $ mult, div, base, offset scaling
      , , , , ;       $ extended precision not used

[ControlNet Physical Layer]
   Delay1 = Param1, 4068, 4068;
```

Figure A.17 — Example EDS for a Physical Layer device

A.4.2.4.2 Keeper section

The Keeper section shall describe the Keeper object within a device. The "Revision" entry keyword shall have one UINT field that shall be the revision (class attribute 1) of the Keeper object within the device as shown in Figure A.18.

```
[Keeper]
   Revision = 2;
```

Figure A.18 — [Keeper] section example

A.4.2.4.3 Scheduling section

The Scheduling section shall describe the Scheduling object within a device. The "Revision" entry keyword shall have one UINT field that shall be the revision (class attribute 1) of the Scheduling object within the device as shown in Figure A.19.

```
[Scheduling]
   Revision = 1;
```

Figure A.19 — [Scheduling] section example

A.4.2.4.4 Capacity section

The capacity section shall contain information about the device link layer capacity.

The entry keywords shall be one of the following character arrays, “MaxReceiveLPacketsPerNUT” or MaxTransmitLPacketsPerNUT”. The MaxReceiveLPacketsPerNUT (respectively MaxTransmitLPacketsPerNUT) entry identifies the maximum number of scheduled LPackets this device can receive (respectively transmit) within the specified network update interval.

If MaxReceiveLPacketsPerNUT (respectively MaxTransmitLPacketsPerNUT) is not specified this device supports an unlimited number of receive (respectively transmit) LPackets, at all NUTs.

Each entry shall contain the formatted fields shown in Table A.31.

Table A.31 — MaxReceiveLPacketsPerNUT and MaxTransmitLPacketsPerNUT format

Field name	Field number	Data type	Required/Optional
Reserved	1, 2, 3, 4	Empty	Not used
LPackets	5, 7, 9 …	UINT	Conditional
NUT	6, 8, 10 …	REAL[a]	Conditional
[a]NUT values shall be encoded using the decimal floating point representation, with a fixed number of two decimal digits.			

Fields 1 to 4 shall be reserved and shall be empty.

The remaining fields shall be paired such that a “LPackets” field is paired with a “NUT” field making the total number of fields even. The number of fields pairs in either entry shall be variable.

The entries in the Capacity section shall provided the information as shown in Table A.32.

Table A.32 — MaxReceiveLPacketsPerNUT / MaxTransmitLPacketsPerNUT format

Fields	Description
LPackets	This field specifies the maximum number of LPackets the device can receive (respectively transmit) within the network update time specified by the associated NUT field.
NUT	NUT values shall be between 2.00 and 100.00 milliseconds. The pairs of fields shall be in ascending network update time order. If the last pair of fields does not specify a NUT value of 100.00, this device supports an unlimited number of receive (respectively transmit) LPackets for NUT values greater than the last specified value.

Figure A.20 is an example that shows a typical [Capacity] section.

```
[Capacity]
   MaxReceiveLPacketsPerNUT = ,,,,4,2.51,
                              5,5.21,
                              6,10.23,
                              7,21.43,
                              8,53.29,
                              9,100.00;

   MaxTransmitLPacketsPerNUT= ,,,,40,2.19,
                              51,27.32,
                              75,100.00;
```

Figure A.20 — [Capacity] section example

Annex B
(normative)

PROFIBUS profile templates

B.1 General

Configuration tools currently available for PROFIBUS devices, which comply with IEC 61784-1:2003 CP3/1 and CP3/2 use a specially formatted ASCII file, referred to as Generic Station Description (GSD) and Electronic Device Description (EDD), which provides information about a device for example:

— information needed to identify the connected device,

— description of device data that can be accessed via the network (e.g. configurable parameters),

— description of the communication capabilities supported by the device (e.g. transmission rate),

— additional vendor-specific information.

GSD objects, syntax and sematic are specified in B.4 and B.5. EDD shall be compliant to IEC 61804-2: Ed.1, F.2 (Profile for PROFIBUS).

The GSD and EDD allow a configuration tool to automate the device configuration process. The GSD and EDD requirements provide an open, consistent and compatible approach for performing device configuration.

All devices with a communication interface according IEC 61784-1:2003 CP3/1 and CP3/2 shall have a GSD file. The main intention of GSD is to give information on a PROFIBUS communication network. In Annex B the name PROFIBUS DP and the acronym DP is used for the protocol and services of devices, which are compliant with IEC 61784-1:2003 CP3/1 and CP3/2.

Some devices have in addition an EDD to describe additional device behavior. The main intention of EDD is to give information on device compared to device profile.

The GSD and EDD information is very similar to the information required in both communication network and device profiles, hence the following subclauses specify format for:

— encapsulation of legacy GSD and EDD files in the ISO 15745 templates (“wrappers”),

— the legacy GSD, including common semantics information.

B.2 Device profile template description

B.2.1 General

The device profile XML files used to encapsulate GSD or EDD files shall comply with the device profile XML schema as specified in B.2.2.

For simple devices the GSD contain the device profile information, and the ProfileTechnology element of the device profile wrapper in B.2.2 shall be set to “GSD”. If an EDD exists, then the EDD is the only necessary device profile, and the ProfileTechnology element of the device profile wrapper in B.2.2 shall be set to “EDDL” .

Table B.1 specifies the equivalents from the XML device header to the GSD or EDD file contents.

Table B.1 — ExternalProfileHandle elements

XML schema elements	GSD elements	EDD elements
ProfileIdentification	Ident_Number [a]	MANUFACTURER [b]
ProfileRevision	GSD_Revision [a]	DD_REVISION [b]
[a] See B.4.2 for more details [b] See IEC 61804-2 Ed.1, 9.2 for more details		

B.2.2 XML schema: GSD_Device_Profile_wrapper.xsd or EDD_Device_Profile_wrapper.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.profibus.com/ISO15745/GSDV5"
xmlns="http://www.profibus.com/ISO15745/GSDV5" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <!--* The ISO15745 Profile is the document element of an XML Document-->
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="DeviceFunction" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:any namespace="##any"/>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="ExternalProfileHandle">
               <xsd:annotation>
                  <xsd:documentation>This external profile handle references the non-XML GSD or EDD file.
In the moment, this is the only allowed element in the DeviceProfile.</xsd:documentation>
               </xsd:annotation>
               <xsd:complexType>
                  <xsd:complexContent>
                     <xsd:extension base="ProfileHandle_DataType"/>
                  </xsd:complexContent>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
```

```
   <xsd:annotation>
      <xsd:documentation>* HEADER DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:simpleType name="ProfileClassID_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AIP"/>
         <xsd:enumeration value="Process"/>
         <xsd:enumeration value="InformationExchange"/>
         <xsd:enumeration value="Resource"/>
         <xsd:enumeration value="Device"/>
         <xsd:enumeration value="CommunicationNetwork"/>
         <xsd:enumeration value="Equipment"/>
         <xsd:enumeration value="Human"/>
         <xsd:enumeration value="Material"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="ISO15745Reference_DataType">
      <xsd:sequence>
         <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
         <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
         <xsd:element name="ProfileTechnology">
            <xsd:simpleType>
               <xsd:restriction base="xsd:string">
                  <xsd:enumeration value="GSD"/>
                  <xsd:enumeration value="EDDL"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="IASInterface_DataType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="CSI"/>
               <xsd:enumeration value="HCI"/>
               <xsd:enumeration value="ISI"/>
               <xsd:enumeration value="API"/>
               <xsd:enumeration value="CMI"/>
               <xsd:enumeration value="ESI"/>
               <xsd:enumeration value="FSI"/>
               <xsd:enumeration value="MTI"/>
               <xsd:enumeration value="SEI"/>
               <xsd:enumeration value="USI"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:length value="4"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
   <xsd:annotation>
      <xsd:documentation>* ISO 15745 DEFINED DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType name="ProfileHandle_DataType">
      <xsd:sequence>
         <xsd:element name="ProfileIdentification" type="xsd:string"/>
         <xsd:element name="ProfileRevision" type="xsd:string"/>
         <xsd:element name="ProfileLocation" type="xsd:anyURI" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

B.3 Communication network profile template description

B.3.1 General

The communication profile XML files shall comply with the communication network profile XML schema as specified below. This profile shall reference a communication network profile XML wrapper pointing to a Generic Station Data (GSD) file according B.4 and B.5.

Table B.2 specifies the equivalents from the XML device header to the GSD file contents.

Table B.2 — ExternalProfileHandle elements

XML schema elements	GSD elements
ProfileIdentification	Ident_Number [a]
ProfileRevision	GSD_Revision [a]
[a] See B.4.2 for more details	

B.3.2 XML schema: GSD_CommNet_Profile_wrapper.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.profibus.com/ISO15745/GSDV5"
xmlns="http://www.profibus.com/ISO15745/GSDV5" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <!--* The ISO15745 Profile is the document element of an XML Document-->
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileBody">
      <xsd:annotation>
         <xsd:documentation>For GSD definition, only the communication network profile is
relevant.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ExternalProfileHandle" type="ProfileHandle_DataType">
               <xsd:annotation>
```

```
                  <xsd:documentation>This external profile handle references the non-XML GSD file.  In
the moment, this is the only allowed element in the CommNetworkProfile for GSD.</xsd:documentation>
               </xsd:annotation>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:simpleType name="ProfileClassID_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AIP"/>
         <xsd:enumeration value="Process"/>
         <xsd:enumeration value="InformationExchange"/>
         <xsd:enumeration value="Resource"/>
         <xsd:enumeration value="Device"/>
         <xsd:enumeration value="CommunicationNetwork"/>
         <xsd:enumeration value="Equipment"/>
         <xsd:enumeration value="Human"/>
         <xsd:enumeration value="Material"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="ISO15745Reference_DataType">
      <xsd:sequence>
         <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
         <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
         <xsd:element name="ProfileTechnology" type="xsd:string" fixed="GSD"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="IASInterface_DataType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="CSI"/>
               <xsd:enumeration value="HCI"/>
               <xsd:enumeration value="ISI"/>
               <xsd:enumeration value="API"/>
               <xsd:enumeration value="CMI"/>
               <xsd:enumeration value="ESI"/>
               <xsd:enumeration value="FSI"/>
               <xsd:enumeration value="MTI"/>
               <xsd:enumeration value="SEI"/>
               <xsd:enumeration value="USI"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:length value="4"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
   <xsd:annotation>
      <xsd:documentation>* ISO 15745 DEFINED DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType name="ProfileHandle_DataType">
      <xsd:sequence>
         <xsd:element name="ProfileIdentification" type="xsd:string"/>
         <xsd:element name="ProfileRevision" type="xsd:string"/>
         <xsd:element name="ProfileLocation" type="xsd:anyURI" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```


B.4 Generic Station Description (GSD)

B.4.1 General

PROFIBUS devices may have different behavior and performance characteristics. Features differ in regard to available functionality (i.e., number of I/O signals and diagnostic messages) or possible bus

parameters such as baud rate and time monitoring. These parameters may vary individually for each device type and vendor and are usually documented in the technical manual. In order to achieve a simple Plug and Play configuration for PROFIBUS devices, electronic device data sheets (GSD files) are defined to describe for the communication features of the devices. These are named Generic Station Description (GSD) files, which allow easy configuration of PROFIBUS networks with devices from different manufacturers.

GSD is a human readable ASCII text file. B.5 specifies keywords as mandatory or optional with the corresponding data type and their border values to support the configuration of PROFIBUS devices.

The GSD files characterize the features and performance capabilities of PROFIBUS devices.

Each vendor of a DP-Slave or a DP-Master (class 1) shall offer the characteristic features of the device as a device data sheet and a GSD-File to the user. Using this information enables the user to check all data in the configuration phase of a PROFIBUS system and errors can be avoided as early as possible. Based on the defined file format in clause B.4.2, B.5, and B.6, it is possible to realize vendor independent configuration tools for PROFIBUS systems. The configuration tool uses the GSD-Files for testing the data. These were entered regarding the observance of limits and validity related to the performance of the individual device.

The distinction of the GSD-files is achieved by the vendor- and device-identifiers.

In the case of a device that supports the PROFIBUS DP protocol and another protocol (e.g. PROFIBUS FMS, see [10]), the other specific device data base information shall be located at the beginning of the GSD file.

NOTE ISO 15745-3 only describes the GSD for PROFIBUS DP.

The manufacturer of a device is responsible for the functionality and the quality of its GSD file. The device certification procedure is requesting either a standard GSD file based on a PROFIBUS profile or a device specific GSD file.

GSD fulfill the requirements of a communication network profile.

GSD file format is specified in B.4.2. The GSD Objects, Syntax and Semantic are specified in B.5 and B.6.

B.4.2 Syntax and format of the GSD files

The GSD-File shall be an ASCII-file and it may be created with every applicable ASCII text editor. The DP-specific part shall begin with the identifier "#Profibus_DP". The device database shall be specified as parameter of a keyword. At the evaluation of the keywords the kind of letters, capital or small, are irrelevant.

NOTE 1 The data medium, which the vendor of the DP-device uses for the delivery of the GSD-file, is not defined here.

The file format shall be line oriented. Each line shall contain statements for exactly one parameter. If a semicolon is detected during the interpretation of the line, it is assumed that the rest of the line is a comment. The maximum number of characters per line shall be fixed to 80. If it is not possible to describe the information in one line, then it is allowed to use continuation lines. A "\" at the end of a line indicates that the following line is a continuation line. It is distinguished between number-parameters and text-parameters. No special end-identifier is defined. But it is to be ensured that the file ends after a complete line. Parameters, which are not used for a DP-Master or a DP-Slave, shall be omitted.

NOTE 2 PROFIBUS-Master and PROFIBUS-Slave means devices, which are compliant with IEC 61784-1:2003 CP 3/1 or 3/2, see IEC 61784-1:2003, 7.2.2.1.2ff.

A GSD file should be created and provided to the user in the respective language. At least a default version (GS**D**) in English language is to be created. The language dependent files may only differ in the parameters of the type Visible String and the Slave_Family. The language dependent device description data files differ regarding the last letter of the extension (*.gs?).

Default:	?=d
English:	?=e
French:	?=f
German:	?=g
Italian:	?=i
Portuguese:	?=p
Spanish:	?=s

General specifications

This section in the GSD file shall contain information on vendor and device names, hardware and software release states, baud rates supported, possible time intervals for monitoring times and the signal assignment on the bus connector.

Master-related specifications

This section in the GSD file shall contain all master-related parameters, such as: the maximum number of slaves that can be connected, or upload and download options. This section does not exist for slave devices.

Slave-related specifications

This section in the GSD file shall contain all slave-related specifications, such as the number and type of I/O channels, specification of diagnostic texts and information on the available modules with modular devices. In the individual sections, the parameters are separated by keywords. A distinction is made between mandatory parameters (i.e., Vendor_Name) and optional parameters (i.e., Sync_Mode_supp). The definition of parameter groups allows selection of options. In addition, bit map files with the symbols of the devices can be integrated. The format of the GSD is designed for flexibility. It contains both lists (such as the baud rates supported by the device) as well as space to describe the modules available in a modular device. Plain text can also be assigned to the diagnostic messages. This section does not exist for master devices.

B.5 Semantic of GSD

B.5.1 Conventions

The attributes specified below shall characterize the features and performance of PROFIBUS devices, complying with IEC 61784-1:2003 CP 3/1 or 3/2. A synonym of these PROFIBUS devices in this Annex B is PROFIBUS DP or only the acronym DP.

The type ID specified for the keywords shall refer to the parameters with the same name. In the case of the parameters, a differentiation shall be made between:

- Mandatory (M): absolutely required

- Optional (O): possible in addition
- Default (D): Optional with default = 0 if not present
- Grouped (G): At least one keyword of the group is required

Expansions of the released GSD specifications (for example, new keywords) are provided in this document with a version ID (GSD_Revision) that indicates the version where the expansion was added. Keywords without version ID belong to the original version.

The keywords are classified in:

— General specifications, see B.5.2

— Master-related specifications, see B.5.3

— Slave-related specifications, see B.5.4.

B.5.2 General specifications

B.5.2.1 General DP keywords

GSD_Revision: (M starting with GSD_Revision 1)
Version ID of the GSD file format.
Type: Unsigned8

Vendor_Name: (M)
Manufacturer's Name.
Type: Visible-String (32)

Model_Name: (M)
Manufacturer's designation (Controller Type) of device.
Type: Visible-String (32)

Revision: (M)
Revision version of the device.
Type: Visible-String (32)

Revision_Number: (O starting with GSD_Revision 1)
Version ID of the device. The value of the Revision_Number has to agree with the value of the Revision_Number in the slave-specific diagnosis.
Type: Unsigned8 (1 to 63)

Ident_Number: (M)
Device type of the device.
The Ident_Number is assigned by the PROFIBUS Nutzer Organisation e.V. (PNO) to each device type.
Manufacturers of devices have to apply for the Ident_Number at the PNO.
Type: Unsigned16

Protocol_Ident: (M)
Protocol ID of the device.
Type: Unsigned8
0: PROFIBUS DP,
16 to 255: Manufacturer-specific

Station_Type: (M)
DP device type.
Type: Unsigned8
0: DP Slave,
1: DP Master (Class 1)

FMS_supp: (D)
This device is an FMS/DP mixed device.
Type: Boolean (1: True)

Hardware_Release: (M)
Hardware release of the device.
Type: Visible-String (32)

Software_Release (M)
Software release of the device.
Type: Visible-String (32)

9.6_supp: (G)
The device supports the baudrate 9.6 kbit/s.
Type: Boolean (1: True)

19.2_supp: (G)
The device supports the baudrate 19.2 kbit/s.
Type: Boolean (1: True)

31.25_supp: (G starting with GSD_Revision 2)
The device supports the baudrate 31.25 kbit/s.
Type: Boolean (1: True)

45.45_supp: (G starting with GSD_Revision 2)
The device supports the baudrate 45.45 kbit/s.
Type: Boolean (1: True)

93.75_supp: (G)
The device supports the baudrate 93.75 kbit/s.
Type: Boolean (1: True)

187.5_supp: (G)
The device supports the baudrate 187.5 kbit/s.
Type: Boolean (1: True)

500_supp: (G)
The device supports the baudrate 500 kbit/s.
Type: Boolean (1: True)

1.5M_supp: (G)
The device supports the baudrate 1.5 Mbit/s.
Type: Boolean (1: True)

3M_supp: (G starting with GSD_Revision 1)
The device supports the baudrate 3 Mbit/s.
Type: Boolean (1: True)

6M_supp: (G starting with GSD_Revision 1)
The device supports the baudrate 6 Mbit/s.
Type: Boolean (1: True)

12M_supp: (G starting with GSD_Revision 1)
The device supports the baudrate 12 Mbit/s.
Type: Boolean (1: True)

NOTE In order to secure the optimized performance of the publisher / subscriber functionality it is necessary to set the MaxTsdr_xx values according to the actual values of the device.

MaxTsdr_9.6: (G)
This is the time a responder needs as a maximum at a baudrate of 9.6 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_19.2: (G)
This is the time a responder needs as a maximum at a baudrate of 19.2 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_31.25: (G starting with GSD_Revision 2)
This is the time a responder needs as a maximum at a baudrate of 31.25 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_45.45: (G starting with GSD_Revision 2)
This is the time a responder needs as a maximum at a baudrate of 45.45 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_93.75: (G)
This is the time a responder needs as a maximum at a baudrate of 93.75 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_187.5: (G)
This is the time a responder needs as a maximum at a baudrate of 187.5 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_500: (G)
This is the time a responder needs as a maximum at a baudrate of 500 kbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_1.5M: (G)
This is the time a responder needs as a maximum at a baudrate of 1.5 Mbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_3M: (G starting with GSD_Revision 1)
This is the time a responder needs as a maximum at a baudrate of 3 Mbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_6M: (G starting with GSD_Revision 1)
This is the time a responder needs as a maximum at a baudrate of 6 Mbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

MaxTsdr_12M: (G starting with GSD_Revision 1)
This is the time a responder needs as a maximum at a baudrate of 12 Mbit/s to respond to a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned16
Time base: Bit Time

Redundancy: (D)
This value specifies whether a device supports redundant transmission engineering.
Type: Boolean
0: No, 1: Redundancy is supported.

Repeater_Ctrl_Sig: (D)
Here, the level of the bus connector signal CNTR-P is specified.
Type: Unsigned8
0: Not connected, 1: RS485, 2:TTL

24V_Pins: (D)
Here, the meaning of the bus connector signal M24V and P24V is specified.
Type: Unsigned8
0: Not connected, 1:Input, 2:Output

Implementation_Type: (O starting with GSD_Revision 1)
Here, a description is provided which standard implementation is used in the DP slave, for example Standard Software, Controller or ASIC (Application Specific Integrated Circuit) solution. The manufacturer of the standard solution provides the name; the specification of that name shall be obeyed.
Type: Visible-String (32)

Bitmap_Device: (O starting with GSD_Revision 1)
Here, the file name (*.DIB) of the bit map file is specified in the DIB-Format (70*40 pixels (width*height) 16 colors) that contains the symbolic representation of the device in standard cases.
Type: Visible-String (8)

Bitmap_Diag: (O starting with GSD_Revision 1)
Here, the file name (*.DIB) of the bit map file is specified in the DIB-Format (70*40 pixels (width*height) 16 colors) that contains the symbolic representation of the device for diagnostic cases.
Type: Visible-String (8)

Bitmap_SF: (O starting with GSD_Revision 1)
Here, the file name (*.DIB) of the bit map file is specified in the DIB-Format (70*40 pixels (width*height) 16 colors) that contains the symbolic representation of the device in special operating modes. The meaning is manufacturer-specific.
Type: Visible-String (8)

B.5.2.2 Additional keywords for different physical interfaces

Physical_Interface: (O starting with GSD_Revision 3)
This value specifies the execution of the Physical Layers of PROFIBUS. With this parameter it is possible to have devices with more than one physical interface or interfaces different from RS485. If this keyword is not used, then RS485 standard copper is the only supported physical interface. Between the keywords Physical_Interface and End_Physical_Interface, the Transmission_Delays and the Reaction_Delay of a slave device are specified, for the physical interface used in the device. The Transmission_Delay defines the delay time for the signal which is to be transmitted through the device. The Reaction_Delay defines the delay of signals processed by the device.

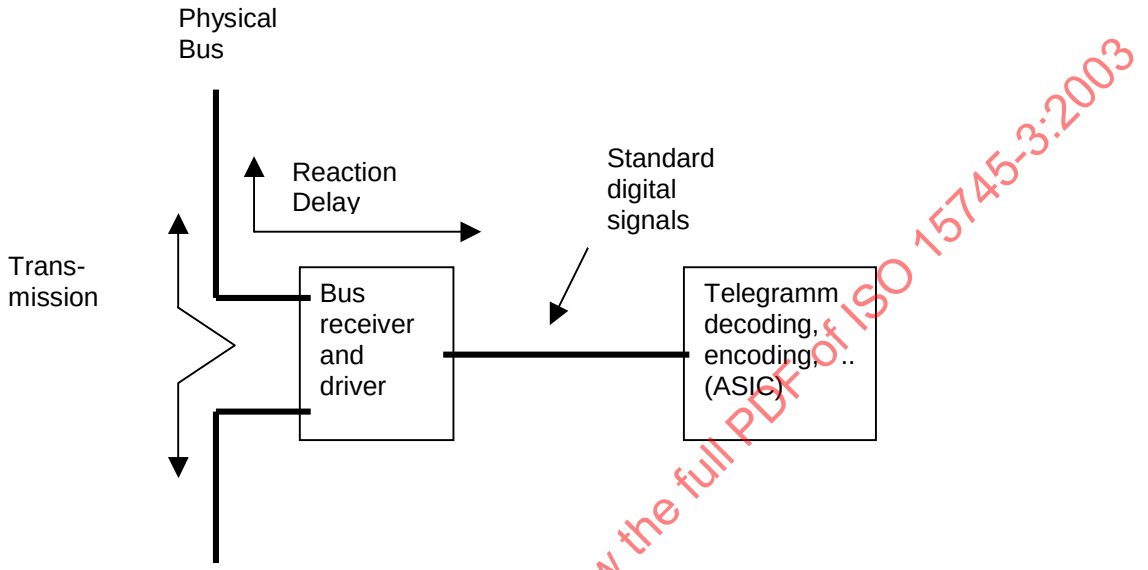

Figure B.1 — Physical_Interface example

EXAMPLE The Transmission_Delay with RS485 is 0, the Reaction_Delay is also 0, because the delay in the driver is lower than 1 bit time, see Figure B.1.

Especially with optical interfaces these parameters are necessary for the bus timing calculation.
Both the Transmission_Delay and the Reaction_Delay has to be defined for each supported baudrate. Otherwise the baudrate is not valid for this physical layer.
