CiA 311 Draft Standard Proposal

CANopen device description

XML schema definition

This DSP is for CiA members only and may be changed without notification.

Version: 1.1.0

10 August 2011

HISTORY

Date	Changes
2007–03–08	*Publication of Version 1.0 as draft standard proposal*
2007–07–17	*Publication of Version 1.0.2 as draft standard proposal*
2011–08–10	*Publication of Version 1.1 as draft standard proposal*
	— added modulare device definition
	— added signal definition

General information on licensing and patents

CAN in AUTOMATION (CiA) calls attention to the possibility that some of the elements of this CiA specification may be subject of patent rights. CiA shall not be responsible for identifying any or all such patent rights.

Because this specification is licensed free of charge, there is no warranty for this specification, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holder and/or other parties provide this specification “as is” without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the correctness and completeness of the specification is with you. Should this specification prove failures, you assume the cost of all necessary servicing, repair or correction.

Trademarks

CANopen® and CiA® are registered community trademarks of CAN in Automation. The use is restricted for CiA members or owners of CANopen vendor ID. More detailed terms for the use are available from CiA.

CAN in Automation e. V.
Kontumazgarten 3
DE - 90429 Nuremberg, Germany
Tel.: +49-911-928819-0
Fax: +49-911-928819-79
Url: www.can-cia.org
Email: headquarters@can-cia.org

CONTENTS

1 Scope

This specification defines the elements and rules for describing device profiles and communication network profiles for devices used in CANopen-based control systems. The content of this specification complies with the definitions and provisions of ISO 15745-1:2005/Amd1.

NOTE Part 1 of ISO 15745 specifies generic elements and rules for describing integration models and application interoperability profiles, together with their component profiles. The formal description is based on XML technology.

2 References

/ISO15745-1/	ISO 15745-1:2003, Industrial automation systems and integration — Open systems application integration framework — Part 1: Generic reference description
/ISO15745-1Amd/	ISO 15745-1:2005/Amd1, Industrial automation systems and integration — Open systems application integration framework — Part 1: Generic reference description (proposed Amendment 1)
/ISO639-2/	ISO 639-2, Codes for the representation of names of languages — Part 2: Alpha-3 code
/ISO3166-1/	ISO 3166-1, Codes for the representation of names of countries and their subdivisions — Part 1: Country codes
/IEC61131-3/	IEC 61131-3, Programmable controllers — Part 3: Programming languages
/ISO1000/	ISO 1000, SI units and recommendations for the use of their multiples and of certain other units

3 Notational conventions

Figure 1 explains the notational conventions used in all those figures within this document, which illustrate the structure of XML schema partitions. The notation is the one used by a specific XML tool.

The figure shows the hierarchical arrangement of sequences or choices of elements. All these items can be characterized as mandatory, optional, or having a given multiplicity - see the different examples in the figure.

Elements without sub-structure may have textual content (icon showing lines - example: Element11)

Elements can be declared local inside parent elements (no arrow shown), or they can be declared global, which means that they can be referenced (arrow shown) from one to many places in the structure (example: Element1 is referenced from inside the ROOT element and from inside Element2).

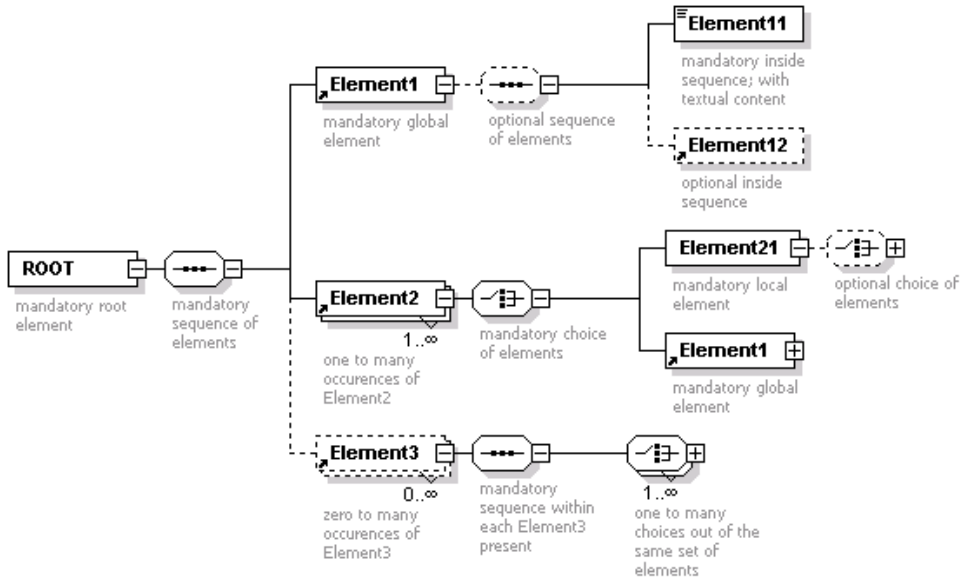

Figure 1 — Notational conventions

4 Abbreviations

4.1 HMI

Human Machine Interface

4.2 ISO

International Organization for Standardization

4.3 LED

Light Emitting Diode

4.4 OSI

Open Systems Interconnection

4.5 URI

Uniform Resource Identifier

4.6 XML

Extensible Markup Language

5 CANopen profiles

5.1 Device profile

5.1.1 General

Figure 2 shows the element structure of a CANopen device profile. These elements are further decomposed and detailed in 6.4.

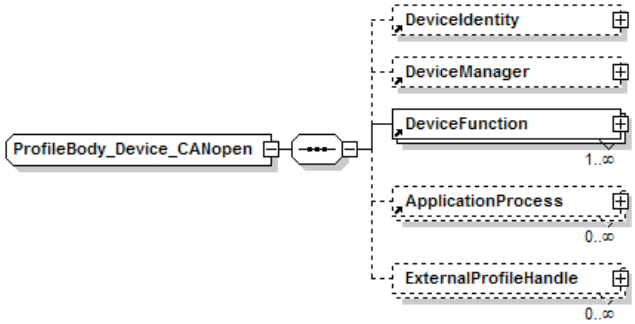

Figure 2 — CANopen device profile element

NOTE 1 The additional ExternalProfileHandle element shown in Figure 2 enables external non-XML data to be referenced.

NOTE 2 The ProfileIdentification, ProfileRevision, and ProfileLocation attributes of the ProfileHandle_DataType refer to the external non-XML data file. This can be used by legacy systems that are in the process of migrating to XML.

The XML schema representing the CANopen device profile template is defined in A.1. This template is based on two parts:

— the CANopen_Profile_Header defined in A.1.1,

— the CANopen_Device_Profile defined in A.1.3.

5.1.2 Device identity

The DeviceIdentity element contains attributes that are independent of the network and of the process, and which uniquely identify the device.

Figure 3 shows the structure of the CANopen DeviceIdentity element.