Coding of the interfaces:

Type: Unsigned8

0:	RS 485 (ANSI TIA/EIA RS-485-A); optional RS 485-intrinsic safety version (see [11])
1:	Manchester coded and bus powered (MBP); optional IS (MBP-IS) and lower power (MBP-LP)
2:	Plastic fibre
3:	Glass multi mode fibre or Glass single mode fibre
4:	Polymer Clad Fibre (PCF)
5-127:	Reserved
128-255:	Manufacturer specific

Parameters Used:

Transmission_Delay_9.6: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_19.2: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_31.25: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_45.45: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_93.75: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_187.5: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_500: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_1.5M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_3M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_6M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Transmission_Delay_12M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the transmission delay of the device attached to the corresponding physical layer.

Reaction_Delay_9.6: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_19.2: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_31.25: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_45.45: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_93.75: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_187.5: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_500: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_1.5M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_3M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_6M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

Reaction_Delay_12M: (G starting with GSD_Revision 3)
Type: Unsigned16
Time base: Bit Time
This parameter specifies the reaction delay of the device attached to the corresponding physical layer.

B.5.3 Master-related specifications

B.5.3.1 DP Master (Class 1) related keywords

Master_Freeze_Mode_supp: (D starting with GSD_Revision 3)
The device supports the Freeze mode.
Type: Boolean (1: True)

Master_Sync_Mode_supp: (D starting with GSD_Revision 3)
The device supports the Sync mode.
Type: Boolean (1: True)

Master_Fail_Safe_supp: (D starting with GSD_Revision 3)
The device supports the Fail Safe.
Type: Boolean (1: True)

Download_supp: (D)
The device supports the functions Download, Start_seq and End_seq.
Type: Boolean (1: True)

Upload_supp: (D)
The device supports the functions Upload, Start_seq and End_seq.
Type: Boolean (1: True)

Act_Para_Brct_supp: (D)
The device supports the function Act_Para_Brct.
Type: Boolean (1: True)

Act_Param_supp: (D)
The device supports the function Act_Param.
Type: Boolean (1: True)

Max_MPS_Length: (M)
Maximum memory size (in bytes) that a device makes available for storing the master parameter set.
Type: Unsigned32

Max_Lsdu_MS: (M)
Here, the maximum L_sdu length for all master-slave communication relations is specified.
Type: Unsigned8

Max_Lsdu_MM: (M)
Here, the maximum L_sdu length for the master-master communication relations is specified.
Type: Unsigned8

Min_Poll_Timeout: (M)
This value indicates how long a DP master (Class 1) needs as a maximum for processing a master-master function.
Type: Unsigned16
Time base: 10 ms

Trdy_9.6: (G)
This value indicates how fast a DP master (Class 1), at a baudrate of 9.6 kbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_19.2: (G)
This value indicates how fast a DP master (Class 1), at a baudrate of 19.2 kbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_31.25: (G starting with GSD_Revision 2)
This value indicates how fast a DP master (Class 1), at a baudrate of 31.25 kbit/s is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_45.45: (G starting with GSD_Revision 2)
This value indicates how fast a DP master (Class 1), at a baudrate of 45.45 kbit/s is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_93.75: (G)
This value indicates how fast a DP master (Class 1), at a baudrate of 93.75 kbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_187.5: (G)
This value indicates how fast a DP master (Class 1), at a baudrate of 187.5 kbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_500: (G)
This value indicates how fast a DP master (Class 1), at a baudrate of 500 kbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_1.5M: (G)
This value indicates how fast a DP master (Class 1), at a baudrate of 1.5 Mbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_3M: (G starting with GSD_Revision 1)
This value indicates how fast a DP master (Class 1), at a baudrate of 3 Mbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_6M: (G starting with GSD_Revision 1)
This value indicates how fast a DP master (Class 1), at a baudrate of 6 Mbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Trdy_12M: (G starting with GSD_Revision 1)
This value indicates how fast a DP master (Class 1), at a baudrate of 12 Mbit/s, is ready to receive again after sending a request message (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tqui_9.6: (G)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 9.6 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_19.2: (G)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 19.2 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_31.25: (G starting with GSD_Revision 2)
This value specifies the modulator fading time (TQUI), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 31.25 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_45.45: (G starting with GSD_Revision 2)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 45.45 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_93.75: (G)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 93.75 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_187.5: (G)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 187.5 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_500: (G)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 500 kbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_1.5M: (G)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 1.5 Mbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_3M: (G starting with GSD_Revision 1)
This value specifies the modulator fading time (TQUI), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 3 Mbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_6M: (G starting with GSD_Revision 1)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 6 Mbit/s.
Type: Unsigned8
Time base: Bit Time

Tqui_12M: (G starting with GSD_Revision 1)
This value specifies the modulator fading time (T_{QUI}), (refer to IEC 61158-4:2003 Annex E) at a baudrate of 12 Mbit/s.
Type: Unsigned8
Time base: Bit Time

Tset_9.6: (G)
This value specifies the trigger time, at the baudrate of 9.6 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_19.2: (G)
This value specifies the trigger time, at the baudrate of 19.2 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_31.25: (G starting with GSD_Revision 2)
This value specifies the trigger time, at the baudrate of 31.25 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_45.45: (G starting with GSD_Revision 2)
This value specifies the trigger time, at the baudrate of 45.45 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_93.75: (G)
This value specifies the trigger time, at the baudrate of 93.75 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_187.5: (G)
This value specifies the trigger time, at the baudrate of 187.5 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_500: (G)
This value specifies the trigger time, at the baudrate of 500 kbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_1.5M: (G)
This value specifies the trigger time, at the baudrate of 1.5 Mbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_3M: (G starting with GSD_Revision 1)
This value specifies the trigger time, at the baudrate of 3 Mbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_6M: (G starting with GSD_Revision 1)
This value specifies the trigger time, at the baudrate of 6 Mbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

Tset_12M: (G starting with GSD_Revision 1)
This value specifies the trigger time, at the baudrate of 12 Mbit/s, in reference to Layer2 (setup time) from the arrival of an event until the corresponding response (refer to IEC 61158-4:2003 Annex E).
Type: Unsigned8
Time base: Bit Time

LAS_Len: (M)
This value indicates how many entries the device in question can manage in the list of active stations (LAS).
Type: Unsigned8

Tsdi_9.6: (G)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 9.6 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_19.2: (G)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 19.2 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_31.25: (G starting with GSD_Revision 2)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 31.25 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_45.45: (G starting with GSD_Revision 2)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 45.45 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_93.75: (G)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) of the initiator at a baudrate of 93.75 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_187.5: (G)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 187.5 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_500: (G)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 500 kbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_1.5M: (G)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 1.5 Mbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_3M: (G starting with GSD_Revision 1)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 3 Mbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_6M: (G starting with GSD_Revision 1)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 6 Mbit/s.
Type: Unsigned16
Time base: Bit Time

Tsdi_12M: (G starting with GSD_Revision 1)
This value specifies the station delay time (Tsdi) of the initiator (refer to IEC 61158-4:2003 Annex E) at a baudrate of 12 Mbit/s.
Type: Unsigned16
Time base: Bit Time

Max_Slaves_supp: (M)
This value indicates how many DP slave stations a DP master (Class 1) can handle.
Type: Unsigned8

Max_Master_Input_Len: (O starting with GSD_Revision 1)
Here, the maximum length of the input data per DP slave is specified that the DP master supports.
Type: Unsigned8

Max_Master_Output_Len: (O starting with GSD_Revision 1)
Here, the maximum length of the output data per DP slave is specified that the DP master supports.

Type: Unsigned8

Max_Master_Data_Len: (O starting with GSD_Revision 1)
Here, the sum of the lengths of the output and input data per DP slave is specified that the DP master supports. If this keyword is not provided, the maximum length will be the sum of the input and output data.
Type: Unsigned16

B.5.3.2 Additional master related keywords for DP extensions

DPV1_Master: (D starting with GSD_Revision 3)
The DP master supports DP-V1-extensions of the DP protocol.
Type: Boolean (1: True)

DPV1_Conformance_Class: (M if DPV1_Master, starting with GSD_Revision 3)
This value specifies the Conformance Class of the DP-Master (Class1). The following Conformance Classes are specified for DP-Master (Class 1):
Type: Unsigned8

1:	Conformance Class A
2:	Conformance Class B
0,3-255:	reserved

C1_Master_Read_Write_supp: (D starting with GSD_Revision 3)
The DP-Master (Class 1) supports the Read and Write services on the C1-communication relationship.
Type: Boolean (1: True)

Master_DPV1_Alarm_supp: (D starting with GSD_Revision 3)
The DP-Master (Class 1) supports alarms.
Type: Boolean (1: True)

Master_Diagnostic_Alarm_supp: (G if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The device supports Diagnostic_Alarm. A diagnostic alarm signals an event within a slot, for instance over temperature, short circuit, etc..
Type: Boolean (1: True)

Master_Process_Alarm_supp: (G if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The device supports Process_Alarm.A process alarm signals the occurrence of an event in the connected process, for instance upper limit value exceeded.
Type: Boolean (1: True)

Master_Pull_Plug_Alarm_supp: (G if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The device supports Pull_Alarm. A pull alarm signals the withdrawal of a module at a slot.
Type: Boolean (1: True)

Master_Status_Alarm_supp: (G if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The device supports Status_Alarm. A status alarm signals a change in the state of a module, for instance run, stop or ready.
Type: Boolean (1: True)

Master_Update_Alarm_supp: (G if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The device supports Update_Alarm. An update alarm signals the change of a parameter in a slot e.g. by a local operation or remote access.
Type: Boolean (1: True)

Master_Manufacturer_Specific_Alarm_supp: (G if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The device supports Manufacturer_Specific_Alarm. A manufacturer specific alarm signals an event defined by the manufacturer.
Type: Boolean (1: True)

Master_Extra_Alarm_SAP_supp: (D if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
In addition to SAP51 it is possible to handle the MSAL_Alarm_Ack via SAP 50 if the Bit Sl_Flag.Extra_Alarm_SAP in the corresponding slave parameter set is set. In this case there may be a higher performance because SAP 50 is used exclusively for the MSAL_Alarm_Ack service and the service cannot be delayed by a running MSAC1_Write or MSAC1_Read service.
Type: Boolean (1: True)

Master_Alarm_Sequence_Mode: (M if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The DP master supports the Alarm Sequence Mode with the specified number of alarms for alarm handling. The Sequence Mode is an option of parallel alarm handling. Several alarms (2 to 32) of the same or different type can be active at one time (fixed by the DDLM_Set_Prm service).

Type: Unsigned8

0 = Sequence_Mode not supported
1 = 2 alarms in total
2 = 4 alarms in total
3 = 8 alarms in total
4 = 12 alarms in total
5 = 16 alarms in total
6 = 24 alarms in total
7 = 32 alarms in total

Master_Alarm_Type_Mode_supp: (M if Master_DPV1_Alarm_supp, starting with GSD_Revision 3)
The DP master supports the Alarm Type Mode. The Type Mode is mandatory if the DP-Master supports parallel alarm handling. One alarm of each type can be active at one time (fixed by the DDLM_Set_Prm service).

Type: Boolean (shall always be set to 1: True)

B.5.3.3 Additional master related keywords for DP-V2

Isochron_Mode_Synchronised (D starting with GSD_Revision 4):
This parameter indicates whether a master device has the capability to run in the Isochron_Mode and which model it does support. Therefore, the following 4 values are allowed:

Type: unsigned8

0: Master device does not support the Isochron_Mode
1: Master device supports only the buffer synchronized Isochron_Mode
2: Master device supports only the enhanced synchronized Isochron_Mode
3: Master device supports both the buffer synchronized and the enhanced synchronized Isochron_Mode.

NOTE For further information about the functionality of the Isochron_Mode see [15].

DXB_Master_supp: (D starting with GSD_Revision 4)
The DP-Master supports the service of Data Exchange with Broadcast.

Type: Boolean (1: True)

X_Master_Prm_SAP_supp: (D starting with GSD_Revision 4)
Indicates, if the X_Prm_SAP of the slave can be addressed by the master. Shall be true, if DPV1_Master = 1 and if the master supports structured `parameterization data.`

Type: Boolean (1: True)

B.5.4 Slave-related specifications

B.5.4.1 Basic DP-Slave related keywords

Freeze_Mode_supp: (D)
The DP device supports the Freeze mode. DP slaves that support the Freeze mode have to guarantee that in the next data cycle after the Freeze control command, the values of the inputs that were frozen last are transferred to the bus.

Type: Boolean (1: True)

Sync_Mode_supp: (D)
The DP device supports the Sync mode.

Type: Boolean (1: True)

Auto_Baud_supp: (D)
The DP device supports automatic baudrate recognition.
Type: Boolean (1: True)

Set_Slave_Add_supp: (D)
The DP device supports the function Set_Slave_Add.
Type: Boolean (1: True)

User_Prm_Data_Len: (D)
Here, the length of User_Prm_Data is specified. The amount of data of User_Prm_Data has to agree with this parameter.
Type: Unsigned8

User_Prm_Data: (O)
Manufacturer-specific field. Specifies the default value for User_Prm_Data. If this parameter is used, its length has to agree with the User_Prm_Data_Len.
Type: Octet-String

Min_Slave_Intervall: (M)
This time specifies the minimum interval between two slave list cycles for the DP device.
Type: Unsigned16
Time base: 100 µs

Modular_Station: (D)
Here it is specified whether the DP device is a modular station. It's strongly recommended to model slaves in the following way: A compact device has only one module with all configuration identifiers. A modular device has only one configuration identifier in each module definition. When a slave accepts only one configuration identifier selected from a number of possible configurations, then the slave should be a modular station with Max_Module =1.
Type: Boolean
0: compact device
1: modular device

Max_Module: (M if Modular_Station)
Here, the maximum number of modules of a modular station is specified.
Type: Unsigned8

Max_Input_Len: (M if Modular_Station)
Here, the maximum length of the input data of a modular station is specified in bytes.
Type: Unsigned8

Max_Output_Len: (M if Modular_Station)
Here, the maximum length of the output data of a modular station is specified in bytes.
Type: Unsigned8

Max_Data_Len: (O only if Modular_Station)
Here, the largest sum of the lengths of the output and input data of a modular station is specified in bytes. Max_Data_Len shall be in minimum the highest value of Max_Input_Len and Max_Output_Len, in maximum the sum of both. If this keyword is not provided, the maximum length is the sum of all input and output data.
Type: Unsigned16

EXAMPLE 1
Max_Input_Len = 24
Max_Output_Len = 30
Max_Data_Len = 30 (minimum)

EXAMPLE 2
Max_Input_Len = 120
Max_Output_Len = 120
Max_Data_Len = 200

EXAMPLE 3
Max_Input_Len = 240
Max_Output_Len = 240
Max_Data_Len = 480 (maximum)

(X_)Unit_Diag_Bit: (O, X_ starting with GSD_Revision 4)
In order to display manufacturer-specific status- and error messages of a DP slave centrally, it is possible to assign to a bit a text (Diag_Text) in the device-related diagnostic field if the bit value equals 1.
Parameters used:
Bit
Type: Unsigned16
Meaning: Bit position in device-related diagnostic field (LSB in first byte is Bit 0).
Diag_Text:
Type: Visible-String (32)

(X_)Unit_Diag_Bit_Help: (O starting with GSD_Revision 5)
Here additional information about the manufacturer-specific status- and error messages is defined. The configuration tool can offer this information to the user additional to the Diag_Text of the (X_)Unit_Diag_Bit corresponding bit position.
Parameters used:
Bit
Type: Unsigned16
Meaning: Bit position in device-related diagnostic field(LSB in first byte is Bit 0).
Help_Text:
Type: Visible-String (256)

(X_)Unit_Diag_Not_Bit: (O starting with GSD_Revision 4)
In order to display manufacturer-specific status- and error messages of a DP slave centrally, it is possible to assign to a bit a text (Diag_Text) in the device-related diagnostic field if the bit value equals 0.
Parameters used:
Bit
Type: Unsigned16
Meaning: Bit position in device-related diagnostic field(LSB in first byte is Bit 0).
Diag_Text:
Type: Visible-String (32)

(X_)Unit_Diag_Not_Bit_Help: (O starting with GSD_Revision 5)
Here additional information about the manufacturer-specific status- and error messages is defined. The configuration tool can offer this information to the user additional to the Diag_Text of the (X_)Unit_Diag_Not_Bit corresponding bit position.
Parameters used:
Bit
Type: Unsigned16
Meaning: Bit position in device-related diagnostic field(LSB in first byte is Bit 0).
Help_Text:
Type: Visible-String (256)

(X_)Unit_Diag_Area: (O, X_ starting with GSD_Revision 4)
Between the keywords (X_)Unit_Diag_Area and (X_)Unit_Diag_Area_End, the assignment of values in a bit field in the device-related diagnostic field to texts (Diag_Text) is specified.
Parameters used:

First_Bit:
Type: Unsigned16
Meaning: First bit position of the bit field(LSB in the first byte is Bit 0)

Last_Bit:
Type: Unsigned16
Meaning: Last bit position of the bit field. The bit field may be 16 bits wide maximum.

(X_)Value:, (X_ starting with GSD_Revision 4)
Type: Unsigned16
Meaning: Value in the bit field

Diag_Text:
Type: Visible-String (32)

(X_)Value_Help: (O starting with GSD_Revision 5)
Type: Unsigned16
Meaning: Value in the bit field

Help_Text:
Type: Visible-String (256)

UnitDiagType: (O starting with GSD_Revision 4)
Between the keywords UnitDiagType and EndUnitDiagType, different structures within the Unit-Diag can be described. This is meaningful especially for DP-V1 slaves. Only the keywords starting with “X_“ are allowed. The counting starts with Octet 2, the first bit of the type, see Figure B.2. The first bit to be defined is Bit 24, the first bit of the Diagnosis_User_Data in octet 5, see also Figure B.3. Description of Diagnosis_User_Data see IEC 61158-6:2003 Table 396, in row Device_Related_Diagnosis).

Octet	Name	7	6	5	4	3	2	1	0
1	Header_Byte								
2	Type	7	6	5	4	3	2	1	0
3	Slot	15	14	13	12	11	10	9	8
4	Specifier	23	22	21	20	19	18	17	16
5	Diagnosis_User_Data	31	30	29	28	27	26	25	24
6		39	38	37	36	35	34	33	32
.							...		
:									

Figure B.2 — Counting of UnitDiagType

Parameters used:

Diag_Type_Number:
Type: Unsigned8
Meaning: Defines, if an alarm block (0-127) or a status block (128-255) is described.

EXAMPLE 4

UnitDiagType	= 1
X_Unit_Diag_Bit(7)	= “Error at location A2“
X_Unit_Diag_Bit_Help(7)	= “Please correct ....“
X_Unit_Diag_Not_Bit(7)	= “No error at location A2“
X_Unit_Diag_Not_Bit_Help(7)	= “No action ....“
UnitDiagType	= 161
X_Unit_Diag_Bit(40)	= “TDP_Verletzung”
X_Unit_Diag_Bit(41)	= “TDX_Verletzung”
X_Unit_Diag_Bit(42)	= “TSYNC_Prm_Fault”
X_Unit_Diag_Area	= 57-63
X_Value(1)	= “Error 1“
X_Value_Help(1)	= “Please correct ....“
X_Value(10)	= “Error 10“
X_Value_Help(10)	= “Please correct ....“
X_Unit_Diag_Area_End	
EndUnitDiagType	

Figure B.3 illustrates the coding of a diagnosis type Alarm, which can be described by a UnitDiagType.

Octet	Name	7	6	5	4	3	2	1	0
1	Header_ Byte	0	0	Block_Length (4...63)					
2	Type	0	Alarm-Type						
3	Slot	Slot_Number (0...244)							
4	Specifier	Sequence_Number					Add_ Ack	Alarm_ Specifier	
5 to length		Diagnosis_User_Data (0...59 Byte)							

Figure B.3 — coding of a diagnosis type alarm

The following Alarm types are defined:

0	reserved
1	Diagnostic_Alarm
2	Process_Alarm
3	Pull_Alarm
4	Plug_Alarm
5	Status_Alarm
6	Update_Alarm
7 – 31	reserved
32 – 126	manufacturer-specific
127	reserved

Figure B.4 illustrates the coding of a diagnosis type status, which can be described by a UnitDiagType too.

Octet	Name	7	6	5	4	3	2	1	0
1	Header_ Byte	0	0	Block_Length (4…63)					
2	Type	1	Status_Type						
3	Slot	Slot_Number (0…244)							
4	Specifier	reserved						Status_ Specifier	
5 to length		Diagnosis_User_Data (0…59 Byte)							

Figure B.4 — coding of a diagnosis type status

The following Status types are defined:

0: reserved
1: Status_Message
2: Module_Status
3: DXB_Link_Status
4-29: reserved
30: PrmCmdAck
31: Red_State
32-126: manufacturer-specific
127: reserved

Module: (M)
Between the keywords Module and EndModule, the IDs of a DP compact device or the IDs of all possible modules of a modular slave are specified, manufacturer-specific error types are specified in the channel-related diagnostic field, and the User_Prm_Data is described. If, in the case of modular slaves, empty slots are to be defined as empty module (ID/s 0x00), the empty module has to be defined. Otherwise, empty slots would not appear in the configuration data.
If the keyword Channel_Diag is used outside the keywords Module and EndModule, the same manufacturer-specific error type is specified in the channel-related diagnostic field for all modules. Channel_Diag definitions for a manufacturer specific error type inside a module will overwrite the definition for this error type defined for the device. Channel_Diag inside a module do not influence other modules. If the keywords Ext_User_Prm_Data_Ref or Ext_User_Prm_Data_Const (X_Ext_User_Prm_Data_Ref or X_Ext_User_Prm_Data_Const) are used outside the keywords Module and EndModule, the associated User_Prm_Data area refers to the entire device, and the data in the parameter offset to the entire User_Prm_Data. This User_Prm_Data area has to be at the start of the User_Prm_Data. The module-specific User_Prm_Data is directly attached to the device-specific User_Prm_Data in the sequence in which the associated modules were configured. If the keywords Ext_User_Prm_Data_Ref or Ext_User_Prm_Data_Const (X_Ext_User_Prm_Data_Ref or X_Ext_User_Prm_Data_Const / F_Ext_User_Prm_Data_Ref or F_Ext_User_Prm_Data_Const) are used within the keywords Module and EndModule, the data in the parameter offset refers only to the start of the User_Prm_Data area that is assigned to this module.
Parameters used:
Mod_Name:
Type: Visible-String (32)
Meaning: Module name of a module used in a modular DP station, or device name of a compact DP slave.

Config:
Type: Octet-String (17)
Type: Octet-String (244) (O starting with GSD_Revision 1)
Meaning: Here, the ID or IDs of the module of a modular DP slave or of a compact DP device are specified.

Module_Reference: (O starting with GSD_Revision 1,
M starting with GSD_Revision 3)
Type: Unsigned16
Meaning: Here, the reference of the module description is specified. This reference shall be unique for a device (same Ident_Number).This referencing is useful in order to make language-independent configuring possible in a language-dependent system, or to recognize modules.

Ext_Module_Prm_Data_Len: (O starting with GSD_Revision 1)
Type: Unsigned8
Meaning: Here, the length of the associated User_Prm_Data is defined.

X_Ext_Module_Prm_Data_Len: (O starting with GSD_Revision 4)
Type: Unsigned8 (1 – 244)
Meaning: Here, the length of the associated User_Prm_Data for the X_Prm_SAP is defined.

F_Ext_Module_Prm_Data_Len: (O starting with GSD_Revision 4)
Type: Unsigned8 (1 – 237)
Meaning: Here, the length of the ExtUserPrmData for the F- module is defined.

Data_Area: (O starting with GSD_Revision 5)
Meaning: Between the keywords Data_Area_Beg and Data_Area_End, the output area of the module is specified. The description always begins with the first area (`slot`) and rise without gaps.

Area_Name: (O starting with GSD_Revision 5)
Type: Visible-String (32)
Meaning: Name of the area that is described.
This parameter is mandatory between a Data_Area.

Related_CFG_Identifier: (O starting with GSD_Revision 5)
Type: unsigned 8
Meaning: Index of the CFG ID byte, begins with 1, even if only one CFG-Identifier exists.
This parameter is mandatory between a Data_Area.

Length: (O starting with GSD_Revision 5)
Type: unsigned 8 (1 – 244)
Meaning: Length of the Data_Area in bytes.
This parameter is mandatory between a Data_Area

Consistency: (O starting with GSD_Revision 5)
Type: unsigned 8
0: Consistency only for the given Data_Types of the Data_Area
1: Consistency of the whole Data_Area
Meaning: Demanded is either the consistency of the given Data_Type or of the whole Data_Area. The CFG ID has to have the same level of the consistency or one level higher.
This parameter is mandatory between a Data_Area.

Publisher_allowed: (O starting with GSD_Revision 5)
Type: Boolean (1: True)
Meaning: Publisher is allowed. This parameter is mandatory between a Data_Area.

DP_Master_allowed: (O starting with GSD_Revision 5)
Type: Boolean (1: True)
Meaning: This parameter is mandatory between a Data_Area.

Data_Type: (M inside Data_Area, starting with GSD_Revision 5)
Type: Unsigned 8
Meaning: Specifies the Data_Type. This value complies with the standard data type specification in IEC 61158-6. One or more data types are possible, i.e. U8 or Float (Idx. 5,8) at PA. This parameter is mandatory between a Data_Area.

EXAMPLE 5 (Drive)

```
Module = "PPO 1: 4 PKW | 2 PZD" 0xF3, 0xF1
; First Data_Area
Data_Area_Beg
Area_Name            = "4 PKW"
Related_CFG_Identifier = 1
Length               = 1
Consistency          = 0
Publisher_allowed    = 0
DP_Master_allowed    = 0
Data_Type            = 6  ;Unsigned16
Data_Area_End
; Second Data_Area
Data_Area_Beg
Area_Name            = "2 PZD"
Related_CFG_Identifier = 2
Length               = 1
Consistency          = 1
Publisher_allowed    = 0
DP_Master_allowed    = 0
Data_Type            = 6  ;Unsigned16
Data_Area_End
; End Data_Area
EndModule
```

Example 6 (PROFIBUS PA device)

```
Module = "SP,READBACK,POS_D" 0xA4,0x96
; First Data_Area
Data_Area_Beg
Area_Name            = "Outputs"
Related_CFG_Identifier = 1
Length               = 5
Consistency          = 1
Publisher_allowed    = 0
DP_Master_allowed    = 0
Data_Type            = 5  ;Unsigned8
Data_Area_End
; Second Data_Area
Data_Area_Beg
Area_Name            = "Inputs"
Related_CFG_Identifier = 2
Length               = 7
Consistency          = 1
Publisher_allowed    = 0
DP_Master_allowed    = 0
Data_Type            = 8  ;Floating Point
Data_Area_End
; End Data_Area
EndModule
```

Channel_Diag: (O)
With the keyword Channel_Diag, the assignment of manufacturer-specific error types (Error_Type)in the channel-related diagnostic field to texts (Diag_Text) is specified.
Parameters Used:
Error_Type:
Type: Unsigned8 (16 <= Error_Type <= 31)

Diag_Text:
Type: Visible-String(32)

Channel_Diag_Help: (O starting with GSD_Revision 5)
Here additional information about channel-related diagnostic is defined. The configuration tool can offer this information to the user additional to the Diag_Text of the Channel_Diag corresponding error type.
Parameters used:
Error_Type:
Type: unsigned8 (16 <= Error_Type <= 31)
Help_Text:
Type: Visible-String (256)

Fail_Safe: (D starting with GSD_Revision 1)
Here it is specified whether the DP slave accepts a data message without data instead of a data message with data = 0 in the CLEAR mode of the DP master (Class 1).
Type: Boolean (1: True)

Max_Diag_Data_Len: (M starting with GSD_Revision 1)
Here, the maximum length of the diagnostic information (Diag_Data) is specified.
Type: Unsigned8 (6 - 244)

Modul_Offset: (D starting with GSD_Revision 1)
Here, the slot number is specified that is to appear in the configuration tool as the first slot number at configuring (is used for improved representation).
Type: Unsigned8

Slave_Family: (M starting with GSD_Revision 1)
Here, the DP slave is assigned to a function class. The family name is structured hierarchically. In addition to the main family, subfamilies can be generated that are respectively added with "@". A maximum of three subfamilies can be defined.

EXAMPLE 7 Slave_Family=3@Digital@24V

Type: Unsigned8
The following main families are specified:
0: General (can't be assigned to the categories below)
1: Drives
2: Switching devices
3: I/O
4: Valves
5: Controllers
6: HMI
7: Encoders
8: NC/RC
9: Gateway
10: Programmable Logic Controllers (PLC)
11: Ident systems
12: PROFIBUS PA Profile (independent of used Physical Layer)
13-255: reserved

Diag_Update_Delay: (D starting with GSD_Revision 3)
The parameter is used to count the number of DDLM_Slave_Diag.con while Diag_Data.Prm_Req is still set (for slaves with reduced performance). The value of the Diag_Update_Delay is related to the Min_Slave_Intervall of the Slave.
Type: Unsigned8
Delay = Diag_Upd_Delay * Min_Slave_Intervall

Fail_Safe_required: (D starting with GSD_Revision 3)
This keyword corresponds to the keyword "Fail_Safe" of GSD_Revision 1.
The information is mapped to the Bit "Fail_Safe" in the DPV1_Status_1 of the DDLM_Set_Prm service.
The combination Fail_Safe = 0 and Fail_Safe_required =1 for the device or any module is not possible.
Type: Boolean (1: True)
True: The device or a module requires the Fail_Safe mode for secure operation and is not optional.
False: The use of the Fail_Safe mode is optional.

Info_Text: (O starting with GSD_Revision 3)
Here additional information about the device or the module can be described. The configuration tool can offer this information to the user additional to the visible string of the Model_Name or Module.
Type: Visible String(256)

Max_User_Prm_Data_Len: (O starting with GSD_Revision 1; M starting with GSD_Revision 5)
Here, the maximum length of the user parameterization data is specified.
The definition of this keyword excludes the evaluation of User_Prm_Data_Len and User_Prm_Data.
Type: Unsigned8 (0 - 237)

Ext_User_Prm_Data_Ref: (O starting with GSD_Revision 1)
Here, a reference to a user parameterization data description is specified. The definition of this key word excludes the evaluation of User_Prm_Data and User_Prm_Data_Len. If areas overlap when describing the parameterization data, the area defined last in the GSD file has priority.
Parameters used:
Reference_Offset:
Type: Unsigned8
Meaning: Here, the offset within the associated part of the User_Prm_Data is defined.

Reference_Number:
Type: Unsigned16
Meaning: This reference number has to be the same as the reference number that is defined in the User_Prm_Data description.

Ext_User_Prm_Data_Const: (O starting with GSD_Revision 1)
Here, a constant part of the user parameterization data is specified. The definition of this keyword excludes the evaluation of User_Prm_Data and User_Prm_Data_Len. If areas overlap when describing the parameterization data, the area defined last in the GSD file has priority.
Parameters used:
Const_Offset:
Type: Unsigned8
Meaning: Here, the offset within the associated part of parameterization data is defined.

Const_Prm_Data:
Type: Octet-String
Meaning: Here, the constants or default selections within the parameterization data are defined.

ExtUserPrmData: (O starting with GSD_Revision 1)
Between the keywords ExtUserPrmData and EndExtUserPrmData, a parameter of the user parameterization data is described. The definition of this keyword excludes the evaluation of User_Prm_Data.
Parameters used:

Reference_Number:
Type: Unsigned16
Meaning: Here, the reference of the parameterization data description is specified. This reference has to be unique.

Ext_User_Prm_Data_Name:
Type: Visible-String (32) `or "[SlotNumber]"`
Meaning: Clear text description of the parameters. Here, the slot number can be entered automatically.
[SlotNumber]: (O starting with GSD_Revision 5)
If the Visible-String of the Ext_User_Prm_Data_Name is "[SlotNumber]", the real slot number will be entered automatically by the configuration tool.

EXAMPLE 8

```
ExtUserPrmData = 17 "[SlotNumber]"
Unsigned8 1 1-11
EndExtUserPrmData
```

Data_Type_Name:
Type: Visible-String (32)
Meaning: Default value of the described parameter.

Default_Value:
Type: DataType (has to correspond to the Data_Type_Name)
Meaning: Default value of the described parameter.

Min_Value:
Type: Data_Type (has to correspond to the Data_Type_Name)
Meaning: Minimum value of the described parameter.

Max_Value:
Type: Data_Type (has to correspond to the Data_Type_Name)
Meaning: Maximum value of the described parameter.

Allowed_Values:
Type: Data_Type_Array (16) (has to correspond to the Data_Type_Name)
Meaning: Permitted values of the described parameter.

Prm_Text_Ref:
Type: Unsigned16
Meaning: This reference number has to be the same as the reference number that is defined in the PrmText description.

Changeable: (O starting with GSD_Revision 4)
Type: Boolean (1:True, default = 1 if not present)
Meaning: Indicates whether this user parameter shall be changeable in the user dialog.

Visible: (O starting with GSD_Revision 4)
Type: Boolean (1:True, default = 1 if not present)
Meaning: Indicates whether this user parameter shall be visible in the user dialog.

PrmText:
Between the keywords PrmText and EndPrmText, possible values of a parameter are described. Texts are also assigned to these values.
Parameters Used:
Reference_Number:
Type: Unsigned16
Meaning: Here, the reference of the PrmText description is specified. This reference must be unique.

Text_Item:
Parameter Used:
Prm_Data_Type:
Type: Data_Type (has to correspond to the Data_Type_Name in the parameter description)
Meaning: Here, the value of the parameter is specified that is to be described.
Text:
Type: Visible-String (32)
Meaning: Description of the parameter value.

Prm_Block_Structure_supp: (O starting with GSD_Revision 4)
Here, the slave indicates that the block structure of the extended parameterization is supported within the user parameterization data. If Prm_Block_Structure_supp = 1, the parameterization data shall be structured. The bit Prm_Structure (DPV1_Status_3) will be set by the configuration tool.
If Prm_Block_Structure_supp = 0, the parameterization data shall not be structured, but can show the form of a Block-Structure. The bit Prm_Structure will not be set by the configuration tool.
- The Prm_Structure is necessary for following blocks:
  PrmCmd(Structure_Type=2), DXB-Linktable(3), IsoM-Parameter(4), DXB-Subscribertable(7), Time AR Parameter(8), Manufacturer specific blocks(32 .. $128_{Decimal}$)
- Following blocks shall not be (pre-)defined within the GSD file: PrmCmd(Structure_Type=2), DXB-Linktable(3), IsoM-Parameter(4), DXB-Subscribertable(7), Time AR Parameter(8).
  For these blocks the configuration tool will insert the corresponding Prm_Block automatically to the parameterization telegram regarding to the keywords and the tool settings after the fixed blocks. The first fixed block contains the 3 DPV1-Status-Bytes.
- The F_Parameter-Block(5) is a fixed block and shall be described by the slave related keywords for PROFIsafe Profile.
- The User_Prm_Data($129_{Decimal}$) and the Manufacturer specific blocks(32 .. $128_{Decimal}$) shall be described by the (X_)Ext_User_Prm_Dat_Ref or by the (X_)Ext_User_Prm_Dat_Const. These blocks shall be fixed defined within the GSD file. Fixed blocks will be inserted always at begin of the parameterization data.

Shall be true, if DPV1_Slave = 1.
Type: Boolean (1: True)

Prm_Block_Structure_req: (O starting with GSD_Revision 4)
This parameter indicates whether the slave does require the master to support the Prm_Block_Structure.
Type: Boolean (1: True)
True: The device cannot be operated by a master that does not support the Prm_Block_Structure
False: The use of the Prm_Block_Structure is optional.

Jokerblock_supp: (O starting with GSD_Revision 5)
Indicates, if the DP-Slave supports a Jokerblock according to the block structure of the extended parameterization within the UserPrmData. Following rules have to be observed:
- the Jokerblock shall be used at the end of the parameterization telegram (after fix defined blocks as well as after blocks who will be inserted by the configuration tool);
- the length of the Jokerblock is defined with “255”;
- the Jokerblock shall not be used for PrmCmd, DXB-Linktable, IsoM_Parameter, DXB-Subscribertable, Time AR parameter; F_Parameter;
- the Jokerblock can be send to every slot;
- the Jokerblock can be used also at the X_Prm_SAP as last block of the extended parameterization telegram
Type: Boolean (1: True)

Jokerblock_Type: (O starting with GSD_Revision 5)
Between the parameter Jokerblock_Type and End_Jokerblock_Type, each single block of the parameterization data of the Jokerblock will be described.
This parameter indicates what Structure_Type is described within the block. Mandatory if Jokerblock_supp = 1.
Type: unsigned 8
0.. 31: Reserved
32 .. 128: Manufacturer specific Data
129: User_Prm_Data
130 .. 255: Reserved

Jokerblock_Slot: (O starting with GSD_Revision 5)
This parameter indicates the referenced Slot_Number. Mandatory if Jokerblock_supp = 1.
Type: unsigned 8

Jokerblock_Location: (D starting with GSD_Revision 5)
This parameter indicates what type of SAP is supported from the Jokerblock.
Type: unsigned 8
0: Prm-Telegram
1: Prm-Telegram or Ext-Prm-Telegram; only allowed, if X_Prm_SAP_supp = 1.
2: Ext-Prm-Telegram; only allowed, if X_Prm_SAP_supp = 1.

PrmCmd_supp: (O starting with GSD_Revision 5)
Indicates, if the DP-Slave supports PrmCmd.
Type: Boolean (1: True)

Max_Switch_Over_Time: (O starting with GSD_Revision 5)
Time needed within DP-Slave from PrmCmd receipt until the update of diagnosis with the calculated Red_State.
Type: Unsigned 16
Time base: 100 ms

Slave_Redundancy_supp: (O starting with GSD_Revision 5)
Indicates, if the DP-Slave supports slave redundancy according [11].
Type: unsigned 8
0: not supported
1 .. 7: Reserved
8: Slave Redundancy Version 1.0
9 .. 255: Reserved

Ident_Maintenance_supp: (O starting with GSD_Revision 5)
The device or module supports identification and maintenance functions according [16].
Type: Boolean (1: True)

Time_Sync_supp: (O starting with GSD_Revision 5)
The device supports clock synchronization according to IEC 61784-1:2003, 7.2.3.2.5.10, that references to IEC 61158-5:2003, 8.2.9 Time ASE and from there to IEC 61158-3:2003, 14.4.5 and others.
Type: Boolean (1: True)

B.5.4.2 Additional keywords for module assignment

SlotDefinition: (O only if Modular_Station, starting with GSD_Revision 3)
Between the keywords SlotDefinition and EndSlotDefinition, the possibilities of using the modules within the slots is described. The modules are referenced by the Module_Reference. The names of the slots are mandatory. The default module will be integrated automatically in the configuration (-telegram). This module can be replaced with one of the permitted modules from the list.
The modules can be encountered using permitted values (8,9,13,...) or using a complete range (17-22).

Slot: (O starting with GSD_Revision 3)
Meaning: This parameter specifies the modules that can be used in the specified slot.

Slot_Number:
Type: Unsigned8
Meaning: Here the number of the slot within the device is specified. The number of the slot must be starting with 1 and arise without gaps. If the SlotDefinition is used, then it's highly recommended, that the Modul_Offset is also equal 1. Not every slot of a device must be described by this slot definition. Additional modules may appear behind the highest defined Slot_Number.

Slot_Name:
Type: Visible-String (32)
Meaning: Text description of the slot (This means the application function name).

Default_Value:
Type: Unsigned16
Meaning: Default value, Module_Reference of the module used in this slot.

Min_Value:
Type: Unsigned16
Meaning: Minimum value, lowest Module_Reference of the modules that can be used in this slot.

Max_Value:
Type: Unsigned16
Meaning: Maximum value, highest Module_Reference of the modules that can be used in this slot.

Allowed_Values:
Type: Data_Type_Array (256) of Unsigned16
Meaning: Permitted values, list of Module_Reference of the modules that can be used in this slot.

B.5.4.3 Slave related keywords for DP extensions

PROFIBUS extensions mean the features of DP-V1 (see IEC 61784-1:2003 A3.1) and list of options (see IEC 61784-1:2003 A3.1 and 7.2.3.2.5), compared to DP-V0.

Table B.3 illustrates the dependence of GSD keywords regarding the PROFIBUS DP extensions. Some of the keywords become only valid when other keywords (main selectors for DP-V1 protocol functions) are set TRUE. The right column of the table shows the resulting features and behavior of the device described by the GSD definitions of the left two columns.

In this GSD description the acyclic channel between master class1 and slave has the name MS1 and between master class2 and slave has the name MS2.

NOTE The corresponding names in previous documents are MSAC_C1 and MSAC_C2.

A configuration tool for the DP extensions has to handle the defined first three byte of the user parameter data itself. These bytes can also be defined by the known mechanism of the GSD (Ext_User_Prm_Dat_Ref,...), but the configuration tool for the DP extensions overwrites than GSD definitions. At last these bytes can be defined by the keywords for DP extensions, the configuration tool for the DP extensions overwrites the definitions from the user parameter and ext user parameter.

Table B.3 — GSD keywords

Main Condition	Additional Condition	Conclusion
DPV1_Slave=0		Device is conform to PROFIBUS DP-V0, see IEC 61784-1:2003 A3.1. Device can not be operated with the following DP extensions (no acyclic services MS1, no data type support, no DP-V1 specific parameterization, no DP-V1 diagnosis model)
DPV1_Slave=0	C1_Read_Write_supp = 1 or DPV1_Data_Types = 1 or Check_Cfg_Mode = 1	invalid combination
DPV1_Slave=1		Device is conform to PROFIBUS DP-V1 extensions, see IEC 61784-1:2003 A3.1. Device supports DP-V1 specific parameterization and DP-V1 diagnosis model. This is an assumption for acyclic services MS1, Data_Type and Check_Cfg_Mode, which are supported as stated by the corresponding keywords.
DPV1_Slave=1 and C1_Read_Write _supp =0	C1_Max_Data_Len > 0 or C1_Response_Time out > 0 or C1_Read_Write _required = 1 or Diagnostic_Alarm_supp = 1 or Process_Alarm_supp= 1 or Pull_Plug_Alarm_supp= 1 or Status_Alarm_supp = 1 or Update_Alarm_supp = 1 or Manufacturer_Specific_Alarm_supp = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp =1		Device conforms to PROFIBUS DP-V1 extensions, see IEC 61784-1:2003 A3.1 and supports MS1 connection. This is an assumption for defining features of the MS1 connection and for Alarm support which are stated by the corresponding keywords.
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Diagnostic_Alarm_supp = 0	Diagnostic_Alarm _required = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Process_Alarm_supp = 0	Process_Alarm _required = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Pull_Plug_Alarm_supp = 0	Pull_Plug_Alarm _required = 1	Invalid combination

Table B.3 — GSD keywords *(continued)*

Main Condition	Additional Condition	Conclusion
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Status_Alarm_supp = 0	Status_Alarm _required = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Status_Alarm_supp = 0	Status_Alarm_required = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Update_Alarm_supp = 0	Update_Alarm_required = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp = 1 and Manufacturer_Specific_Alarm_supp = 0	Manufacturer_Specific_Alarm_required = 1	Invalid combination
DPV1_Slave=1 and C1_Read_Write _supp =1 and Diagnostic_Alarm_supp = 1 or Process_Alarm_supp= 1 or Pull_Plug_Alarm_supp= 1 or Status_Alarm_supp = 1 or Update_Alarm_supp = 1 or Manufacturer_Specific_Alarm_supp = 1		Device is conform to PROFIBUS DP extensions and supports MS1 connection and Alarms. This is an assumption for defining features of the Alarms, which are stated by the corresponding keywords.
C2_Read_Write _supp =0	C2_Max_Data_Len > 0 or C2_Response_Timeout > 0 or C2_Read_Write _required =1 or C2_Max_Count_Channels > 0 or Max_Initiate_PDU _Length > 0	Invalid combination
C2_Read_Write _supp =1		Device supports MS2 connection. The support of DP-V1 specific parameterization and DP-V1 diagnosis model is strongly recommended for migration of the whole DP extensions. Features of the MS2 connection are stated by the corresponding keywords.
WD_Base_1ms _supp		This works independent from the other PROFIBUS DP extensions. The assumption is that User_Prm_Data_Len > 0 are supported.

DPV1_Slave (D starting with GSD_Revision 3)
True, if the device uses DP-V1 functionality. This keyword is an extension to "Station_Type" and indicates if the slave operates as a standard DP- or DP-Slave with extended functionality.

The support of the several DP-V1 functionalities is defined in the following function specific keywords.
Type: Boolean (1: True)

C1_Read_Write_supp (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module with extended functionality is supporting the Read and Write services on the C1-communication relationship.
Type: Boolean (1: True)

C2_Read_Write_supp (D starting with GSD_Revision 3)
The DP-Slave with extended functionality is supporting the Read and Write services on the C2-communication relationship.
Type: Boolean (1: True)

C1_Max_Data_Len: (D starting with GSD_Revision 3)
The parameter specifies the maximum length of user data, excluding Function_Num, Slot_Number, Index, and Length, transferred on the MSAC_1 communication channel. This parameter is mandatory if C1_read_write_supp = 1.
Type: Unsigned8 (0 .. 240)

C2_Max_Data_Len: (D starting with GSD_Revision 3)
The parameter specifies the maximum length of user data, excluding Function_Num, Slot_Number, Index, and Length, transferred on the MSAC_2 communication channel. This parameter is mandatory if C2_read_write_supp = 1.
Type: Unsigned8 (0,48 .. 240)

C1_Response_Timeout: (O starting with GSD_Revision 3)
The parameter C1_Response_Timeout represents the efficiency of a DP-Slave with extended functionality. Each DP-Slave with extended functionality has to ensure that the parameter C1_Response_Timeout reaches the smallest value that is possible. By means of this parameter the DP-Slave with extended functionality indicates the maximum time to process an acyclic service (read, write, alarm_ack) on the C1-communication relationship. This parameter is mandatory if C1_read_write_supp = 1.
Type: Unsigned16 (1 .. 65535)
Time base: 10 ms

C2_Response_Timeout: (O starting with GSD_Revision 3)
The parameter C2_Response_Timeout represents the efficiency of a DP-Slave with extended functionality. Each DP-Slave with extended functionality has to ensure that the parameter C2_Response_Timeout reaches the smallest value that is possible. By means of this parameter the DP-Slave with extended functionality indicates the maximum time to process an acylic service (read, write, Data_Transport) on the C2-communication relationship. This parameter is mandatory if C2_read_write_supp = 1.
Type: Unsigned16 (1 .. 65535)
Time base: 10 ms

C1_Read_Write_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires C1_Read_Write services to be accessed.
Type: Boolean (1: True)

C2_Read_Write_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires C2_Read_Write services to be accessed.
Type: Boolean (1: True)

C2_Max_Count_Channels: (D starting with GSD_Revision 3)
The parameter defines the maximal amount of active C2 channels of the DP-V1 Slave. This parameter is mandatory if C2_read_write_supp = 1.
Type: Unsigned8 (0 .. 49)

Max_Initiate_PDU_Length: (D starting with GSD_Revision 3)
The parameter specifies the maximum length of an Initiate Request PDU including the Function_Num to the Resource Manager. This parameter is mandatory if C2_read_write_supp = 1.
Type: Unsigned8 (0,52 .. 244)

Diagnostic_Alarm_supp (D starting with GSD_Revision 3)
The device supports Diagnostic_Alarm. A diagnostic alarm signals an event within a slot, for instance over temperature, short circuit, etc..
Type: Boolean (1: True)

Process_Alarm_supp (D starting with GSD_Revision 3)
The device supports Process_Alarm. A process alarm signals the occurrence of an event in the connected process, for instance upper limit value exceeded.
Type: Boolean (1: True)

Pull_Plug_Alarm_supp (D starting with GSD_Revision 3)
The device supports Pull_ Plug_Alarm. A pull alarm signals the withdrawal of a module at a slot.
Type: Boolean (1: True)

Status_Alarm_supp (D starting with GSD_Revision 3)
The device supports Status_Alarm. A status alarm signals a change in the state of a module, for instance run, stop or ready.
Type: Boolean (1: True)

Update_Alarm_supp: (D starting with GSD_Revision 3)
The device supports Update_Alarm. An update alarm signals the change of a parameter in a slot e.g. by a local operation or remote access.
Type: Boolean (1: True)

Manufacturer_Specific_Alarm_supp: (D starting with GSD_Revision 3)
The device supports Manufacturer_Specific_Alarm. A manufacturer specific alarm signals an event defined by the manufacturer.
Type: Boolean (1: True)

Extra_Alarm_SAP_supp (D starting with GSD_Revision 3)
Additional to SAP51 it is possible to handle the MSAL_Alarm_Ack via SAP 50 if the Bit Sl_Flag.Extra_Alarm_SAP in the corresponding Slave Parameter Set is set. In this case there may be a higher performance because SAP 50 is used exclusively for the MSAL_Alarm_Ack service and the service cannot be delayed by a running MSAC1_Write or MSAC1_Read service.
Type: Boolean (1: True)

Alarm_Sequence_Mode_Count: (D starting with GSD_Revision 3)
The DP-Slave supports the Alarm_Sequence_Mode for alarm handling when this parameter is not 0. If this parameter is set to 0 only the Type Mode is supported by the slave.
The Sequence Mode is an option of the parallel alarm handling.
Several alarms (2 to 32) of the same or different type can be active (unacknowledged) at one time (fixed by the DDLM_Set_Prm service) at the DP-V1 Slave.
Type: Unsigned8 (0, 2 .. 32)

Alarm_Type_Mode_supp: (D starting with GSD_Revision 3;
M if the DP-Slave supports alarms, starting with GSD_Revision 4)
The DP-Slave supports the Type Mode for alarm handling. The Type Mode is mandatory if the DP-Slave supports alarms. Only one alarm of a specific Alarm_Type can be active at one time (fixed by the DDLM_Set_Prm service).
Type: Boolean (shall always be set to 1: True)

Diagnostic_Alarm_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires alarm handling to be accessed.
Type: Boolean (1: True)

Process_Alarm_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires alarm handling to be accessed.
Type: Boolean (1: True)

Pull_Plug_Alarm_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires alarm handling to be accessed.
Type: Boolean (1: True)

Status_Alarm_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires alarm handling to be accessed.
Type: Boolean (1: True)

Update_Alarm_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires alarm handling to be accessed.
Type: Boolean (1: True)

Manufacturer_Specific_Alarm_required: (D starting with GSD_Revision 3)
The DP-Slave or a Slave Module requires alarm handling to be accessed.
Type: Boolean (1: True)

DPV1_Data_Types: (O starting with GSD_Revision 3)
The DP-Slave uses the vendor specific data of the extended identifier format for all modules with extended identifier format for coding of data types.
Type: Boolean (1: True)

WD_Base_1ms_supp: (D starting with GSD_Revision 3)
The DP-Slave supports the time base of 1 millisecond for the watchdog.
Type: Boolean (1: True)

Check_Cfg_Mode: (D starting with GSD_Revision 3)
With this parameter the slave indicates the possibility of a different user specific way to check the Cfg-Data. This mode is switched on by the "Check_Cfg_Mode" in the DPV1_Status_2 of the Prm data.
Type: Boolean (1: True)

B.5.4.4 Slave related keywords for Data Exchange with Broadcast

Publisher_supp: (D starting with GSD_Revision 3)
The DP-Slave supports the Publisher functionality of Data Exchange with Broadcast.
Type: Boolean (1: True)

Subscriber_supp: (D starting with GSD_Revision 4)
The DP-Slave supports the Subscriber functionality of Data Exchange with Broadcast. If Subscriber_supp = 1, DPV1_Slave shall be 1.
Type: Boolean (1: True)

NOTE In order to secure the optimized performance of the publisher / subscriber functionality it is necessary to set the MaxTsdr_xx values (see B.5.2.1) according to the actual values of the device.

DXB_Max_Link_Count: (O starting with GSD_Revision 4)
The maximum number of supported links to different publishers. Has to be unequal 0, if Subscriber_supp = 1.
Type: Unsigned8 (0 - 125)

DXB_Max_Data_Length: (O starting with GSD_Revision 4)
The maximum data length (in one piece) for a supported link to one publisher. Has to be unequal 0, if Subscriber_supp = 1.
Type: Unsigned8 (1 - 244)

DXB-Subscribertable_Block_Location: (D starting with GSD_Revision 5)
This parameter indicates what type of SAP is supported by the DXB-Subscribertable.
Type: unsigned 8
0: Prm-Telegram
1: Prm-Telegram or Ext-Prm-Telegram;
Only allowed, if X_Prm_SAP_supp = 1.
2: Ext-Prm-Telegram;
Only allowed, if X_Prm_SAP_supp = 1.

EXAMPLE
```
; Slave related keywords for DXB - Start
Publisher_supp      =  1;
Subscriber_supp     =  1;
DXB_Max_Link_Count  = 10;
DXB_Max_Data_Length = 32;
; Slave related keywords for DXB - End
```

B.5.4.5 Slave related keywords for Isochronous Mode

Isochron_Mode_supp: (D starting with GSD_Revision 4)
This parameter indicates if the slave supports the Isochron_Mode. If the parameter is set to FALSE, all other isochronous parameters are not significant.
Type: Boolean (1: True)

Isochron_Mode_required: (D starting with GSD_Revision 4)
This parameter indicates whether the slave does require the master to support Isochron_Mode. If the parameter is set to TRUE, the slave cannot be operated by a master that does not support Isochron_Mode
Type: Boolean (1: True)

TBASE_DP: (O starting with GSD_Revision 4)
Time base of T_{DP}, the DP cycle time, TDP_MIN and TDP_MAX, in units of $^{1}/_{12}$ µs. The smallest value shall be declared. This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned32, allowed values are 375,750,1500,3000,6000,12000 which correspond to 31.25,62.5,125,250,500,1000µs respectively.

NOTE 1 A configuration tool will calculate the smallest common value for T_{DP}, T_I and T_O for all corresponding slaves at the bus.

TDP_MIN: (O starting with GSD_Revision 4)
Minimum of T_{DP}, the DP cycle time, based on TBASE_DP. The values of this parameter for higher time bases of T_{DP} shall be calculated out of this value . This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned16, with range from 1 to 2^{16}-1

TDP_MAX: (O starting with GSD_Revision 4)
The maximum DP cycle time supported by the DP device in Isochron mode, based on TBASE_DP. The values of this parameter for higher time bases of T_{DP} shall be calculated out of this value . TDP_MAX shall not exceed the range of 32ms. This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned16, with range from 1 to 2^{16}-1

T_PLL_W_MAX: (O starting with GSD_Revision 4)
The maximum value of the jitter which is acceptable at the device input (RS485 receiver) based on $^{1}/_{12}$ µs. This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned16, with range from 12 to 2^{16}-1

TBASE_IO: (O starting with GSD_Revision 4)
Time base of T_I and T_O, where T_I is the point in time when the input values are collected and T_O is the point in time when the output values are taken over. The allowed values for the time base are equal to the definition for TBASE_DP (see above). The smallest value shall be declared. This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned32

TI_MIN: (O starting with GSD_Revision 4)
The minimum time based on TBASE_IO that is necessary to get and update the input values of an individual DP Slave. The values of this parameter for higher time bases of T_I and T_O shall be calculated out of this value . This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned16, with range from 0 (special case), 1 to 2^{16}-1

NOTE 2 The values TI_MIN = TO_MIN = 0 shall cause the master to set the values $T_I = T_O = 0$. With the values $T_I = T_O = 0$ the "simple mode" of a PROFIdrive slave is adjusted according to [15] V3.1 page 98.

TO_MIN: (O starting with GSD_Revision 4)
The minimum time based on TBASE_IO that is necessary at the end of the cyclic part of the Isochron DP cycle (T_{DX}) to get and output the output values given in units of TBASE_IO of an individual DP Slave. . The values of this parameter for higher time bases of T_I and T_O shall be calculated out of this value . This parameter is mandatory if Isochron_Mode_supp = 1.
Type: Unsigned16

EXAMPLE

; Slave related keywords for Isochronous Mode - Start
Isochron_Mode_supp = 1
Isochron_Mode_required = 0
TBASE_DP = 1500 ; equal to 125µs
TDP_MAX = 256 ; 256 * 125µs = 32ms
TDP_MIN = 16 ; 16 * 125µs = 2ms
TBASE_IO = 1500 ; equal to 125µs
TI_MIN = 1 ; 1 * 125µs = 125µs
TO_MIN = 1 ; 1 * 125µs = 125µs
T_PLL_W_MAX = 12 ; equal $12*^{1}/_{12}$ µs = 1µs
; Slave related keywords for Isochronous Mode – End

This example means, the device supports Isochron_Mode and can be run by either master whether it supports Isochron_Mode or not. Further, the time base for both, the DP cycle time and the TI/TO values is 1500 which corresponds to 125µs. Therefore the minimal DP cycle time necessary for 3 Mbit/s is 16*125µs which equals 2ms, for 6 Mbit/s is 8*125µs which equals 1ms, the maximum cycle time supported by the device is 256*125ms which equals 32ms, the TI and TO can be calculated with 125µs each (TO 125ms greater than TDX), the maximum value of the jitter is 12*1/12 µs which equals 1 µs.

B.5.4.6 Slave related keywords for PROFIsafe Profile

A DP-Slave device that implements a behavior according to the PROFIsafe profile shall specify its capabilities and the user parameters with the following set of keywords.

NOTE Further information to PROFIsafe is provided in [13].

F_ParamDescCRC (O starting with GSD_Revision 4)
In order to read the PROFIsafe parameter description safely from the GSD file, 2 byte of CRC code are necessary. The CRC code has to be calculated according to the PROFIsafe guidelines and certified by a registered authority (e.g. TUEV). The value of this parameter will not be transferred to the slave device but is needed to avoid errors during the parameterization with the configuration tool.
Type: Unsigned16

F_Ext_User_Prm_Data_Ref: (O starting with GSD_Revision 4)
Here, a reference to a User_Prm_Data description is specified. The definition of this keyword excludes the evaluation of User_Prm_Data. If areas overlap when describing the ExtUserPrmData, the area defined last in the device description block has priority.
Parameters used:
Reference_Offset:
Type: Unsigned8
Meaning: Here, the offset within the associated part of the ExtUserPrmData is defined.

Reference_Number:
Type: Unsigned16
Meaning: This reference number has to be the same as the reference number that is defined in the ExtUserPrmData description.

F_Ext_User_Prm_Data_Const: (O starting with GSD_Revision 4)
Here, a constant part of the ExtUserPrmData is specified. The definition of this keyword excludes the evaluation of User_Prm_Data. If areas overlap when describing the ExtUserPrmData, the area defined last in the GSD file has priority.
Parameters used:
Const_Offset:
Type: Unsigned8
Meaning: Here, the offset within the associated part of User_Prm_Data is defined.

Const_Prm_Data:
Type: Octet-String
Meaning: Here, the constants or default selections within the ExtUserPrmData are defined.

B.5.4.7 Slave related keywords for extended parameterization

X_Prm_SAP_supp: (D starting with GSD_Revision 4)
Indicates, if the X_Prm_SAP is supported by the slave. Shall be true, if DPV1_Slave = 1.
Type: Boolean (1: True)

X_Max_User_Prm_Data_Len: (O starting with GSD_Revision 4)
Here, the maximum length of the ExtUserPrmData is specified. The use of this keyword is only allowed if DPV1_Slave = 1 and if X_Prm_SAP_supp = 1. Mandatory, if X_Prm_SAP_supp = 1.
Type: Unsigned8 (5 - 244)

X_Max_Sum_Prm_Data_Len: (O starting with GSD_Revision 5)
Here, the largest sum of the lengths of the UserPrmData and ExtUserPrmData is specified in bytes. X_Max_Sum_Prm_Data_Len shall be in minimum the highest value of Max_User_Prm_Data_Len and X_Max_User_Prm_Data_Len, in maximum the sum of both. If this keyword is not provided, the maximum length is the sum of X_Max_User_Prm_Data_Len and Max_User_Prm_Data_Len.
Type: Unsigned16 (0, 5 - 481)

X_Ext_Module_Prm_Data_Len: (O starting with GSD_Revision 4)
Here, the length of the associated ExtUserPrmData is defined.
The use of this keyword is only allowed if DPV1_Slave = 1 and if X_Prm_SAP_supp = 1.
Type: Unsigned8 (1 - 244)

X_Ext_User_Prm_Data_Ref: (O starting with GSD_Revision 4)
Here, a reference to ExtUserPrmData description is specified. If areas overlap when describing the ExtUserPrmData, the area defined last in the GSD file has priority.
Parameters used:
Reference_Offset:
Type: Unsigned8
Meaning: Here, the offset within the associated part of the ExtUserPrmData is defined.

Reference_Number:
Type: Unsigned16
Meaning: This reference number has to be the same as the reference number that is defined in the ExtUserPrmData description.

X_Ext_User_Prm_Data_Const: (O starting with GSD_Revision 4)
Here, a constant part of the ExtUserPrmData is specified. If areas overlap when describing the ExtUserPrmData, the area defined last in the GSD file has priority.
Parameters used:
Const_Offset:
Type: Unsigned8
Meaning: Here, the offset within the associated part of ExtUserPrmData is defined.

Const_Prm_Data:
Type: Octet-String
Meaning: Here, the constants or default selections within the ExtUserPrmData are defined.

X_Prm_Block_Structure_supp: (O starting with GSD_Revision 4)
Here, the slave indicates that the block structure of the extended parameterization is supported when using the X_Prm_Service.
Shall be true, if DPV1_Slave = 1.
Shall be true, if X_Prm_SAP_supp = 1.
Type: Boolean (1: True)

B.5.4.8 Slave related keywords for subsystems

A PROFIBUS DP slave device which has gateway capability towards an underlying communication system, also called *subsystem*, can provide a directory which holds DP indexes of the internal buffers representing the addressable Process Data objects. The user needs the information where to find this directory in order to get access to the data buffers representing the underlying communication system. The device manufacturer may provide one directory in slot 0 (this makes sense for a compact slave) or one directory in each slot for a modular slave.

Both keywords are optional, but only one keyword shall be used at the same time. This is because a modular slave could also use slot 0 for this directory, which is then valid for all type of modules. In that case no module specific definition is required.

Subsys_Dir_Index: (O starting with GSD_Revision 4)
The device has capabilities of a gateway towards a subsystem. The index of the subsystem object directory is given by this value. This definition has to appear within the unit definition. In order to decode the directory, the kind of the subsystem shall be specified in brackets.
Type of Index: unsigned 8
Type of Subsystem: unsigned 8, the values standing for:

1:	Gateway capability according to [13]
0, 2 .. 127:	Reserved
128 .. 255:	User specific

EXAMPLE 1

Subsys_Dir_Index (1) = 15

means, the device is a gateway with a subsystem master device according to [13] where the subsystems master device object directory can be found in slot 0 at index 15.

Subsys_Module_Dir_Index: (O starting with GSD_Revision 4)

The device has capabilities of a gateway towards a subsystem. The index of the subsystem object directory is module specific and is given by this value. The slot corresponds to the module.

This definition has to appear within the module definition In order to decode the directory, the kind of the subsystem shall be specified.

Type of Index: unsigned 8

Type of Sybsystem: unsigned 8, the values standing for:

1:	Gateway capability according to [13]
0, 2 .. 127:	Reserved
128 .. 255:	User specific

EXAMPLE 2

Subsys_Module_Dir_Index (1) = 42

means, the device is a gateway with a subsystem master device according to [13]. The subsystem master device object directory of the module where this definition appears can be found in the corresponding slot at index 42.

B.6 Formal description of GSD

Table B.4 specifies GSD in a formal way. All data in brackets are optional. The symbol "|" means the logical or-operation. The number before every rule is a sequence number (S#) enabling the rules to be referenced.

Table B.4 — Formal Description of GSD format

S#	Formal description
255)	<Backslash> = \
254)	<Long-Line> = <Backslash><LineEnd>
253)	<WS> = <Space> | <Tab> | <Long-Line> | <WS><Space> | <WS><Tab> | <WS><Long-Line>
252)	<CRLF> = <Carriage Return><Line Feed> | <Carriage Return> | <Line Feed>
251)	<Num> = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
250)	<Namechar> = a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | _ | . | - | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | <Num>
249)	<Otherchar> = + | * | / | < | > | (|) | [|] | { | } | ! | $ | % | & | ? | ´ | ^ | | | = | # | ; | , | : | `
248)	<Baudrate> = 9.6 | 19.2 | 31.25 | 45.45 | 93.75 | 187.5 | 500 | 1.5M | 3M | 6M | 12M
247)	<Stringchar> = <Namechar> | <Otherchar>
246)	<Char> = <Stringchar> | "
245)	<Com> = ; | <Com><Char> | <Com><WS>
244)	<ComLn> = <Com><CRLF>
243)	<LineStart> = [<WS>] | [<WS>]<LineEnd><LineStart> { empty line }
242)	<LineEnd> = <CRLF> | <Com><CRLF> | <WS><LineEnd> | <LineEnd><ComLn> | <LineEnd><CRLF>
241)	<Boolean> = 0 | 1
240)	<Decimal> = <Num> | <Decimal><Num>
239)	<Hexchar> = <Num> | A | B | C | D | E | F | a | b | c | d | e | f
238)	<Hexadecimal> = 0x<Hexchar> | <Hexadecimal><Hexchar>
237)	<Number> = <Decimal> | <Hexadecimal>
236)	<Octet> = <Number> { 0 <= <Octet> <= 255 }
235)	<Unsigned8> = <Octet>
234)	<Unsigned16> = <Number> {0<=<Unsigned16><=65535}
233)	<Unsigned32> = <Number> { 0 <= <Unsigned32> <= 4294967295 }
232)	<Signed8> = [-] <Number> { -128 <= <Signed8> <= 127 }
231)	<Signed16> = [-] <Number> { -32768 <= <Signed16> <= 32767 }
230)	<Signed32> = [-] <Number> { -2147483648 <= <Signed32> <= 2147483647 }
229)	<Octet-String> = [<WS>]<Octet> | <Octet-String>[<WS>],[<WS>]<Octet>
228)	<String> = <Stringchar> | <Space> | <String><Stringchar> | <String><Space>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
227)	<Visible-String> = "<String>"
226)	<Keyword> = <Namechar> | <Keyword><Namechar>
225)	<Any-String> = <Char> | <WS> | <Any-String><Char> | <Any-String><WS>
224)	<Any-Line> = <CRLF> | <Any-String><CRLF>
223)	<Any-Text> = <Any-Line> | <Any-Text><Any-Line>
222)	<User-Definition> = <Keyword>[<WS>] [<Otherchar><Any-Line>]
221)	<GSD_Revision> = <Unsigned8>
220)	<Vendor_Name> = <Visible-String> { Length <= 32 }
219)	<Model_Name> = <Visible-String> { Length <= 32 }
218)	<Revision> = <Visible-String> { Length <= 32 }
217)	<Revision_Number> = <Unsigned8>
216)	<Ident_Number> = <Unsigned16>
215)	<Protocol_Ident> = <Unsigned8>
214)	<Station_Type> = <Unsigned8>
213)	<FMS_supp> = <Boolean>
212)	<Hardware_Release> = <Visible-String> { Length <= 32 }
211)	<Software_Release> = <Visible-String> { Length <= 32 }
210)	<Baudrate_supp> = <Boolean>
209)	<MaxTsdr> = <Unsigned16>
208)	<Redundancy> = <Boolean>
207)	<Repeater_Ctrl_Sig> = <Unsigned8>
206)	<24V_Pins> = <Unsigned8>
205)	<Implementation_Type> = <Visible-String> { Length <= 32}
204)	<Bitmap_Device> = <Visible-String> { Length <= 8}
203)	<Bitmap_Diag> = <Visible-String> { Length <= 8}
202)	<Bitmap_SF> = <Visible-String> { Length <= 8}
201)	<Transmission_Delay> = <Unsigned16>
200)	<Reaction_Delay> = <Unsigned16>
199)	<Master_Freeze_Mode_supp> = <Boolean>
198)	<Master_Sync_Mode_supp> = <Boolean>
197)	<Master_Fail_Safe_supp> = <Boolean>
196)	<Download_supp> = <Boolean>
195)	<Upload_supp> = <Boolean>
194)	<Act_Para_Brct_supp> = <Boolean>
193)	<Act_Param_supp> = <Boolean>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
192)	<Max_MPS_Length> = <Unsigned32>
191)	<Max_Lsdu_MM> = <Unsigned8>
190)	<Max_Lsdu_MS> = <Unsigned8>
189)	<Min_Poll_Timeout> = <Unsigned16>
188)	<Trdy> = <Unsigned8>
187)	<Tqui> = <Unsigned8>
186)	<Tset> = <Unsigned8>
185)	<Tsdi> = <Unsigned16>
184)	<LAS_Len> = <Unsigned8>
183)	<Max_Slaves_supp> = <Unsigned8>
182)	<Max_Master_Input_Len> = <Unsigned8>
181)	<Max_Master_Output_Len>= <Unsigned8>
180)	<Max_Master_Data_Len> = <Unsigned16>
179)	<Isochron_Mode_Synchronised> = <Unsigned8>
178)	<DXB_Master_Supp> = <Boolean>
177)	<X_Master_Prm_SAP_supp> = <Boolean>
176)	<DPV1_Master> = <Boolean>
175)	<DPV1_Conformance_Class> = <Unsigned8>
174)	<C1_Master_Read_Write_supp> = <Boolean>
173)	<Master_DPV1_Alarm_supp> = <Boolean>
172)	<Master_Diagnostic_Alarm_supp> = <Boolean>
171)	<Master_Process_Alarm_supp> = <Boolean>
170)	<Master_Pull_Plug_Alarm_supp> = <Boolean>
169)	<Master_Status_Alarm_supp> = <Boolean>
168)	<Master_Update_Alarm_supp> = <Boolean>
167)	<Master_Manufacturer_Specific_Alarm_supp> = <Boolean>
166)	<Master_Extra_Alarm_SAP_supp> = <Boolean>
165)	<Master_Alarm_Sequence_Mode> = <Unsigned8>
164)	<Master_Alarm_Type_Mode_supp> = <Boolean>
163)	<Freeze_Mode_supp> = <Boolean>
162)	<Sync_Mode_supp> = <Boolean>
161)	<Set_Slave_Add_supp> = <Boolean>
160)	<Auto_Baud_supp> = <Boolean>
159)	<User_Prm_Data_Len> = <Unsigned8>
158)	<User_Prm_Data> = <Octet-String>
157)	<Min_Slave_Intervall> = <Unsigned16>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
156)	<Modular_Station> = <Boolean>
155)	<Max_Module> = <Unsigned8>
154)	<Max_Input_Len> = <Unsigned8>
153)	<Max_Output_Len> = <Unsigned8>
152)	<Max_Data_Len> = <Unsigned16>
151)	<Modul_Offset> = <Unsigned8>
150)	<Bit> = <Unsigned16>
149)	<Diag_Text> = <Visible-String> { Length <= 32}
148)	<Help_Text> = <Visible-String> { Length <= 256}
147)	<First_Bit> = <Bit>
146)	<Last_Bit> = <Bit>
145)	<Value> = <Unsigned16>
144)	<Mod_Name> = <Visible-String> { Length <= 32}
143)	<Config> = <Octet-String>
142)	<Error_Type> = <Unsigned8> { 16 <= <Error_Type> <= 31 }
141)	<Subfamily_Name> = <String> { Length <= 32}
140)	<Family_Name> = <Unsigned8> | <Unsigned8>@<Subfamily_Name> | <Unsigned8>@<Subfamily_Name> @<Subfamily_Name> | <Unsigned8>@<Subfamily_Name> @<Subfamily_Name>@<Subfamily_Name>
139)	<Info_Text> = Info_Text[<WS>]=[<WS>]<Visible-String>{Length<=256}
138)	<Prm_Block_Structure_req> = <Boolean>
137)	<Prm_Block_Structure_supp> = <Boolean>
136)	<Jokerblock_supp> = <Boolean>
135)	<Jokerblock_Type> = <Boolean>
134)	<Jokerblock_Slot> = <Boolean>
133)	<Jokerblock_Location> = <Boolean>
132)	<Jokerblock-Item> = Jokerblock_Slot[<WS>]=[<WS>]<Jokerblock_Slot> | Jokerblock_Location[<WS>]= [<WS>]<Jokerblock_Location>
131)	<Jokerblock-List> = <Jokerblock-Item> | <Jokerblock-List><Jokerblock-Item>
130)	<Jokerblock-Def> = Jokerblock_Type[<WS>]= [<WS>]<Jokerblock_Type><LineEnd> <Jokerblock-List> End_Jokerblock_Type
129)	<Fail_Safe> = <Boolean>
128)	<Fail_Safe_Required> = <Boolean>
127)	<Max_Diag_Data_Len> = <Unsigned8>
126)	<Diag_Update_Delay> = <Unsigned8>
125)	<PrmCmd_supp> = <Boolean>
124)	<Max_Switch_Over_Time> = <Unsigned16>
123)	<Slave_Redundancy_supp> = <Boolean>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
122)	<Ident_Maintenance_supp> = <Boolean>
121)	<Time_Sync_supp> = <Boolean>
120)	<DPV1_Slave> = <Boolean>
119)	<C1_Read_Write_supp> = <Boolean>
118)	<C2_Read_Write_supp> = <Boolean>
117)	<C1_Max_Data_Len> = <Unsigned8>
116)	<C2_Max_Data_Len> = <Unsigned8>
115)	<C1_Response_Timeout> = <Unsigned16>
114)	<C2_Response_Timeout> = <Unsigned16>
113)	<C1_Read_Write_Required> = <Boolean>
112)	<C2_Read_Write_Required> = <Boolean>
111)	<C2_Max_Count_Channels> = <Unsigned8>
110)	<Max_Initiate_PDU_Length> = <Unsigned8>
109)	<Diagnostic_Alarm_supp> = <Boolean>
108)	<Process_Alarm_supp> = <Boolean>
107)	<Pull_Plug_Alarm_supp> = <Boolean>
106)	<Status_Alarm_supp> = <Boolean>
105)	<Update_Alarm_supp> = <Boolean>
104)	<Manufacturer_Specific_Alarm_supp> = <Boolean>
103)	<Extra_Alarm_SAP_supp> = <Boolean>
102)	<Alarm_Sequence_Mode_Count> = <Unsigned8>
101)	<Alarm_Type_Mode_supp> = <Boolean>
100)	<Alarm_required> = <Boolean>
99)	<DPV1_Data_Types> = <Boolean>
98)	<WD_Base_1ms_supp> = <Boolean>
97)	<Check_Cfg_Mode> = <Boolean>
96)	<Max_User_Prm_Data_Len> = <Unsigned8>
95)	<Reference_Number> = <Unsigned16>
94)	<Reference_Offset> = <Unsigned8>
93)	<Const_Offset> = <Unsigned8>
92)	<Const_Prm_Data> = <Octet-String>
91)	<Module_Reference> = <Unsigned16>
90)	<Mod-Ref-String> = [<WS>]<Module_Reference> | <Mod-Ref-String>[<WS>],[<WS>]<Module_Reference>
89)	<Slot_Number> = <Unsigned8>
88)	<Slot_Name> = <Visible-String> { Length <= 32}
87)	<Bit-Area> = BITAREA(<First_Bit>-<Last_Bit>){0<=First_Bit<=Last_Bit<=7} {Value Range: UNSIGNED(Last_Bit-First_Bit+1)}
86)	<Data_Type_Name> = UNSIGNED8 | UNSIGNED16 | UNSIGNED32 | SIGNED8 | SIGNED16 | SIGNED32 | BIT(<Bit>) | <Bit-Area> {0<=Bit<=7}

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
85)	<Data_Type> = <Unsigned8> | <Unsigned16> | <Unsigned32> | <Signed8> | <Signed16> | <Signed32> | <Bit>
84)	<Data_Type_Array> = [<WS>]<Data_Type> | <Data_Type_Array>[<WS>],[<WS>]<Data_Type>
83)	<Default_Value> = <Data_Type>
82)	<Min_Value> = <Data_Type>
81)	<Max_Value> = <Data_Type>
80)	<Allowed_Values> = <Data_Type_Array>
79)	<Prm_Data_Value> = <Data_Type>
78)	<Prm_Text_Ref> = Prm_Text_Ref[<WS>]= [<WS>]<Reference_Number><LineEnd>
77)	<Ext_User_Prm_Data_Name> = <Visible-String> { Length <= 32}
76)	<Text> = <Visible-String> { Length <= 32}
75)	<First_Bit> = <Unsigned16>
74)	<Last_Bit> = <Unsigned16> {0 <= Last_Bit < =495}
73)	<X_Value_Item> = X_Value[<WS>](<Value>)[<WS>]= [<WS>]<Diag_Text><LineEnd> | X_Value_Help[<WS>](<Value>)[<WS>]= [<WS>]<Help_Text><LineEnd>
72)	<Value_Item> = Value[<WS>](<Value>)[<WS>]= [<WS>]<Diag_Text><LineEnd> | Value_Help[<WS>](<Value>)[<WS>]= [<WS>]<Help_Text><LineEnd>
71)	<X_Value_List> = <X_Value_Item> | <X_Value-List><X_Value-Item>
70)	<Value_List> = <Value_Item> | <Value-List><Value-Item>
69)	<X-Unit-Diag-Area-Def> = X_Unit_Diag_Area[<WS>]= [<WS>]<First_Bit>-<Last_Bit><LineEnd><X_Value_List> X_Unit_Diag_Area_End {0<=First_Bit<=Last_Bit<=495}
68)	<Unit-Diag-Area-Def> = Unit_Diag_Area[<WS>]= [<WS>]<First_Bit>-<Last_Bit><LineEnd><Value_List> Unit_Diag_Area_End {0<=First_Bit <= Last_Bit<=495}
67)	<X-Unit-Diag-Def> = X_Unit_Diag_Bit[<WS>](<Bit>)[<WS>]= [<WS>]<Diag_Text> {0<=Bit<=495} | X_Unit_Diag_Not_Bit[<WS>](<Bit>)[<WS>]= [<WS>]<Diag_Text> {0<=Bit<=495} | X_Unit_Diag_Bit_Help[<WS>](<Bit>)[<WS>]= [<WS>]<Help_Text> | X_Unit_Diag_Not_Bit_Help[<WS>](<Bit>)[<WS>]= [<WS>]<Help_Text> | <X-Unit-Diag-Area-Def>
66)	<Diag_Type_Number> = <Unsigned8>
65)	<Unit-Diag-List> = <X-Unit-Diag-Def> [|<Unit-Diag-List><X-Unit-Diag-Def>]<LineEnd>
64)	<Unit-Diag-Type-Def> = UnitDiagType[<WS>]= [<WS>]<Diag_Type_Number><LineEnd> <Unit-Diag-List> EndUnitDiagType
63)	<Channel-Diag-Definition> = Channel_Diag[<WS>](<Error_Type>)[<WS>]= [<WS>]<Diag_Text> | Channel_Diag_Help[<WS>](<Error_Type>)[<WS>]= [<WS>]<Help_Text><LineEnd>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
62)	<Ph_Delay_Item> = Transmission_Delay_9.6[<WS>]= [<WS>]<Transmission_Delay> | Transmission_Delay_19.2[<WS>] = [<WS>]<Transmission_Delay> | Transmission_Delay_31.25[<WS>]= [<WS>]<Transmission_Delay> | Transmission_Delay_45.45[<WS>]= [<WS>]<Transmission_Delay> | Transmission_Delay_93.75[<WS>]= [<WS>]<Transmission_Delay> | Transmission_Delay_187.5[<WS>]= [<WS>]<Transmission_Delay> | Transmission_Delay_500[<WS>] = [<WS>]<Transmission_Delay> | Transmission_Delay_1.5M[<WS>] = [<WS>]<Transmission_Delay> | Transmission_Delay_3M[<WS>] = [<WS>]<Transmission_Delay> | Transmission_Delay_6M[<WS>] = [<WS>]<Transmission_Delay> | Transmission_Delay_12M[<WS>] = [<WS>]<Transmission_Delay> | Reaction_Delay_9.6[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_19.2[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_31.25[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_45.45[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_93.75[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_187.5[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_500[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_1.5M[<WS>]= [<WS>]<Reaction_Delay> | Reaction_Delay_3M[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_6M[<WS>] = [<WS>]<Reaction_Delay> | Reaction_Delay_12M[<WS>] = [<WS>]<Reaction_Delay> <LineEnd>
61)	<Ph-Delay-List> = <Ph_Delay_Item> | <Ph-Delay-List><Ph_Delay_Item>
60)	<Ph-Interface-Def> = Physical_Interface[<WS>]=[<WS>]<Unsigned8><LineEnd> <Ph-Delay-List> End_Physical_Interface
59)	<Slot_Item> = Slot(<Slot_Number>)[<WS>]=[<WS>]<Slot_Name> <WS><Module_Reference> [<WS><Module_Reference>[<WS>]- [<WS>]<Module_Reference> | <WS><Mod-Ref-String>] <LineEnd>
58)	<Slot-List> = <Slot_Item> | <Slot-List><Slot_Item>
57)	<Slot-Def> = SlotDefinition<LineEnd> <Slot-List> EndSlotDefinition
56)	<Data-Type-Item> = Data-Type[<WS>]=[<WS>]<Unsigned8>
55)	<Data-Type-List> = <Data-Type-Item> | <Data-Type-List><Data-Type-Item>
54)	<Data-Area-Item> = Area_Name[<WS>] =[<WS>]<Visible-String><LineEnd> Related_CFG_Identifier[<WS>] = [<WS>]<Unsigned8> <LineEnd> Length[<WS>] =[<WS>]<Unsigned8><LineEnd> Consistency<WS> =[<WS>]<Unsigned8><LineEnd> Publisher_allowed[<WS>]=[<WS>]<Unsigned8><LineEnd> DP_Master_allowed[<WS>]=[<WS>]<Unsigned8><LineEnd> <Data-Type-List>
53)	<Data-Area-List> = <Data-Area-Item> | <Data-Area-List><Data-Area-Item>
52)	<Data-Area-Def> = Data_Area_Beg<LineEnd> <Data-Area-List> Data_Area_End

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
51)	<Alarm_Requirement> = Diagnostic_Alarm_required[<WS>] = [<WS>]<Alarm_required> | Process_Alarm_required[<WS>] = [<WS>]<Alarm_required> | Pull_Plug_Alarm_required[<WS>] = [<WS>]<Alarm_required> | Status_Alarm_required[<WS>] = [<WS>]<Alarm_required> | Update_Alarm_required[<WS>] = [<WS>]<Alarm_required> | Manufacturer_Specific_Alarm_required[<WS>]= [<WS>]<Alarm_required>
50)	<DXB-Subscribertable_Block_Location> = DXB- Subscribertable_Block_Location [<WS>]= [<WS>]<Unsigned 8>
49)	<DXB_Max_Data_Length> = DXB_Max_Data_Length[<WS>]=[<WS>]<Unsigned 8>
48)	<DXB_Max_Link_Count> = DXB_Max_Link_Count[<WS>]=[<WS>]<Unsigned 8>
47)	<Subscriber_supp> = Subscriber_supp[<WS>]=[<WS>]<Boolean>
46)	<Publisher_supp> = Publisher_supp[<WS>]=[<WS>]<Boolean>
45)	<DXB-List> = [<WS>]<Publisher_supp> | <Subscriber_supp> | <DXB_Max_Link_Count> | <DXB_Max_Data_Length> | <DXB-Subscribertable_Block_Location>
44)	<X_Prm_Block_Structure_supp> = [<WS>] X_Prm_Block_Structure_supp[<WS>]= [<WS>]<Boolean><LineEnd>
43)	<X_Ext_User_Prm_Data_Const> = [<WS>] X_Ext_User_Prm_Data_Const(<Const_Offset>)[<WS>]= [<WS>]<Const_Prm_Data><LineEnd>
42)	<X_Ext_User_Prm_Data_Ref> = [<WS>] X_Ext_User_Prm_Data_Ref(<Reference_Offset>)[<WS>]= [<WS>]<Reference_Number><LineEnd>
41)	<X_Max_User_Prm_Data_Len> = [<WS>] X_Max_User_Prm_Data_Len[<WS>]= [<WS>]<Unsigned8><LineEnd>
40)	<X_Prm_SAP_supp> = [<WS>] X_Prm_SAP_supp[<WS>]= [<WS>]<Boolean><LineEnd>
39)	<X-Prm-List> = [<WS>]<X_Prm_SAP_supp> | <X_Max_User_Prm_Data_Len> | <X_Ext_User_Prm_Data_Ref> | <X_Ext_User_Prm_Data_Const> | <X_Prm_Block_Structure_supp>
38)	<Isochron_Mode_supp> = [<WS>] Isochron_Mode_supp[<WS>]= [<WS>]<Boolean><LineEnd>
37)	<Isochron_Mode_required> = [<WS>] Isochron_Mode_required[<WS>]= [<WS>]<Boolean><LineEnd>
36)	<TBASE-DP> = [<WS>] TBASE_DP[<WS>]=[<WS>]<unsigned32><LineEnd>
35)	<TDP-MIN> = [<WS>] TDP_MIN[<WS>]=[<WS>]<unsigned16><LineEnd>
34)	<TDP-MAX> = [<WS>] TDP_MAX[<WS>]=[<WS>]<unsigned16><LineEnd>
33)	<T_PLL_W_MAX> = [<WS>] T_PLL_W_MAX[<WS>]=[<WS>]<unsigned16><LineEnd>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
32)	<TBASE-IO> = [<WS>] TBASE_IO[<WS>]=[<WS>]<unsigned32><LineEnd>
31)	<TI-MIN> = [<WS>] TI_MIN[<WS>]=[<WS>]<unsigned16><LineEnd>
30)	<TO-MIN> = [<WS>] TO_MIN[<WS>]=[<WS>]<unsigned16><LineEnd>
29)	<Isochron_Mode_List> = [<WS>]<Isochron_Mode_supp> | <Isochron_Mode_required> | <T_PLL_W_MAX> | <TBASE-DP> | <TDP-MIN> | <TDP-MAX> | <TBASE-IO> | <TI-MIN> | <TO-MIN>
28)	<Visible> = [<WS>] Visible[<WS>]=[<WS>]<Boolean><LineEnd>
27)	<Changeable> = [<WS>] Changeable[<WS>]=[<WS>]<Boolean><LineEnd>
26)	<F_Ext_User_Prm_Data_Const> = [<WS>] F_Ext_User_Prm_Data_Const(<Const_Offset>)[<WS>]= [<WS>]<Const_Prm_Data><LineEnd>
25)	<F_Ext_User_Prm_Data_Ref> = [<WS>] F_Ext_User_Prm_Data_Ref(<Reference_Offset>)[<WS>]= [<WS>]<Reference_Number><LineEnd>
24)	<F_ParamDescCRC> = [<WS>] F_ParamDescCRC[<WS>]=[<WS>]<unsigned16><LineEnd>
23)	<F-Param_List> = [<WS>]<F_ParamDescCRC> | <F_Ext_User_Prm_Data_Ref> | <F_Ext_User_Prm_Data_Const>
22)	<Subsys-Type>=<Unsigned8>
21)	<Subsys-Dir-Index>=<Unsigned8>
20)	<Subsys-Module-Dir-Index-Def> = Subsys_Module_Dir_Index[<WS>] (<Subsys-Type>)[<WS>]=[<WS>]<Subsys-Dir-Index>
19)	<Subsys-Dir-Index-Def> = Subsys_Dir_Index[<WS>] (<Subsys-Type>)[<WS>]=[<WS>]<Subsys-Dir-Index>
18)	<Ext-User-Prm-Data-Const> = Ext_User_Prm_Data_Const(<Const_Offset>)[<WS>]= [<WS>]<Const_Prm_Data>
17)	<Ext-User-Prm-Data-Ref> = Ext_User_Prm_Data_Ref(<Reference_Offset>)[<WS>]= [<WS>]<Reference_Number>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
16)	<Unit-Def-Item> = GSD_Revision[<WS>] =[<WS>]<GSD_Revision> | Vendor_Name[<WS>] =[<WS>]<Vendor_Name> | Model_Name[<WS>] =[<WS>]<Model_Name> | Revision[<WS>] =[<WS>]<Revision> | Revision_Number[<WS>]=[<WS>]<Revision_Number> | Ident_Number[<WS>] =[<WS>]<Ident_Number> | Protocol_Ident[<WS>] =[<WS>]<Protocol_Ident> | Station_Type[<WS>] =[<WS>]<Station_Type> | FMS_supp[<WS>] =[<WS>]<FMS_supp> | Hardware_Release[<WS>]=[<WS>]<Hardware-Release> | Software_Release[<WS>] =[<WS>]<Software-Release> | <Info_Text> | 9.6_supp[<WS>] =[<WS>]<Baudrate_supp> | 19.2_supp[<WS>] =[<WS>]<Baudrate_supp> | 31.25_supp[<WS>] =[<WS>]<Baudrate_supp> | 45.45_supp[<WS>] =[<WS>]<Baudrate_supp> | 93.75_supp[<WS>] =[<WS>]<Baudrate_supp> | 187.5_supp[<WS>] =[<WS>]<Baudrate_supp> | 500_supp[<WS>] =[<WS>]<Baudrate_supp> | 1.5M_supp[<WS>] =[<WS>]<Baudrate_supp> | 3M_supp[<WS>] =[<WS>]<Baudrate_supp> | 6M_supp[<WS>] =[<WS>]<Baudrate_supp> | 12M_supp[<WS>] =[<WS>]<Baudrate_supp> | MaxTsdr_9.6[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_19.2[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_31.25[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_45.45[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_93.75[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_187.5[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_500[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_1.5M[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_3M[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_6M[<WS>] =[<WS>]<MaxTsdr> | MaxTsdr_12M[<WS>] =[<WS>]<MaxTsdr> | Redundancy[<WS>] =[<WS>]<Redundancy> | Repeater_Ctrl_Sig[<WS>]=[<WS>]<Repeater_Ctrl_Sig> | 24V_Pins[<WS>] =[<WS>]<24V_Pins> | Implementation_Type[<WS>] = [<WS>]<Implementation_Type> | Bitmap_Device[<WS>] =[<WS>]<Bitmap_Device> | Bitmap_Diag[<WS>] =[<WS>]<Bitmap_Diag> | Bitmap_SF[<WS>] =[<WS>]<Bitmap_SF> | Master_Freeze_Mode_supp[<WS>] = [<WS>]<Master_Freeze_Mode_supp> | Master_Sync_Mode_supp[<WS>] = [<WS>]<Master_Sync_Mode_supp> | Master_Fail_Safe_supp[<WS>] = [<WS>]<Master_Fail_Safe_supp> | Download_supp[<WS>]=[<WS>]<Download_supp> | Upload_supp[<WS>]=[<WS>]<Upload_supp> | Act_Para_Brct_supp[<WS>] = [<WS>]<Act_Para_Brct_supp> | Act_Param_supp[<WS>] = [<WS>]<Act_Param_supp> | Max_MPS_Length[<WS>] =[<WS>]<Max_MPS_Length> | Max_Lsdu_MM[<WS>] =[<WS>]<Max_Lsdu_MM> | Max_Lsdu_MS[<WS>] =[<WS>]<Max_Lsdu_MS> | Min_Poll_Timeout[<WS>] =[<WS>]<Min_Poll_Timeout>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
16) cont.	| Trdy_9.6[<WS>] =[<WS>]<Trdy> | Trdy_19.2[<WS>] =[<WS>]<Trdy> | Trdy_31.25[<WS>] =[<WS>]<Trdy> | Trdy_45.45[<WS>] =[<WS>]<Trdy> | Trdy_93.75[<WS>] =[<WS>]<Trdy> | Trdy_187.5[<WS>] =[<WS>]<Trdy> | Trdy_500[<WS>] =[<WS>]<Trdy> | Trdy_1.5M[<WS>] =[<WS>]<Trdy> | Trdy_3M[<WS>] =[<WS>]<Trdy> | Trdy_6M[<WS>] =[<WS>]<Trdy> | Trdy_12M[<WS>] =[<WS>]<Trdy> | Tqui_9.6[<WS>] =[<WS>]<Tqui> | Tqui_19.2[<WS>] =[<WS>]<Tqui> | Tqui_31.25[<WS>] =[<WS>]<Tqui> | Tqui_45.45[<WS>] =[<WS>]<Tqui> | Tqui_93.75[<WS>] =[<WS>]<Tqui> | Tqui_187.5[<WS>] =[<WS>]<Tqui> | Tqui_500[<WS>] =[<WS>]<Tqui> | Tqui_1.5M[<WS>] =[<WS>]<Tqui> | Tqui_3M[<WS>] =[<WS>]<Tqui> | Tqui_6M[<WS>] =[<WS>]<Tqui> | Tqui_12M[<WS>] =[<WS>]<Tqui> | Tset_9.6[<WS>] =[<WS>]<Tset> | Tset_19.2[<WS>] =[<WS>]<Tset> | Tset_31.25[<WS>] =[<WS>]<Tset> | Tset_45.45[<WS>] =[<WS>]<Tset> | Tset_93.75[<WS>] =[<WS>]<Tset> | Tset_187.5[<WS>] =[<WS>]<Tset> | Tset_500[<WS>] =[<WS>]<Tset> | Tset_1.5M[<WS>] =[<WS>]<Tset> | Tset_3M[<WS>] =[<WS>]<Tset> | Tset_6M[<WS>] =[<WS>]<Tset> | Tset_12M[<WS>] =[<WS>]<Tset> | Tsdi_9.6[<WS>] =[<WS>]<Tsdi> | Tsdi_19.2[<WS>] =[<WS>]<Tsdi> | Tsdi_31.25[<WS>] =[<WS>]<Tsdi> | Tsdi_45.45[<WS>] =[<WS>]<Tsdi> | Tsdi_93.75[<WS>] =[<WS>]<Tsdi> | Tsdi_187.5[<WS>] =[<WS>]<Tsdi> | Tsdi_500[<WS>] =[<WS>]<Tsdi> | Tsdi_1.5M[<WS>] =[<WS>]<Tsdi> | Tsdi_3M[<WS>] =[<WS>]<Tsdi> | Tsdi_6M[<WS>] =[<WS>]<Tsdi> | Tsdi_12M[<WS>] =[<WS>]<Tsdi> | LAS_Len[<WS>] =[<WS>]<LAS_Len> | Max_Slaves_supp[<WS>] =[<WS>]<Max_Slaves_supp> | Max_Master_Input_Len[<WS>] = [<WS>]<Max_Master_Input_Len> | Max_Master_Output_Len[<WS>] = [<WS>]<Max_Master_Output_Len> | Max_Master_Data_Len[<WS>] = [<WS>]<Max_Master_Data_Len> | DPV1_Master[<WS>] =[<WS>]<DPV1_Master> | DPV1_Conformance_Class[<WS>]= [<WS>]<DPV1_Conformance_Class> | C1_Master_Read_Write_supp[<WS>] = [<WS>]< C1_Master_Read_Write_supp> | Master_DPV1_Alarm_supp[<WS>]= [<WS>]<Master_DPV1_Alarm_supp> | Master_Diagnostic_Alarm[<WS>] = [<WS>]<Master_Diagnostic_Alarm_supp> | Master_Process_Alarm_supp[<WS>] = [<WS>]<Master_Process_Alarm_supp> | Master_Pull_Plug_Alarm_supp[<WS>] = [<WS>]<Master_Pull_Plug_Alarm_supp> | Master_Status_Alarm_supp[<WS>] = [<WS>]<Master_Status_Alarm_supp> | Master_Update_Alarm_supp[<WS>] = [<WS>]<Master_Update_Alarm_supp> | Master_Manufacturer_Specific_Alarm_supp[<WS>] = [<WS>]<Master_Manufacturer_Specific_Alarm_supp> | Master_Extra_Alarm_SAP_supp[<WS>]= [<WS>]<Master_Extra_Alarm_SAP_supp> | Master_Alarm_Sequence_Mode[<WS>] = [<WS>]<Master_Alarm_Sequence_Mode> | Master_Alarm_Type_Mode_supp[<WS>] = [<WS>]<Master_Alarm_Type_Mode_supp> | X_Master_Prm_SAP_supp[<WS>] = [<WS>]< X_Master_Prm_SAP_supp> | DXB_Master_supp[<WS>] = [<WS>]<DXB_Master_supp> | Isochron_Mode_Synchronised[<WS>] = [<WS>]<Isochron_Mode_Synchronised> | Freeze_Mode_supp[<WS>] =[<WS>]<Freeze_Mode_supp> | Sync_Mode_supp[<WS>] =[<WS>]<Sync_Mode_supp> | Auto_Baud_supp[<WS>] =[<WS>]<Auto_Baud_supp> | Set_Slave_Add_supp[<WS>] = [<WS>]<Set_Slave_Add_supp>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
16) cont.	| User_Prm_Data_Len[<WS>] = [<WS>]<User_Prm_Data_Len> | User_Prm_Data[<WS>] =[<WS>]<User_Prm_Data> | Min_Slave_Intervall[<WS>] = [<WS>]<Min_Slave_Intervall> | Modular_Station[<WS>] =[<WS>]<Modular_Station> | Max_Module[<WS>] =[<WS>]<Max_Module> | Max_Input_Len[<WS>] =[<WS>]<Max_Input_Len> | Max_Output_Len[<WS>] =[<WS>]<Max_Output_Len> | Max_Data_Len[<WS>] =[<WS>]<Max_Data_Len> | Fail_Safe[<WS>] =[<WS>]<Fail_Safe> | Fail_Safe_required[<WS>] = [<WS>]<Fail_Safe_required> | Diag_Update_Delay[<WS>] = [<WS>]<Diag_Update_Delay> | Max_Diag_Data_Len[<WS>] = [<WS>]<Max_Diag_Data_Len> | Modul_Offset[<WS>] =[<WS>]<Modul_Offset> | Max_User_Prm_Data_Len[<WS>] = [<WS>]<Max_User_Prm_Data_Len> | Slave_Family[<WS>] =[<WS>]<Family_Name> | Prm_Block_Structure_supp[<WS>] = [<WS>]<Prm_Block_Structure_supp> | Prm_Block_Structure_req[<WS>] = [<WS>]<Prm_Block_Structure_req> | Jokerblock_supp[<WS>] =[<WS>]<Jokerblock_supp> | [<Jokerblock-Def>] | PrmCmd_supp[<WS>] =[<WS>]<PrmCmd_supp> | Max_Switch_Over_Time[<WS>] = [<WS>]<Max_Switch_Over_Time> | Slave_Redundancy_supp[<WS>]= [<WS>]<Slave_Redundancy_supp> | Ident_Maintenance_supp[<WS>] =[<WS>]<Ident_Maintenance_supp> | Time_Sync_supp[<WS>] =[<WS>]<Time_Sync_supp> | DPV1_Slave[<WS>] =[<WS>]<DPV1_Slave> | C1_Read_Write_supp[<WS>]= [<WS>]<C1_Read_Write_supp> | C2_Read_Write_supp[<WS>]= [<WS>]<C2_Read_Write_supp> | C1_Max_Data_Len[<WS>] = [<WS>]<Max_C1_Data_Len> | C2_Max_Data_Len[<WS>] = [<WS>]<Max_C2_Data_Len> | C1_Response_Timeout[<WS>] = [<WS>]<C1_Response_Timeout> | C2_Response_Timeout[<WS>] = [<WS>]<C2_Response_Timeout> | C1_Read_Write_required[<WS>] = [<WS>]<C1_Read_Write_required> | C2_Read_Write_required[<WS>] = [<WS>]<C2_Read_Write_required> | C2_Max_Count_Channels[<WS>] = [<WS>]<Max_Count_C2_Channels> | Max_Initiate_PDU_Length[<WS>] = [<WS>]<Max_Initiate_PDU_Length> | Diagnostic_Alarm_supp[<WS>] = [<WS>]<Diagnostic_Alarm_supp> | Process_Alarm_supp[<WS>] = [<WS>]<Process_Alarm_supp> | Pull_Plug_Alarm_supp[<WS>] = [<WS>]<Pull_Plug_Alarm_supp> | Status_Alarm_supp[<WS>] = [<WS>]<Status_Alarm_supp> | Update_Alarm_supp[<WS>] = [<WS>]<Update_Alarm_supp> | Manufacturer_Specific_Alarm_supp[<WS>] = [<WS>]<Manufacturer_Specific_Alarm_supp> | Extra_Alarm_SAP_supp[<WS>] = [<WS>]<Extra_Alarm_SAP_supp> | Alarm_Sequence_Mode_Count[<WS>] = [<WS>]<Alarm_Sequence_Mode_Count> | Alarm_Type_Mode_supp[<WS>] = [<WS>]<Alarm_Type_Mode_supp> | <Alarm_Requirement> | DPV1_Data_Types[<WS>] = [<WS>]<DPV1_Data_Types> | WD_Base_1ms_supp[<WS>] = [<WS>]<WD_Base_1ms_supp> | Check_Cfg_Mode[<WS>] =[<WS>]<Check_Cfg_Mode> | <Unit_Diag_Bit[<WS>](<Bit>)[<WS>] = [<WS>]<Diag_Text> {0<=Bit<=495} | Unit_Diag_Not_Bit[<WS>](<Bit>)[<WS>] = [<WS>]<Diag_Text> {0<=Bit<=495} | Unit_Diag_Bit_Help[<WS>](<Bit>)[<WS>] = [<WS>]<Help_Text> | Unit_Diag_Not_Bit_Help[<WS>](<Bit>)[<WS>] = [<WS>]<Help_Text> | Unit-Diag-Area-Def> | <Channel-Diag-Definition> | <Ext-User-Prm-Data-Const> | <Ext-User-Prm-Data-Ref> | <X-Prm-List> | <User-Definition>
15)	<Ext_Module_Prm_Len> = <Unsigned8>
14)	<F-Ext-Module-Prm-Data-Len> = F_Ext_Module_Prm_Data_Len[<WS>] = [<WS>]<Ext_Module_Prm_Len><LineEnd>
13)	<X-Ext-Module-Prm-Data-Len> = X_Ext_Module_Prm_Data_Len[<WS>] = [<WS>]<Ext_Module_Prm_Len><LineEnd>
12)	<Ext-Module-Prm-Data-Len> = Ext_Module_Prm_Data_Len[<WS>] = [<WS>]<Ext_Module_Prm_Len><LineEnd>

Table B.4 — Formal Description of GSD format *(continued)*

S#	Formal description
11)	<Ext-User-Prm-Data-Def> = ExtUserPrmData[<WS>] =[<WS>]<Reference_Number><WS> <Ext_User_Prm_Data_Name> | <"[Slot_Number]"><LineEnd> <Data_Type_Name><WS><Default_Value> [<WS><Min_Value>[<WS>]-[<WS>]<Max_Value> | <WS><Allowed_Values>]<LineEnd> [<Prm-Text-Ref>] [<Changeable>] [<Visible>] EndExtUserPrmData
10)	<Text_Item> = Text(<Prm_Data_Value>)[<WS>] = [<WS>]<Text><LineEnd>
9)	<Text_List> = <Text_Item> | <Text_List><Text_Item>
8)	<Prm-Text-Def> = PrmText[<WS>] =[<WS>]<Reference_Number><LineEnd> <Text_List> EndPrmText
7)	<Module-Def-Item> = <Info_Text> | <Channel-Diag-Definition> | <Ext-User-Prm-Data-Const> | <Ext-User-Prm-Data-Ref> | <X_Ext_User_Prm_Data_Const> | <X_Ext_User_Prm_Data_Ref> | <F_Ext_User_Prm_Data_Const> | <F_Ext_User_Prm_Data_Ref> | <Alarm_Requirement> | <Ext-Module-Prm-Data-Len> | <X-Ext-Module-Prm-Data-Len> | <F-Param-List> | <F-Ext-Module-Prm-Data-Len> | [<Data-Area-Def>] | Ident_Maintenance_supp[<WS>]= [<WS>]<Ident_Maintenance_supp> | <Subsys-Module-Dir-Index-Def> | <User-Definition>
6)	<Module-Def-List> = <Module-Def-Item> | <Module-Def-List><Module-Def-Item>
5)	<Module-Definition> = Module[<WS>] =[<WS>]<Mod_Name><WS><Config><LineEnd> <Module-Reference> [<Module-Def-List>] EndModule
4)	<GSD-Item> = [<Prm-Text-Def>] | [<Ext-User-Prm-Data-Def>] | [<X-Prm-List>] | <Unit-Def-Item> | <Module-Definition> | [<Slot-Def>] | [<Ph-Interface-Def>] | [<Subsys-Dir-Index-Def>] | [<Isochron-Mode-List>] | [<DXB-List>] | [Unit-Diag-Type-Def]
3)	<GSD-Line> = <LineStart><GSD-Item><LineEnd>
2)	<GSD-List> = <GSD-Line>|<GSD-List><GSD-Line>
1)	<GSD> = [<Any-Text>] <LineStart>#Profibus_DP<LineEnd> <GSD-List> [<LineStart>#<Keyword><LineEnd> [<Any-Text>]]

Annex C
(normative)

P-NET profile templates

C.1 Device profile template description

The device profile XML files shall comply with the device profile XML schema as specified below.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="PNETDeviceProfile" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="PNETDeviceProfile" elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0">
   <xsd:element name="ISO15745Profile">
      <xsd:annotation>
         <xsd:documentation>P-NET Device Profile Template</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="ProfileClassID_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AIP"/>
         <xsd:enumeration value="Process"/>
         <xsd:enumeration value="InformationExchange"/>
         <xsd:enumeration value="Resource"/>
         <xsd:enumeration value="Device"/>
         <xsd:enumeration value="CommunicationNetwork"/>
         <xsd:enumeration value="Equipment"/>
         <xsd:enumeration value="Human"/>
         <xsd:enumeration value="Material"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="ISO15745Reference_DataType">
      <xsd:sequence>
         <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
         <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
         <xsd:element name="ProfileTechnology" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="IASInterface_DataType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="CSI"/>
               <xsd:enumeration value="HCI"/>
               <xsd:enumeration value="ISI"/>
               <xsd:enumeration value="API"/>
               <xsd:enumeration value="CMI"/>
               <xsd:enumeration value="ESI"/>
               <xsd:enumeration value="FSI"/>
               <xsd:enumeration value="MTI"/>
               <xsd:enumeration value="SEI"/>
               <xsd:enumeration value="USI"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:length value="4"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
   <xsd:complexType name="SWNoType">
```

```
      <xsd:sequence>
         <xsd:element name="Name" type="xsd:string"/>
         <xsd:element name="TypeName" type="xsd:string"/>
         <xsd:element name="TypeElement">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="Basictype"/>
                  <xsd:enumeration value="Recordtype"/>
                  <xsd:enumeration value="Enumerated"/>
                  <xsd:enumeration value="Arraytype"/>
                  <xsd:enumeration value="Bitarraytype"/>
                  <xsd:enumeration value="Buffertype"/>
                  <xsd:enumeration value="Settype"/>
                  <xsd:enumeration value="String"/>
                  <xsd:enumeration value="Bitmaptype"/>
                  <xsd:enumeration value="Pointertype"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:element>
         <xsd:element name="ReadAccess" type="xsd:string"/>
         <xsd:element name="WriteAccess" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="DeviceIdentity"/>
            <xsd:element ref="DeviceManager"/>
            <xsd:element ref="DeviceFunction" maxOccurs="unbounded"/>
            <xsd:element ref="ApplicationProcess" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ApplicationProcess">
      <xsd:complexType>
         <xsd:sequence minOccurs="0" maxOccurs="unbounded">
            <xsd:element ref="Channel" maxOccurs="unbounded"/>
            <xsd:element ref="SWNo" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Channel">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="SWNo" maxOccurs="11"/>
            <xsd:element name="PrimaryValue" type="SWNoType"/>
            <xsd:element name="ChConfig" type="SWNoType"/>
            <xsd:element name="Maintenance" type="SWNoType"/>
            <xsd:element name="ChType" type="SWNoType"/>
            <xsd:element name="ChError" type="SWNoType"/>
         </xsd:sequence>
         <xsd:attribute name="Name" type="xsd:string"/>
         <xsd:attribute name="TypeName" type="xsd:string"/>
         <xsd:attribute name="ReadAccess" type="xsd:string"/>
         <xsd:attribute name="WriteAccess" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="SWNo" type="SWNoType"/>
   <xsd:element name="DeviceFunction">
      <xsd:complexType/>
```

```
   </xsd:element>
   <xsd:element name="DeviceIdentity">
      <xsd:complexType>
         <xsd:attribute name="DeviceNumber" type="xsd:positiveInteger" use="required"/>
         <xsd:attribute name="ProgramVersion" type="xsd:string" use="required"/>
         <xsd:attribute name="ManufacturerNo" type="xsd:string" use="required"/>
         <xsd:attribute name="Manufacturer" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DeviceManager">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ServiceChannel"/>
         </xsd:sequence>
         <xsd:attribute name="Revision" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ServiceChannel">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="NumberOfSWNo">
               <xsd:complexType>
                  <xsd:complexContent>
                     <xsd:extension base="SWNoType">
                        <xsd:attribute name="GUID" type="xsd:string"/>
                     </xsd:extension>
                  </xsd:complexContent>
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="Reset" type="SWNoType"/>
            <xsd:element name="WriteEnable" type="SWNoType"/>
            <xsd:element name="ChType" type="SWNoType"/>
            <xsd:element name="CommonError" type="SWNoType"/>
            <xsd:element ref="SWNo"/>
         </xsd:sequence>
         <xsd:attribute/>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

C.2 Communication network profile template description

The communication network profile XML files shall comply with the communication network profile XML schema as specified below.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="PNETCommNetworkProfile" xmlns="PNETCommNetworkProfile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
   <xsd:element name="ISO15745Profile">
      <xsd:annotation>
         <xsd:documentation>P-NET Communication Network Profile Template</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ApplicationLayers"/>
            <xsd:element ref="TransportLayers"/>
            <xsd:element ref="NetworkManagement" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
```

```
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="APDU">
      <xsd:complexType>
         <xsd:attribute name="InternalVariableField" type="xsd:string"/>
         <xsd:attribute name="ExternalVariableField" type="xsd:string"/>
         <xsd:attribute name="DataLength" type="xsd:string"/>
         <xsd:attribute name="OperationCode"/>
         <xsd:attribute name="StatusReturnAddress" type="xsd:string"/>
         <xsd:attribute name="UnacceptableError"/>
         <xsd:attribute name="ResponseDelayTimer" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ApplicationLayers">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="APDU"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DatalinkLayer">
      <xsd:complexType>
         <xsd:attribute name="ErrorDetection" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="IASInterface_DataType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="CSI"/>
               <xsd:enumeration value="HCI"/>
               <xsd:enumeration value="ISI"/>
               <xsd:enumeration value="API"/>
               <xsd:enumeration value="CMI"/>
               <xsd:enumeration value="ESI"/>
               <xsd:enumeration value="FSI"/>
               <xsd:enumeration value="MTI"/>
               <xsd:enumeration value="SEI"/>
               <xsd:enumeration value="USI"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:length value="4"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
   <xsd:complexType name="ISO15745Reference_DataType">
      <xsd:sequence>
         <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
         <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
         <xsd:element name="ProfileTechnology" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="Master">
      <xsd:complexType>
         <xsd:attribute name="NodeAddress" type="xsd:string" use="required"/>
         <xsd:attribute name="ErrorDetection" type="xsd:string" use="required"/>
         <xsd:attribute name="NoOfMasters" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="NetworkManagement">
      <xsd:complexType>
         <xsd:choice minOccurs="0" maxOccurs="unbounded">
            <xsd:element ref="Master"/>
            <xsd:element ref="SimpleNode"/>
```

```
            <xsd:element ref="Slave"/>
         </xsd:choice>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="PhysicalLayer">
      <xsd:complexType>
         <xsd:attribute name="ElectricalStandard" use="required">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="RS232"/>
                  <xsd:enumeration value="RS485"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
         <xsd:attribute name="Baudrate" use="required">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="1200"/>
                  <xsd:enumeration value="2400"/>
                  <xsd:enumeration value="4800"/>
                  <xsd:enumeration value="9600"/>
                  <xsd:enumeration value="19200"/>
                  <xsd:enumeration value="38400"/>
                  <xsd:enumeration value="76800"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
         <xsd:attribute name="PortNo" type="xsd:string" use="required"/>
         <xsd:attribute name="Name" type="xsd:string" use="required"/>
         <xsd:attribute name="SWNo" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="ProfileClassID_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AIP"/>
         <xsd:enumeration value="Process"/>
         <xsd:enumeration value="InformationExchange"/>
         <xsd:enumeration value="Resource"/>
         <xsd:enumeration value="Device"/>
         <xsd:enumeration value="CommunicationNetwork"/>
         <xsd:enumeration value="Equipment"/>
         <xsd:enumeration value="Human"/>
         <xsd:enumeration value="Material"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="SimpleNode">
      <xsd:complexType>
         <xsd:attribute name="NodeAddress" type="xsd:string" use="required"/>
         <xsd:attribute name="ErrorDetection" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Slave">
      <xsd:complexType>
         <xsd:attribute name="NodeAddress" type="xsd:string" use="required"/>
         <xsd:attribute name="ErrorDetection" type="xsd:string" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransportLayers">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="PhysicalLayer"/>
            <xsd:element ref="DatalinkLayer"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

Annex D
(normative)

WorldFIP profile templates

D.1 Device profile template description

D.1.1 Overview

The device profile template XML schema defined in D.1.3 contains the mapping of the device profile class diagrams shown in 6.4.1. Besides the mapped classes and attributes, it contains additional elements, with or without XML attributes, to facilitate unambiguous device profile and device descriptions in XML.

Four classes of conformity are defined, with two subclasses for class1, according to the devices capabilities. Table D.1 gives the main characteristics of these classes, specifying which components are relevant for each of them.

Table D.1 — Main characteristics of devices conformity classes

DeviceConformityClass	Application Management	Network Management	Application
1.1 1.2	MPS	SM_MPS Minimal	MPS with at least 2 Applications Variables : one input, one output
2	MPS + Micro-MMS V1	SM_MPS	MPS + Micro-MMS V1
3	MPS + Micro-MMS V2	SMS + SM_MPS Minimal	MPS + Micro-MMS V2
4	MPS + SubMMS	SMS + SM_MPS Minimal	MPS + SubMMS

D.1.2 DeviceConformityClass

D.1.2.1 Device conformity class 1, plug and play devices

These classes have been defined to identify devices which handle minimal configuration capabilities, and start on reception of a simple order.

Conformity class 1.1 allows to handle a content length of 2 bytes for all application variables.

Conformity class 1.2 allows to handle up to 120 bytes for application variables.

Conformity class 1 devices shall use communication network conformity class 1.

PhysicalNode

The device pertaining to these classes shall have a PhysicalNode using SM_MPS with:

— A Presence Variable whose WorldFIP DLL Identifier value is 14XXh (where XXh is the device address)

— A Simple Identification Variable whose WorldFIP Identifier value is 10XXh (where XXh is the device address)

LogicalNode

The device pertaining to these classes shall have one LogicalNode whose only state in **IN SERVICE**.

FunctionBlock

The device pertaining to these classes shall have one FunctionBlock with:

— Two network visible states : **IN SERVICE**, **NOT IN SERVICE**

— One consumed control/configuration variable whose WorldFIP DLL Identifier value is 03XXh (where XXh is the device address)

— One consumed application variable whose WorldFIP DLL Identifier value is 05XXh (where XXh is the device address)

— One produced application variable whose WorldFIP DLL Identifier value is 06XXh (where XXh is the device address)

ExchangeBlock

The device pertaining to these classes shall have no ExchangeBlock.

D.1.2.2 Device conformity class 2, configurable and controllable simple devices

This class shall identify devices which handle few informations for configuration and setting capabilities. The essential information exchange is of cyclic nature. The device may handle events.

Conformity Class 2 devices shall use communication network conformity class 2.

PhysicalNode

The device pertaining to this class shall have a PhysicalNode using SM_MPS with:

— A Presence Variable whose WorldFIP DLL Identifier value is 14XXh (where XXh is the device address)

— An Identification Variable whose WorldFIP Identifier value is 10XXh (where XXh is the device address)

— A Report Variable whose WorldFIP DLL Identifier value is 11XXh (where XXh is the device address)

— A Global Control Variable whose WorldFIP DLL Identifier value is 12XXh (where XXh is the device address)

— A Configuration Unloading Variable whose WorldFIP DLL Identifier value is 21XXh (where XXh is the device address)

— Other optional SM_MPS variables

LogicalNode

The device pertaining to this class shall have one to eight LogicalNode's. The LogicalNode network visible states are **EMPTY**, **COM-EMPTY** and **READY**. These states are sub-states of the interoperability general rules logical node states **EXISTENT**, **UNCOMPLETE ASSIGNMENT** and **COMPLETE ASSIGNMENT** respectively. The **NOT IN SERVICE**, **IN SERVICE**, **CONFIGURATION OK** and **CONFIGURATION NOK** states are not used.

FunctionBlock

The device pertaining to this class shall have one or more FunctionBlock's. Each FunctionBlock may have:

— All interface variables are accessible using MPS or Micro-MMS using the index

— One consumed application process control variable whose WorldFIP DLL Identifier value is 05XXh (where XXh is the device address)

— One produced application process status variable whose WorldFIP DLL Identifier value is 06XXh (where XXh is the device address)

— Six application management variables and n additional application variables

The network visible function block configuration states are **EMPTY**, **WAITING**, **READY** and **INVALID**. These states are sub-states of the interoperability general function block states **EXISTENT**, **COMPLETE ASSIGNMENT**, **CONFIGURATION OK** and **CONFIGURATION NOK** respectively. The **NOT IN SERVICE** and **IN SERVICE** states are not used for configuration modes description.

The network visible function block states are **IDLE**, **STOPPED** and **RUNNING**. These states are substates of the interoperability general rules function block **IN SERVICE** state.

ExchangeBlock

ExchangeBlock's are pre-existing, and as a consequence the characteristics of communication and interface objects are pre-determined and cannot be changed.

D.1.2.3 Device conformity class 3, configurable and controllable devices

This class shall identify devices which handle lots of information for configuration and setting purposes. Event handling is essential.

Conformity Class 3 devices shall use communication network conformity class 3.

PhysicalNode

The device pertaining to this class shall have a PhysicalNode using SM_MPS with:

— A Presence Variable whose WorldFIP DLL Identifier value is 14XXh (where XXh is the device address)

— An Identification Variable whose WorldFIP Identifier value is 10XXh (where XXh is the device address)

— A Report Variable whose WorldFIP DLL Identifier value is 11XXh (where XXh is the device address)

LogicalNode

The device pertaining to this class shall have one or more pre-existing first level LogicalNode's. The configuration by the network of a first level LogicalNode is total and includes the configurations of the LogicalNode itself and encapsulated entities (being assigned or created and assigned by the configuration). The configuration is downloaded using message services. The entities encapsulated (LogicalNode, FunctionBlock's and so on) by a first level LogicalNode are under the dependence of the first level LogicalNode (top of the hierarchy). The first level LogicalNode operating mode management, if any, is using message services or MPS services.

The LogicalNode network visible states shall be **IDLE**, **READY**, **RUNNING-RUNTIME**, **RUNNING-DEBUGGING**, **UNRUNNABLE**. The states **READY**, **RUNNING-RUNTIME**, **RUNNING-DEBUGGING** are **IN SERVICE** sub-states. The state **UNRUNNABLE** shall be an **EXISTENT** sub-state. The state **IDLE** shall be a **CONFIGURATION OK** substate or an **UNCOMPLETE ASSIGNMENT** substate.

D.1.2.4 Device conformity class 4, complex devices

This class has been defined to identify devices with total configuration functionality.

D.1.3 Device profile template XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema     targetNamespace="http://www.worldfip.org"
                xmlns="http://www.worldfip.org"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:xlink="http://www.w3.org/1999/xlink"
                elementFormDefault="qualified"
                attributeFormDefault="unqualified"
                version="1.0">
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:simpleType name="ProfileClassID_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AIP"/>
         <xsd:enumeration value="Process"/>
         <xsd:enumeration value="InformationExchange"/>
         <xsd:enumeration value="Resource"/>
         <xsd:enumeration value="Device"/>
         <xsd:enumeration value="CommunicationNetwork"/>
         <xsd:enumeration value="Equipment"/>
         <xsd:enumeration value="Human"/>
         <xsd:enumeration value="Material"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="ISO15745Reference_DataType">
      <xsd:sequence>
         <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
         <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
         <xsd:element name="ProfileTechnology" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="IASInterface_DataType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="CSI"/>
               <xsd:enumeration value="HCI"/>
               <xsd:enumeration value="ISI"/>
               <xsd:enumeration value="API"/>
               <xsd:enumeration value="CMI"/>
               <xsd:enumeration value="ESI"/>
```

```
                <xsd:enumeration value="FSI"/>
                <xsd:enumeration value="MTI"/>
                <xsd:enumeration value="SEI"/>
                <xsd:enumeration value="USI"/>
             </xsd:restriction>
          </xsd:simpleType>
          <xsd:simpleType>
             <xsd:restriction base="xsd:string">
                <xsd:length value="4"/>
             </xsd:restriction>
          </xsd:simpleType>
       </xsd:union>
    </xsd:simpleType>
    <xsd:annotation>
       <xsd:documentation>* ISO 15745 DEFINED DATA TYPES *</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType name="ProfileHandle_DataType">
       <xsd:sequence>
          <xsd:element name="ProfileIdentification" type="xsd:string"/>
          <xsd:element name="ProfileRevision" type="xsd:string"/>
          <xsd:element name="ProfileLocation" type="xsd:anyURI" minOccurs="0"/>
       </xsd:sequence>
    </xsd:complexType>
    <xsd:annotation>
       <xsd:documentation>* BODY SECTION *</xsd:documentation>
    </xsd:annotation>
    <!-- All Profile Templates compliant with Part 3 shall include basic profile object template
information above -->
    <!-- The following information is developed from the UML models -->
    <xsd:element name="ProfileBody">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="DeviceIdentity" minOccurs="0"/>
             <xsd:element ref="DeviceManager" minOccurs="0"/>
             <xsd:element ref="DeviceFunction" maxOccurs="unbounded"/>
             <xsd:element ref="ApplicationProcess" minOccurs="0" maxOccurs="unbounded"/>
             <xsd:element name="ExternalProfileHandle" type="ProfileHandle_DataType" minOccurs="0"
maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeviceIdentity">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element name="DeviceVendor" type="xsd:string"/>
             <xsd:element name="DeviceProductType" type="xsd:string"/>
             <xsd:element name="DeviceConformityClass" type="DeviceClass_DataType"/>
             <xsd:element name="DeviceProductName" type="xsd:string"/>
             <xsd:element name="DeviceProductCode" type="xsd:string"/>
             <xsd:element name="DeviceRevision" type="xsd:string"/>
             <xsd:element name="DeviceSerialNumber" type="xsd:string"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:simpleType name="DeviceClass_DataType">
       <xsd:restriction base="xsd:string">
          <xsd:enumeration value="1.1"/>
          <xsd:enumeration value="1.2"/>
          <xsd:enumeration value="2"/>
          <xsd:enumeration value="3"/>
          <xsd:enumeration value="4"/>
       </xsd:restriction>
    </xsd:simpleType>
    <xsd:element name="DeviceManager">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="PhysicalNode" maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="PhysicalNode">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element name="Identifier" type="xsd:string"/>
             <xsd:element name="ManagementVariable" type="DeviceVariable_DataType" maxOccurs="unbounded"/>
             <xsd:element ref="LogicalNode" maxOccurs="unbounded"/>
```

```
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="DeviceVariable_DataType">
      <xsd:sequence>
         <xsd:element name="DLIdentifier" type="xsd:hexBinary"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="LogicalNode">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Identifier" type="xsd:string"/>
            <xsd:element name="VisibleStates" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="FunctionBlock" type="FunctionBlock_DataType" maxOccurs="unbounded"/>
            <xsd:element ref="LogicalNode" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="ExchangeBlock" type="FunctionBlock_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="FunctionBlock_DataType">
      <xsd:sequence>
         <xsd:element name="Identifier" type="xsd:string"/>
         <xsd:element name="VisibleStates" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
         <xsd:element name="FunctionBlock" type="FunctionBlock_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         <xsd:element name="ApplicationVariable" type="DeviceVariable_DataType" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="DeviceFunction">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="DeviceDataSheet" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ApplicationProcess">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="FunctionBlockIdentifier" type="xsd:string" maxOccurs="unbounded"/>
            <xsd:element name="ExchangeBlockIdentifier" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

D.2 Communication network profile template description

D.2.1 Overview

The communication network profile template XML schema defined in D.2.5 contains the mapping of the communication network profile class diagrams shown in 6.4.2. Besides the mapped classes and attributes, it contains additional elements, with or without XML attributes, to facilitate unambiguous communication network profile descriptions in XML.

The three communication network conformity classes are used to derive the MPS, MMS and Data Link Layer conformity classes, and the presence of the MCS component.

D.2.2 Application layers

D.2.2.1 MPS communication profiles

D.2.2.1.1 TypeConstructor

PRIMITIVE TYPES, used for **SIMPLE**, shall be: Boolean, Bit string, Integer, Unsigned, Octet string, Visible string, Generalised time, Floating point, Binary time, BCD.

PREDEFINED construction types are mostly used for description variables and refer to types that are reserved by the standard.

EXPLICIT construction types allow the associated variables to be provided with an explicit transfer syntax, in order to carry the type semantics with the data.

PREDEFINED and **EXPLICIT** construction types cannot be used to compose types of **ARRAY** and **STRUCTURE**.

D.2.2.1.2 MPSConformityClass

Table D.2 presents the WorldFIP MPS conformity classes. Class 3 is divided in two subclasses 3.1 and 3.2. The MPS conformity class number shall be derived from the number in the CommunicationNetworkConformityClass attribute.

Table D.2 — WorldFIP MPS conformity classes

MPSConformityClass	Service	Constraint
1	A_Writeloc	Mandatory
	A_Readloc	Mandatory
	A_Update	Not supported
	A_Writefar	Not supported
	A_Readfar	Not supported
	A_Write	Not supported
	A_Read	Not supported
	A_Sent	Optional
	A_Received	Optional
	Asynchronous Refreshness	Mandatory
	Asynchronous Promptness	Mandatory
	Transmitted Variable Significance	Mandatory
	Status Byte Transmission	Mandatory
2	A_Writeloc	Mandatory
	A_Readloc	Mandatory
	A_Update	Optional
	A_Writefar	Optional
	A_Readfar	Optional
	A_Write	Not supported
	A_Read	Not supported
	A_Sent	Optional
	A_Received	Optional
	Asynchronous Refreshness	Conditional
	Asynchronous Promptness	Conditional
	Transmitted Variable Significance	Not supported
	Status Byte Transmission	Mandatory

Table D.2 — WorldFIP MPS conformity classes (continued)

MPSConformityClass	Service	Constraint
3.1	A_Writeloc	Mandatory
	A_Readloc	Mandatory
	A_Update	Not supported
	A_Writefar	Not supported
	A_Readfar	Not supported
	A_Write	Optional
	A_Read	Optional
	A_Sent	Optional
	A_Received	Optional
	Asynchronous Refreshness	Mandatory
	Asynchronous Promptness	Mandatory
	Transmitted Variable Significance	Mandatory
	Status Byte Transmission	Mandatory
3.2	A_Writeloc	Mandatory
	A_Readloc	Mandatory
	A_Update	Optional
	A_Writefar	Optional
	A_Readfar	Optional
	A_Write	Optional
	A_Read	Optional
	A_Sent	Optional
	A_Received	Optional
	Asynchronous Refreshness	Mandatory
	Asynchronous Promptness	Mandatory
	Transmitted Variable Significance	Mandatory
	Status Byte Transmission	Mandatory

D.2.2.2 MCS communication profiles

The WorldFIP MCS communication network profile definition is meaningful only for WorldFIP communication network conformity class 3.

D.2.2.3 MMS communication profiles

D.2.2.3.1 Micro-MMS definitions

Two subsets for MMS services shall be defined.

Micro-MMS V1 is the name given to the minimum conformance class of SubMMS. It shall include 3 variable object services: Read, Write and Information Report.

Table D.3 shows the WorldFIP Micro-MMS V1 variable object services.

Table D.3 — WorldFIP Micro-MMS V1 variable object services

Service	Constraint
Read	Mandatory
Write	Mandatory
Information Report	Mandatory
Maximum PDU length	256 bytes

Table D.4 shows the WorldFIP Micro-MMS V2 object services.

Table D.4 — WorldFIP Micro-MMS V2 object subset definition

Service	Constraint
Association Management	Optional
Variable	Mandatory
Variable List	Optional
Program Invocation	Optional
Domain	Optional

D.2.2.3.2 MMS implementations

Table D.5 shows the WorldFIP MMS implementations for the three WorldFIP communication network conformity classes.

Table D.5 — WorldFIP MMS communication network profiles definition

Conformity Class	MMS Services
1	No MMS implementation
2	Micro-MMS V1
3	Micro-MMS V2

D.2.3 Transport layers; DLConformityClass

Table D.6 presents the WorldFIP Data Link Layer conformity classes, as documented in the DLConformityClass attribute.

Table D.6 — WorldFIP Data Link Layer conformity classes

DLConformityClass	Service	Constraint
1	Buffer Transfer	Mandatory
	Buffer Write	Mandatory
	Buffer Read	Mandatory
	Explicit Request of Buffer Transfer	Not supported
	Message Transfer	Not supported
2	Buffer Transfer	Mandatory
	Buffer Write	Mandatory
	Buffer Read	Mandatory
	Explicit Request of Buffer Transfer	Optional
	Message Transfer	Mandatory
3	Buffer Transfer	Mandatory
	Buffer Write	Mandatory
	Buffer Read	Mandatory
	Explicit Request specified of Buffer Transfer	Not supported
	Explicit Request free of Buffer Transfer	Optional
	Message Transfer SDA	Optional
	Message Transfer SDN	Optional
	Aperiodic Message	Optional
	Periodic Message	Optional

D.2.4 Network Management

D.2.4.1 Network Management services

Table D.7, Table D.8 and Table D.9 present the WorldFIP SM_MPS minimal services.

Table D.7 — WorldFIP SM_MPS agent device services

Service	Constraint
Identification	Optional
Address assignment	Optional
Tag Name assignment	Optional
Presence	Mandatory
Remote Loading	Optional
Remote Reading	Conditional
Command	Optional
Check	Conditional
Report	Optional

Table D.8 — WorldFIP SM_MPS system services

Service	Constraint
Presence Check	Optional
Bus Arbitrator Synchro	Optional
Segment Parameter	Optional
Absolute Time	Optional
100 µs Relative Time	Optional
Counter Global Mgnt	Optional

Table D.9 — WorldFIP SM_MPS minimal subset

Service	Constraint
Presence Check	Optional
Presence	Mandatory
Segment Parameter	Optional
Identification	Optional
Report	Optional

D.2.4.2 NMConformityClass

Table D.10 defines the WorldFIP network management conformity classes. The network conformity class number shall be derived from the number in the DeviceConformityClass attribute.

Table D.10 — WorldFIP network management conformity classes

NMConformityClass	SM_MPS Services
1	SM_MPS Minimal
2	SM_MPS
3	SMS + SM_MPS Minimal
4	SMS + SM_MPS Minimal

D.2.5 Communication network profile template XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema     targetNamespace="http://www.worldfip.org"
                xmlns="http://www.worldfip.org"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:xlink="http://www.w3.org/1999/xlink"
                elementFormDefault="qualified"
                attributeFormDefault="unqualified"
                version="1.0">
   <xsd:element name="ISO15745Profile">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ProfileHeader"/>
            <xsd:element ref="ProfileBody"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER SECTION *</xsd:documentation>
   </xsd:annotation>
   <xsd:element name="ProfileHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProfileIdentification" type="xsd:string"/>
            <xsd:element name="ProfileRevision" type="xsd:string"/>
            <xsd:element name="ProfileName" type="xsd:string"/>
            <xsd:element name="ProfileSource" type="xsd:string"/>
            <xsd:element name="ProfileClassID" type="ProfileClassID_DataType"/>
            <xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
            <xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
            <xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
            <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:annotation>
      <xsd:documentation>* HEADER DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:simpleType name="ProfileClassID_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="AIP"/>
         <xsd:enumeration value="Process"/>
         <xsd:enumeration value="InformationExchange"/>
         <xsd:enumeration value="Resource"/>
         <xsd:enumeration value="Device"/>
         <xsd:enumeration value="CommunicationNetwork"/>
         <xsd:enumeration value="Equipment"/>
         <xsd:enumeration value="Human"/>
         <xsd:enumeration value="Material"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:complexType name="ISO15745Reference_DataType">
      <xsd:sequence>
         <xsd:element name="ISO15745Part" type="xsd:positiveInteger"/>
         <xsd:element name="ISO15745Edition" type="xsd:positiveInteger"/>
         <xsd:element name="ProfileTechnology" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:simpleType name="IASInterface_DataType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="CSI"/>
               <xsd:enumeration value="HCI"/>
               <xsd:enumeration value="ISI"/>
               <xsd:enumeration value="API"/>
               <xsd:enumeration value="CMI"/>
               <xsd:enumeration value="ESI"/>
               <xsd:enumeration value="FSI"/>
               <xsd:enumeration value="MTI"/>
               <xsd:enumeration value="SEI"/>
               <xsd:enumeration value="USI"/>
```

```
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:length value="4"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
   <xsd:annotation>
      <xsd:documentation>* ISO 15745 DEFINED DATA TYPES *</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType name="ProfileHandle_DataType">
      <xsd:sequence>
         <xsd:element name="ProfileIdentification" type="xsd:string"/>
         <xsd:element name="ProfileRevision" type="xsd:string"/>
         <xsd:element name="ProfileLocation" type="xsd:anyURI" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:annotation>
      <xsd:documentation>* BODY SECTION *</xsd:documentation>
   </xsd:annotation>
   <!-- All Profile Templates compliant with Part 3 shall include basic profile object template
information above -->
   <!-- The following information is developed from the UML models -->
   <xsd:element name="ProfileBody">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="ApplicationLayers"/>
            <xsd:element ref="TransportLayers"/>
            <xsd:element ref="NetworkManagement" minOccurs="0"/>
            <xsd:element name="ExternalProfileHandle" type="ProfileHandle_DataType" minOccurs="0"/>
         </xsd:sequence>
         </xsd:attribute name="CommunicationNetworkConformityClass" type="CommNetClass_DataType"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="CommNetClass_DataType">
      <xsd:restriction base="xsd:integer">
         <xsd:minInclusive value="1"/>
         <xsd:maxInclusive value="3"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="ApplicationLayers">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="SubMMS" minOccurs="0"/>
            <xsd:element ref="MCS" minOccurs="0"/>
            <xsd:element ref="MPS"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransportLayers">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="DataLinkLayer"/>
            <xsd:element ref="PhysicalLayer"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="NetworkManagement">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="SMS" minOccurs="0"/>
            <xsd:element ref="SM_MPS"/>
         </xsd:sequence>
         <xsd:attribute name="NMConformityClass" type="NMClass_DataType"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="NMClass_DataType">
      <xsd:restriction base="xsd:integer">
         <xsd:minInclusive value="1"/>
         <xsd:maxInclusive value="4"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="SMS">
      <xsd:complexType>
```

```
      <xsd:sequence>
         <xsd:element ref="SubMMS"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="SM_MPS">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="MPS"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="SubMMS">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="VMDProfile" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="MMSConformityClass" type="MMSClass_DataType"/>
      <xsd:attribute name="Title" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>
<xsd:simpleType name="MMSClass_DataType">
   <xsd:restriction base="xsd:integer">
      <xsd:minInclusive value="1"/>
      <xsd:maxInclusive value="3"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:element name="VMDProfile">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="Capability" type="xsd:string" maxOccurs="unbounded"/>
         <xsd:element ref="ProgramInvocation" maxOccurs="unbounded"/>
         <xsd:element ref="Domain" maxOccurs="unbounded"/>
         <xsd:element ref="VMDVariable" maxOccurs="unbounded"/>
         <xsd:element ref="VMDVariableList" minOccurs="0" maxOccurs="unbounded"/>
         <xsd:element ref="Event" minOccurs="0" maxOccurs="unbounded"/>
         <xsd:element name="AdditionalDetail" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="ExecutiveFunction" type="xsd:string"/>
      <xsd:attribute name="VendorName" type="xsd:string"/>
      <xsd:attribute name="ModelName" type="xsd:string"/>
      <xsd:attribute name="Revision" type="xsd:string"/>
      <xsd:attribute name="LogicalStatus" type="LogicalStatus_DataType"/>
   </xsd:complexType>
</xsd:element>
<xsd:simpleType name="LogicalStatus_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="STATE-CHANGES-ALLOWED"/>
      <xsd:enumeration value="NO-STATE-CHANGES-ALLOWED"/>
      <xsd:enumeration value="OTHER"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="AccessProtection_DataType">
   <xsd:sequence>
      <xsd:element name="Password" type="xsd:string"/>
      <xsd:element name="AccessGroups" type="xsd:unsignedByte"/>
      <xsd:element name="AccessRights" type="xsd:hexBinary"/>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name="ProgramInvocation">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="Name" type="xsd:string"/>
         <xsd:element name="Index" type="xsd:string"/>
         <xsd:element name="DomainReference" type="xsd:string" maxOccurs="unbounded"/>
         <xsd:element name="Deletable" type="xsd:boolean"/>
         <xsd:element name="Reusable" type="xsd:boolean"/>
         <xsd:element name="Argument" type="xsd:string"/>
         <xsd:element name="AccessProtection" type="AccessProtection_DataType" minOccurs="0"/>
         <xsd:element name="Extension" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="Domain">
   <xsd:complexType>
      <xsd:sequence>
```

```
            <xsd:element name="Name" type="xsd:string"/>
            <xsd:element name="Index" type="xsd:string"/>
            <xsd:element name="Capability" type="xsd:string" maxOccurs="unbounded"/>
            <xsd:element name="Predefined" type="xsd:boolean"/>
            <xsd:element name="Sharable" type="xsd:boolean"/>
            <xsd:element name="ProgramInvocationReference" type="xsd:string" maxOccurs="unbounded"/>
            <xsd:element name="AccessProtection" type="AccessProtection_DataType" minOccurs="0"/>
            <xsd:element name="Extension" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Event">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Name" type="xsd:string"/>
            <xsd:element name="Index" type="xsd:string"/>
            <xsd:element name="Type" type="xsd:hexBinary"/>
            <xsd:element name="AccessProtection" type="AccessProtection_DataType" minOccurs="0"/>
            <xsd:element name="Extension" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="VMDVariable">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Name" type="xsd:string"/>
            <xsd:element name="Index" type="xsd:string"/>
            <xsd:element name="Type" type="xsd:string"/>
            <xsd:element name="AccessProtection" type="AccessProtection_DataType" minOccurs="0"/>
            <xsd:element name="Extension" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="VMDVariableList">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Name" type="xsd:string"/>
            <xsd:element name="Index" type="xsd:string"/>
            <xsd:element name="Deletable" type="xsd:boolean"/>
            <xsd:element name="VMDVariableReference" type="xsd:string" maxOccurs="unbounded"/>
            <xsd:element name="AccessProtection" type="AccessProtection_DataType" minOccurs="0"/>
            <xsd:element name="Extension" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="MCS">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="CommunicationMode" type="CommunicationMode_DataType"/>
            <xsd:element name="AssociationType" type="AssociationType_DataType"/>
            <xsd:element name="EstablishmentDuration" type="xsd:duration"/>
            <xsd:element name="PDUSize" type="xsd:positiveInteger"/>
            <xsd:element name="TransferRate" type="xsd:positiveInteger"/>
            <xsd:element name="NumberOfRetries" type="xsd:positiveInteger"/>
            <xsd:element name="AnticipationFactor" type="xsd:positiveInteger"/>
            <xsd:element name="SDUSize" type="xsd:positiveInteger"/>
            <xsd:element name="TerminationDuration" type="xsd:positiveInteger"/>
            <xsd:element name="Priority" type="xsd:positiveInteger"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="CommunicationMode_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="ASSOCIATED"/>
         <xsd:enumeration value="NON-ASSOCIATED"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:simpleType name="AssociationType_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="NEGOCIATED"/>
         <xsd:enumeration value="PRENEGOCIATED"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="MPS">
      <xsd:complexType>
         <xsd:sequence>
```

```
            <xsd:element ref="IdentifiedVariable" maxOccurs="unbounded"/>
            <xsd:element ref="ProducedVariable" maxOccurs="unbounded"/>
            <xsd:element ref="ConsumedVariable" maxOccurs="unbounded"/>
            <xsd:element ref="ThirdPartyVariable" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="IdentifiedVariableList" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="TypeConstructor" maxOccurs="unbounded"/>
            <xsd:element ref="VariableAccess" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
         <xsd:attribute name="MPSConformityClass" type="MPSClass_DataType"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="MPSClass_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="1"/>
         <xsd:enumeration value="2"/>
         <xsd:enumeration value="3.1"/>
         <xsd:enumeration value="3.2"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="IdentifiedVariable">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="A_Name" type="xsd:string"/>
            <xsd:element name="TypeConstructorReference" type="xsd:string"/>
            <xsd:element name="TransmittedStatus" type="xsd:boolean"/>
            <xsd:element name="SignificantStatus" type="xsd:hexBinary" minOccurs="0"/>
            <xsd:element name="Identifier" type="xsd:string"/>
            <xsd:element name="TransmissionMode" type="TransmissionMode_DataType"/>
            <xsd:element name="NetworkPeriod" type="xsd:duration" minOccurs="0"/>
            <xsd:element name="Class" type="Class_DataType"/>
            <xsd:element name="ConsistencyVariable" type="xsd:boolean" minOccurs="0"/>
            <xsd:element name="IdentifiedVariableListReference" type="xsd:string" minOccurs="0"/>
            <xsd:element name="UniversalServicesRequested" type="xsd:boolean"/>
            <xsd:element name="UniversalServicesScope" type="Scope_DataType" minOccurs="0"/>
            <xsd:element name="Priority" type="Priority_DataType" minOccurs="0"/>
            <xsd:element name="VariableAccessReference" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TypeConstructor">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="A_Name" type="xsd:string"/>
            <xsd:element name="Construction" type="Construction_DataType"/>
            <xsd:choice>
               <xsd:element ref="Simple"/>
               <xsd:element ref="Array"/>
               <xsd:element ref="StructureField" maxOccurs="unbounded"/>
               <xsd:element name="Predefined" type="xsd:string"/>
               <xsd:element name="Explicit" type="xsd:string"/>
            </xsd:choice>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="Construction_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="SIMPLE"/>
         <xsd:enumeration value="ARRAY"/>
         <xsd:enumeration value="STRUCTURE"/>
         <xsd:enumeration value="PREDEFINED"/>
         <xsd:enumeration value="EXPLICIT"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="Simple">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="PrimitiveType" type="Primitive_DataType"/>
            <xsd:element name="Size" type="xsd:positiveInteger"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="Primitive_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="BOOLEAN"/>
         <xsd:enumeration value="INTEGER"/>
         <xsd:enumeration value="UNSIGNED"/>
```

```
      <xsd:enumeration value="OCTETSTRING"/>
      <xsd:enumeration value="VISIBLESTRING"/>
      <xsd:enumeration value="GENERALISEDTIME"/>
      <xsd:enumeration value="FLOATINGPOINT"/>
      <xsd:enumeration value="BINARYTIME"/>
      <xsd:enumeration value="BCD"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:element name="Array">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="Compressed" type="xsd:boolean"/>
         <xsd:element name="Dimension" type="xsd:positiveInteger"/>
         <xsd:element name="TypeConstructorReference" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="StructureField">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="NamedField" type="xsd:boolean"/>
         <xsd:element name="FieldName" type="xsd:string"/>
         <xsd:element name="TypeConstructorReference" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:simpleType name="TransmissionMode_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="PERIODIC"/>
      <xsd:enumeration value="APERIODIC"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Class_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="NORMAL"/>
      <xsd:enumeration value="SYNCHRONIZATION"/>
      <xsd:enumeration value="DESCRIPTION"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:element name="VariableAccess">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="A_Name" type="xsd:string"/>
         <xsd:element name="AccessMode" type="AccessMode_DataType"/>
         <xsd:element name="AccessPath" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:simpleType name="AccessMode_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="PARTIAL"/>
      <xsd:enumeration value="GLOBAL"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Scope_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="LOCAL"/>
      <xsd:enumeration value="DISTANT"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Priority_DataType">
   <xsd:restriction base="xsd:string">
      <xsd:enumeration value="NORMAL"/>
      <xsd:enumeration value="URGENT"/>
   </xsd:restriction>
</xsd:simpleType>
<xsd:element name="ProducedVariable">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="IdentifiedVariableReference" type="xsd:string"/>
         <xsd:element name="RefreshmentElaborated" type="xsd:boolean"/>
         <xsd:element ref="Refreshment" minOccurs="0"/>
         <xsd:element name="PunctualRefreshmentElaborated" type="xsd:boolean"/>
         <xsd:element ref="PunctualRefreshment" minOccurs="0"/>
         <xsd:element name="ResynchronizedVariable" type="xsd:boolean"/>
         <xsd:element ref="ProductionResynchronization" minOccurs="0"/>
```

```
            <xsd:element name="EmissionIndication" type="xsd:boolean"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Refreshment">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProductionPeriod" type="xsd:duration"/>
            <xsd:element name="TimeOut" type="xsd:duration"/>
            <xsd:element name="RefreshmentCharacteristic" type="Synchronization_DataType"/>
            <xsd:element name="SynchronizationVariableReference" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="Synchronization_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="SYNCHRONOUS"/>
         <xsd:enumeration value="ASYNCHRONOUS"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="PunctualRefreshment">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ProductionTimeSlot" type="xsd:positiveInteger"/>
            <xsd:element name="TimeOut" type="xsd:duration"/>
            <xsd:element name="SynchronizationVariableReference" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ProductionResynchronization">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="PrivateRefreshmentStatus" type="Synchronization_DataType" minOccurs="0"/>
            <xsd:element name="TimeOut" type="xsd:duration" minOccurs="0"/>
            <xsd:element name="PrivatePunctualRefreshmentStatus" type="Synchronization_DataType"
minOccurs="0"/>
            <xsd:element name="SynchronizationVariableReference" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ThirdPartyVariable">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="IdentifiedVariableReference" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ConsumedVariable">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="IdentifiedVariableReference" type="xsd:string"/>
            <xsd:element name="PromptnessElaborated" type="xsd:boolean"/>
            <xsd:element ref="Promptness" minOccurs="0"/>
            <xsd:element name="PunctualPromptnessElaborated" type="xsd:boolean"/>
            <xsd:element ref="PunctualPromptness" minOccurs="0"/>
            <xsd:element name="ResynchronizedVariable" type="xsd:boolean"/>
            <xsd:element ref="ConsumptionResynchronization"/>
            <xsd:element name="ReceptionIndication" type="xsd:boolean"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Promptness">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ConsumptionPeriod" type="xsd:duration"/>
            <xsd:element name="TimeOut" type="xsd:duration"/>
            <xsd:element name="PromptnessCharacteristic" type="Synchronization_DataType"/>
            <xsd:element name="SynchronizationVariableReference" type="xsd:string" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="PunctualPromptness">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="ConsumptionTimeSlot" type="xsd:positiveInteger"/>
            <xsd:element name="TimeOut" type="xsd:duration"/>
```

```
            <xsd:element name="SynchronizationVariableReference" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ConsumptionResynchronization">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="PrivatePromptnessStatus" type="Synchronization_DataType" minOccurs="0"/>
            <xsd:element name="TimeOut" type="xsd:duration" minOccurs="0"/>
            <xsd:element name="PrivatePunctualPromptnessStatus" type="Synchronization_DataType"
minOccurs="0"/>
            <xsd:element name="SynchronizationVariableReference" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="IdentifiedVariableList">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="A_Name" type="xsd:string"/>
            <xsd:element name="ConsumedVariableReference" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element name="SpatialConsistencyRequired" type="xsd:boolean"/>
            <xsd:element name="RecoveryRequired" type="xsd:boolean"/>
            <xsd:element name="SynchronizationVariableReference" type="xsd:string"/>
            <xsd:element name="InconsistencyDetection" type="Inconsistency_DataType" minOccurs="0"/>
            <xsd:element name="ConsistencyVariableReference" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element name="RecoveryPeriod" type="xsd:duration" minOccurs="0"/>
            <xsd:element name="TransmissionConsistencyRequired" type="xsd:boolean"/>
            <xsd:element name="ProductionConsistencyRequired" type="xsd:boolean"/>
            <xsd:element name="PunctualTransmissionConsistencyRequired" type="xsd:boolean"/>
            <xsd:element name="PunctualProductionConsistencyRequired" type="xsd:boolean"/>
            <xsd:element name="ProducedVariableReference" type="xsd:string" minOccurs="0"/>
            <xsd:element name="RecoveryNature" type="Recovery_DataType" minOccurs="0"/>
            <xsd:element name="TimeOut" type="xsd:duration" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="Inconsistency_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="TRANSITORY"/>
         <xsd:enumeration value="PERMANENT"/>
         <xsd:enumeration value="UNDEFINED"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:simpleType name="Recovery_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="FRAGMENTED"/>
         <xsd:enumeration value="INDIVISIBLE"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="DataLinkLayer">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="Message" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="DLVariable" maxOccurs="unbounded"/>
         </xsd:sequence>
         <xsd:attribute name="DLConformityClass" type="DLClass_DataType"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="DLClass_DataType">
      <xsd:restriction base="xsd:integer">
         <xsd:minInclusive value="1"/>
         <xsd:maxInclusive value="3"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="Message">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="SourceAddress" type="MessageAddress_DataType"/>
            <xsd:element name="DestinationAddress" type="MessageAddress_DataType"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="MessageAddress_DataType">
      <xsd:sequence>
```

```
         <xsd:element name="IndividualGroup" type="xsd:boolean"/>
         <xsd:element name="LSAP" type="xsd:byte"/>
         <xsd:element name="Station" type="xsd:byte"/>
         <xsd:element name="SN" type="xsd:boolean"/>
         <xsd:element name="Segment" type="xsd:byte"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="DLVariable">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Identifier" type="xsd:hexBinary"/>
            <xsd:element name="MsgTransferType" type="Transfer_DataType"/>
            <xsd:element name="RqInhibit" type="xsd:boolean"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="Transfer_DataType">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="CYCLIC"/>
         <xsd:enumeration value="APERIODIC"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name="PhysicalLayer">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="FIPLLProfile" type="xsd:string"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

Annex E
(normative)

INTERBUS profile templates

E.1 Device profile template description

E.1.1 Overview

The device profile template XML schemas defined in E.1.6 contain the mapping of the device profile class diagrams shown in 6.5.1. Besides the mapped classes and attributes, it contains additional elements, with or without XML attributes, to facilitate unambiguous device profile and device descriptions in XML. To allow the reuse of certain element definitions and to allow a flexible extension of device profiles and device descriptions by models not anticipated here, the classes have been mapped to more than one XML schema. Table E.1 lists all XML schemas of the INTERBUS device profile template exchange description.

NOTE Describing a device profile or device description may not need the application of all XML schemas defined herein.

Table E.1 — Overview of XML schemas

Name	Content	Namespace
FDCML.xsd	Basic Classes	http://www.fdcml.org
FDCMLdt.xsd	Data Type Definitions	http://www.fdcml.org
FDCMLISO15745DeviceFunction.xsd	Device Function Classes	http://www.fdcml.org/ISO15745DeviceFunction
FDCMLTextResource.xsd	XML schema for text resources	http://www.fdcml.org/TextResource
xmldef.xsd	Definitions in the XML namespace	http://www.w3.org/XML/1998/namespace
xlinkdef.xsd	Definitions in the XLink namespace	http://www.w3.org/1999/xlink

E.1.2 Basics

E.1.2.1 Data type elements

The device profile uses data types defined in IEC 61158-5:2003 clause 5. To allow additional attributes the data type information is modelled using XML element type declarations. IEC 61131-3 and VARIANT data type aliases are provided as fixed attributes. These data type elements are defined in a XML schema named "FDCMLdt.xsd" which is defined in E.1.6.2.

E.1.2.2 Collection pattern

The device profile uses the Collection Element Pattern. These elements appear in list form of the collected elements (example: `processDataDescriptionList / processDataDescription`)

E.1.2.3 Descriptive text for elements

E.1.2.3.1 General

The device profile offers three distinct possibilities to provide descriptive text for elements. E.1.2.3.2 and E.1.2.3.3 are to be used exclusively. E.1.2.3.4 may be used together with E.1.2.3.2 or E.1.2.3.3.

E.1.2.3.2 Text embedded in the device profile

Every element which requires a descriptive text shall have a `label` child element with the attribute `xml:lang`. This allows multiple languages within one device profile. Additionally a short help text may be provided with the `help` element. It has the attribute `xml:lang`, too. This attribute is composed of a two-letter language code and an optional two-letter country code separated by a dash. The format is:

`ISO 639 code for name of language ["-" ISO 3166-1-Alpha-2 code]`

EXAMPLE `xml:lang='pt-pr'` states a Portugese text with Brazilian wording.

E.1.2.3.3 Text provided by external text resource files

The elements `labelRef` and `helpRef` shall provide a pointer to a text resource stored in an external text resource file. The AIF has to replace these references by the text provided by the text resource file. The text resource files used by a device profile shall be defined in a `dictionary` element.

The format of the text resource file is defined in the XML schema "FDCMLTextResource.xsd" which is defined in E.1.6.4.

E.1.2.3.4 Pointer to external documentation

Additionally a pointer to an external documentation may be provided. The `helpRefFile` element shall point to a distinctive position in a file defined with the `helpFile` element.

EXAMPLE Examples for external documentation are *.hlp, *.htm or *.pdf files.

E.1.2.4 Valid element values

If a device profile element has a value one of the following value descriptions shall be used:

- `const` constant element value
- `edit` editable string value
- `enumeration` enumerated value
- `range` value range, a value can have more than one range
- `yes,no` TRUE/FALSE value combination
- `reference` reference to another element
- `instanceValue` value if a instance instead of a profile or type is decribed

E.1.2.5 Modelling of conditional device behaviour

To model conditional device behaviour, the following elements shall be used:

- disable disable the referenced device property depending on the value of this device property
- enable enable the referenced device property depending on the value of this device property
- change change the referenced device property value depending on the value of this device property

Every possible target of a disable or enable has an `enabled` attribute in the XML schema.

E.1.2.6 Internal referencing of elements

A path specified in a `ref` attribute shall be a valid XPath (see [8]) path.

E.1.2.7 Unique element identification

Elements, which require identification, have the attribute `uniqueID`. The value of this unique identifier shall consist of:

`token_index[[_subindex]...]`, index and subindex of type unsigned16

E.1.2.8 Assemblies

All Assembly elements allow a grouping of their respective object elements. An assembly contains a list of XPath pointers to the assembled objects.

E.1.2.9 Vendor specific categories

An AIP Designer may add category elements to the collection elements.

EXAMPLE An AIP Designer adds a `processDataCategory` for input signals and a `processDataCategory` for output signals.

E.1.3 DeviceIdentity object - deviceType object

Table E.2 shows a list of predefined values for the `deviceType` element.

NOTE The values in Table E.2 are given in the language 'en-us', 'de', 'fr', 'es' and 'it'. Device profiles may contain additional languages.

Table E.2 — Device types

en-us	de	fr	es	it
Actuator	Aktor	Actionneur	Activador	Attuatore
Bus Coupler	Buskoppler	Coupleur de bus	Acoplador de bus	Accoppiatore bus
Closed Loop Controller	Regler	Régulateur en boucle fermée	Regulador	Regolatore
Dosing Device	Dosiergerät	Équipement doseur	Dispositivo dosificador	Dosatore
Drive	Antrieb	Moteur	Accionamiento	Azionamento
Drive - Frequency Inverter	Antrieb - Frequenzumrichter	Moteur - Variateur de vitesse	Accionamiento - Desviador de frecuencia	Azionamento - Inverter
Drive - Motor Starter	Antrieb - Motorschalter	Moteur - Démarreur	Accionamiento - Conmutador de motor	Azionamento - Salvamotore
Drive - Servo Amplifier	Antrieb - Servoverstärker	Moteur - Amplificateur d'asservissement	Accionamiento - Refuerzo por servomotor	Azionamento - Servoamplificatore
Drive - Stepper Motor Controller	Antrieb - Schrittmotor-Steuerungcontroller	Moteur - Variateur pas à pas	Accionamiento - Controlador de dirección del motor de pasos	Azionamento - Controller di comando motore passo-passo
Encoder	Encoder	Codeur	Codificador	Encoder
Gateway	Gateway	Passerelle	Pasarela	Gateway
General	Allgemeines	Généralités	Varios	Generalità

Table E.2 — Device types *(continued)*

en-us	de	fr	es	it
HMI	HMI	IHM	HMI	HMI
HMI Display	HMI-Anzeige	Affichage IHM	Representación HMI	Visualizzazione HMI
HMI Operator Panel	HMI-Bediengerät	Pupitre de commande IHM	Dispositivo de manejo HMI	Apparecchio di comando HMI
Hydraulic Device	Hydraulik-Gerät	Équipement hydraulique	Dispositivo hidráulico	Apparecchio idraulico
I/O	E/A	E/S	E/S	I/O
I/O analog	E/A analog	E/S analogique	E/S analogical	I/O analogico
I/O digital	E/A digital	E/S numérique	E/S digital	I/O digitale
I/O Function Module	E/A-Funktionsmodul	Module fonction E/S	Módulo de función E/S	Modulo funzione I/O
Identification System	Identifikationssystem	Système d'identification	Sistema de identificación	Sistema di identificazione
Media Converter active	Medienkonverter aktiv	Convertisseur de support actif	Conversor de medios activo	Convertitore di mezzi attivo
Media Converter passive	Medienkonverter passiv	Convertisseur de support passif	Conversor de medios pasivo	Media Converter passive
NC	NC	NC	NC	NC
NC/RC	NC/RC	NC/RC	NC/RC	NC/RC
PC	PC	PC	PC	PC
PC Board	PC-Karte	Carte PC	Tarjeta PC	Scheda per PC
PLC	SPS	API	SPS	PLC
PLC board	SPS-Karte	Carte API	Tarjeta SPS	Scheda per PLC
Pneumatic Device	Pneumatik-Gerät	Équipement pneumatique	Dispositivo neumático	Apparecchio pneumatico
Positioning Controller	Positionier-Steuerung	Commande de positionnement	Control de posición	Unità di controllo posizionamento
Power Supply	Stromversorgung	Alimentation	Alimentación eléctrica	Alimentazione elettrica
RC	RC	RC	RC	RC
Sensor	Sensor	Capteur	Sensor	Sensore
Switching Device	Schaltgerät	Appareil de connexion	Dispositivo de conmutación	Commutatore
Technology Controller	Technologie-Steuerung	Contrôleur haute technologie	Control tecnológico	Controllori ad alte tecnologie
Valve	Ventil	Vanne	Válvula	Valvola
Weighing or Batching System	Wiege- oder Dosiersystem	Système de pesée ou de dosage	Sistema de pesado o dosificación	Sistema di pesatura o di dosaggio
Welding Controller	Schweißsteuerung	Commande de soudure	Control de soldado	Unità di controllo saldatura
Wrenching Controller	Schraubersteuerung	Visseuse	Control de atornillado	Unità di controllo avvitatrice

E.1.4 DeviceManager object

E.1.4.1 datatypeTemplateList, datatypeTemplate objects

The `datatypeTemplate` allows the definition of AIP Designer or Device Profile specific data types. These data types are invoked with the `datatypeInstance` element. These data types can be described as follows:

— `directlyDerivedType` directly derived data type

— `enumeratedType` enumerated data type or list of constants (C-style enumeration)

— `subrangeType` range data type

— `arrayType` array data type

— `structuredType` structured data type

E.1.4.2 Attributes of the communicationEntity object

Table E.3 describes the attributes of the `communicationEntity` object as defined in 6.5.1.3.4.1.

Table E.3 — Attributes of `communicationEntity` object

Attribute	Description	Data Type	Values
protocol	communication protocol	xsd:string	'INTERBUS'
communicator	specifies, whether this entity takes an active role in the network communication	xsd:string	'YES' – participates in network communication 'NO' – does not participate in network communication
communicationEntityType	type of communication entity	xsd:string	'SLAVE' 'MASTER' 'CLIENT' 'SERVER' 'PEER' (an entity which acts as client and server) 'MASTER_SLAVE' (an entity which acts a master and slave) 'DEVICEMODULE' (an entity which needs a parent to communicate via a network 'PASSIVE' (an entity which does not participate in network communication)
communicationProfile	communication profile identifier	xsd:string	See 6.5.2.4.3

E.1.4.3 Mapping communication network profile classes to device profile elements

Table E.4 provides a list which communication network profile classes are mapped to which dedicated configuration items of the device manager object of an INTERBUS master profile. Table E.5 shows the respective mapping for an INTERBUS slave profile.

Table E.4 — Mapping of communication network profile classes to dedicatedCfgItem objects for an INTERBUS master

communication network profile class or attribute	dedicatedcfgItemType	Data Type	Unit	Description
baud500k, baud2M, baud8M, baud16M	IB:Baudrate	uint	kbit/s	supported Baudrates
maxDeviceCount	IB:MaxDeviceCount	uint	-	max. number of supported devices
maxIOCount	IB:MaxIOCount	uint	bit	max. number of I/O data
maxLevelCount	IB:MaxLevelCount	uint	-	max. number of supported levels
maxPCPCount	IB:MaxPCPCount	uint	-	max. number of PCP devices
maxSegmentCount	IB:MaxSegmentCount	uint	-	max. number of supported segments
—	IB:PCPMaxClientParallel	usint	-	max. number of parallel PCP services per connection as client
—	IB:PCPMaxServerParallel	usint	-	max. number of parallel PCP services per connection as server
—	IB:MaxCRCount	uint	-	max. number of communication references
—	IB:MaxLocalbusDevices	usint	-	max. number of local bus devices per segment
read	IB:PCPServerRead	Boolean	[a]	PCP Read supported as Server
write	IB:PCPServerWrite	Boolean	[a]	PCP Write supported as Server
—	IB:PCPServerGetODLong	Boolean	[a]	PCP Get OD Long as Server
—	IB:PCPServerUpload	Boolean	[a]	PCP Upload as Server
—	IB:PCPServerDownload	Boolean	[a]	PCP Download as Server
—	IB:PCPServerInfoReport	Boolean	[a]	PCP Information Report as Server
—	IB:PCPServerReqDomain	Boolean	[a]	PCP Request Domain Upload as Server
start, stop, resume, reset	IB:PCPServerFunctionInvocation	Boolean	[a]	PCP Start, Stop, Resume, Reset as Server
—	IB:PCPServerRWWName	Boolean	[a]	PCP Read With Name, Write With Name as Server
—	IB:PCPClientRead	Boolean	[a]	PCP Read supported as Client
—	IB:PCPClientWrite	Boolean	[a]	PCP Write supported as Client
—	IB:PCPClientGetODLong	Boolean	[a]	PCP Get OD Long as Client
—	IB:PCPClientUpload	Boolean	[a]	PCP Upload as Client
—	IB:PCPClientDownload	Boolean	[a]	PCP Download as Client
—	IB:PCPClientInfoReport	Boolean	[a]	PCP Information Report as Client
—	IB:PCPClientReqDomain	Boolean	[a]	PCP Request Domain Upload as Client
—	IB:PCPClientFunctionInvocation	Boolean	[a]	PCP Start, Stop, Resume, Reset as Client
—	IB:PCPClientRWWName	Boolean	[a]	PCP Read With Name, Write With Name as Client
—	IB:PNM7ServerLoadCRL	Boolean	[a]	PNM7 Load CRL as Server
—	IB:PNM7ServerReadCRL	Boolean	[a]	PNM7 Read CRL as Server
—	IB:PNM7ClientLoadCRL	Boolean	[a]	PNM7 Load CRL as Client
—	IB:PNM7ClientReadCRL	Boolean	[a]	PNM7 Read CRL as Client
—	IB:PCPServerParallel	usint	-	number of parallel PCP services as Server
—	IB:PCPClientParallel	usint	-	number of parallel PCP services as Client
—	IB:PDUSizeReceive	usint	octet	PCP PDU Size (reveive)
—	IB:PDUSizeSend	usint	octet	PCP PDU Size (send)
—	IB:PNM7PDUSizeReceive	usint	octet	PNM7 PDU Size (receive)
—	IB:PNM7PDUSizeSend	usint	octet	PNM7 PDU Size (send)
[a] boolean value of '0' denotes 'not supported', a value of '1' denotes 'supported'				

An AIP Designer may specify additional implementation specific dedicated configuration items.

Table E.5 — Mapping of communication network profile classes to dedicatedCfgItem objects for an INTERBUS slave

communication network profile Class or Attribute	DedicatedcfgItemType	Data Type	Unit	Description
assignedIDCode	IB:IDCode	usint	-	ID code[b]
inLen	IB:InLen	uint	bit	process data input length
outLen	IB:OutLen	uint	bit	process data output length
length of processDataChannel	IB:PDLen	uint	bit	process data channel length
length of parameterChannel	IB:PCPLen	unsigned16	octet	length of the parameter channel
read	IB:PCPServerRead	Boolean	[a]	PCP Read supported as Server
write	IB:PCPServerWrite	Boolean	[a]	PCP Write supported as Server
—	IB:PCPServerGetODLong	Boolean	[a]	PCP Get OD Long as Server
—	IB:PCPServerUpload	Boolean	[a]	PCP Upload as Server
—	IB:PCPServerDownload	Boolean	[a]	PCP Download as Server
—	IB:PCPServerInfoReport	Boolean	[a]	PCP Information Report as Server
—	IB:PCPServerReqDomain	Boolean	[a]	PCP Request Domain Upload as Server
start, stop, resume, reset	IB:PCPServerFunctionInvocation	Boolean	[a]	PCP Start, Stop, Resume, Reset as Server
—	IB:PCPServerRWWName	Boolean	[a]	PCP Read With Name, Write With Name as Server
—	IB:PCPClientRead	Boolean	[a]	PCP Read supported as Client
—	IB:PCPClientWrite	Boolean	[a]	PCP Write supported as Client
—	IB:PCPClientGetODLong	Boolean	[a]	PCP Get OD Long as Client
—	IB:PCPClientUpload	Boolean	[a]	PCP Upload as Client
—	IB:PCPClientDownload	Boolean	[a]	PCP Download as Client
—	IB:PCPClientInfoReport	Boolean	[a]	PCP Information Report as Client
—	IB:PCPClientReqDomain	Boolean	[a]	PCP Request Domain Upload as Client
—	IB:PCPClientFunctionInvocation	Boolean	[a]	PCP Start, Stop, Resume, Reset as Client
—	IB:PCPClientRWWName	Boolean	[a]	PCP Read With Name, Write With Name as Client
—	IB:PNM7ServerLoadCRL	Boolean	[a]	PNM7 Load CRL as Server
—	IB:PNM7ServerReadCRL	Boolean	[a]	PNM7 Read CRL as Server
—	IB:PNM7ClientLoadCRL	Boolean	[a]	PNM7 Load CRL as Client
—	IB:PNM7ClientReadCRL	Boolean	[a]	PNM7 Read CRL as Client
—	IB:PCPServerParallel	usint	-	number of parallel PCP services as Server
—	IB:PCPClientParallel	usint	-	number of parallel PCP services as Client
—	IB:PDUSizeReceive	usint	octet	PCP PDU Size (reveive)
—	IB:PDUSizeSend	usint	octet	PCP PDU Size (send)
—	IB:PNM7PDUSizeReceive	usint	octet	PNM7 PDU Size (receive)
—	IB:PNM7PDUSizeSend	usint	octet	PNM7 PDU Size (send)
baud500k	IB:500kBaudSupp	Boolean	[a]	500 kbit/s supported
baud2M	IB:2MBaudSupp	Boolean	[a]	2 Mbit/s supported
baud8M	IB:8MBaudSupp	Boolean	[a]	8 Mbit/s supported
baud16M	IB:16MBaudSupp	Boolean	[a]	16 Mbit/s supported

[a] boolean value of '0' denotes 'not supported', a value of '1' denotes 'supported'

[b] least significant octet of an INTERBUS device code containing device class, data direction or parameter channel length

An AIP Designer may specify additional implementation specific dedicated configuration items.

E.1.4.4 Attributes of `processDataDescription` objects

Table E.6 describes the attributes of the `processDataDescription` object as defined in 6.5.1.3.4.3.

Table E.6 — Attributes of `processDataDescription` object

Attribute	Description	Data Type	Values
direction	data direction	xsd:string	'I': input 'Q': output
processDataDescriptionType	type of data item	xsd:string	"IBPD"

E.1.4.5 Attributes of `parameterDescription` objects

Table E.7 describes the attributes of the `parameterDescription` object as defined in 6.5.1.3.4.5.

Table E.7 — Attributes of `parameterDescription` object

Attribute	Description	Data Type	Values
access	access rights	xsd:string	'RO': READONLY 'RW': READWRITE 'WO': WRITEONLY
parameterDescriptionType	type of parameter item	xsd:string	"IBPAVariable" for a variable "IBPAFunctionInvocation" for a FunctionInvocation

E.1.4.6 Attributes of `logicalConnectionPoint` objects

Table E.8 describes the attributes of the `logicalConnectionPoint` object as defined in 6.5.1.3.4.8.

Table E.8 — Attributes of `logicalConnectionPoint` object

Attribute	Description	Data Type	Values
maxRelationships	number of maximum possible communication relationships	xsd:nonNegativeIntgeger	
role	role of logical connection point	xsd:string	"CLIENT" "SERVER" "PEER" "PUBLISHER" "SUBSCRIBER" "PUBLISHERSUBSCRIBER"
logicalConnectionPointType	type of logical connection point	xsd:string	"IBPD": process data channel "IBPA": parameter channel
newLevel	defines if this connection point open a new structural level	xsd:string	"YES" "NO"

NOTE A `logicalConnectionPoint` has one or more provides child elements. If provides contains a ref attribute value of `"//parameterItem[@parameterItemType='IBPAVariable']"` it resembles the parameter object dictionary.

E.1.4.7 Attributes of `channel` objects

Table E.9 describes the attributes of the `channel` object as defined in 6.5.1.3.3.2.

Table E.9 — Attributes of `channel` object

Attribute	Description	Data Type	Values
channelType	type of channel	xsd:string	
direction	direction of data flow through this channel	xsd:string	"I": Input "Q": Output "X": don't care

E.1.4.8 Attributes of `MAU` objects

Table E.10 describes the attributes of the `MAU` object as defined in 6.5.1.3.3.3.

Table E.10 — Attributes of `MAU` object

Attribute	Description	Data Type	Values
interfaceType	provides additional information on the type of the MAU	xsd:string	"IBREMOTE" "IBLOCAL"
direction	defines the logical direction of the data flow through this MAU	xsd:string	"INOUT": transmit and receive "IN": receive and loopback circuit "OUT": transmit and loopback circuit "IN_UNI": receive only "OUT_UNI": transmit only
directlyConnected	defines if other devices are connected directly to this MAU	xsd:string	"YES" "NO"
MAUType	vendor specific MAU Type identifier	xsd:string	value is vendor specific
newLevel	defines, if a new structual level is opened by this MAU (example: local bus branch of a bus terminal)	xsd:string	"YES" "NO"
protocol	defines the protocol run by the MAU	xsd:string	"INTERBUS"
sequenceNumber	identification of the MAU, starting with 1, separatly numbered for each direction	xsd:nonNegativeInteger	default = 1

E.1.4.9 Attributes of `slot` objects

Table E.11 describes the attributes of the `slot` object as defined in 6.5.1.3.3.4

Table E.11 — Attributes of `slot` object

Attribute	Description	Data Type	Values
number	indicates the position at which a child device may be added in the sequence of child devices	xsd:string	The `number` attribute can contain a single number (`'3'`), a list of numbers (`'3,5,7'`), a number range (`'1-3'`) or a combination (`'1-2,4,6,9-10'`) [1].

[1] If the child devices may be attached to any slot, use a number, defining the maximum number of attachable child devices. If for example 64 child devices may be attached to a parent device use `slot number='1-64'`. Every slot position requires a `MAUUsage` element pointing to an attached MAU.

E.1.4.10 Attributes of `LED` and `LEDState` objects

Table E.12 describes the attributes of the `LED` object defined in 6.5.1.3.3.5. The child object `LEDState` describes distinct states. The attributes of the `LEDState` are described in Table E.13.

Table E.12 — Attributes of LED object

Attribute	Description	Data Type	Values
LEDType	type of LED	xsd:string	"IOStatus" "IODiagnostic" "DeviceStatus" "DeviceDiagnostic" "CommStatus" "CommDiagnostic"

Table E.13 — Attributes of `LEDState` object

Attribute	Description	Data Type	Values
LEDCondition	condition of LED in state	xsd:string	"ON" "OFF"
LEDColor	color of LED in state	xsd:string	"GREEN" "YELLOW" "RED" "ORANGE" "BLUE" "WHITE"
LEDFrequency	frequency of blink rate in state in Hz	xsd:float	
LEDFlashCount	number of flashes in state	xsd:nonNegativeInteger	
ref	XPath to object or state of object causing the LED state	xsd:string	

E.1.5 Supplementary element descriptions

E.1.5.1 `accessPath` object

An accessPath object shall describe a path relative to a `communicationEntity` for an application to gain access to a `processDataDescription`, `parameterDescription`, or `localDataDescription` object. The format is:

— for a `processDataDescription`: byteoffset[.bitoffset] bitoffset is optional; 0 = LSB

— for a `parameterDescription`: index[#subindex] subindex is optional

— for a `localDataDescription`: undefined

NOTE The format for a `localDataDescription` is out of the scope of this International Standard.

Figure E.1 illustrates the order and numbering of the byte and bit offsets.

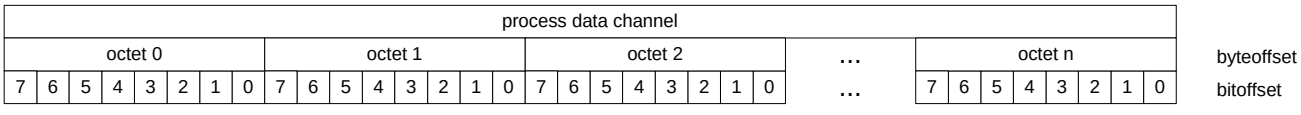

Figure E.1 — Byte and bit offsets in the process data channel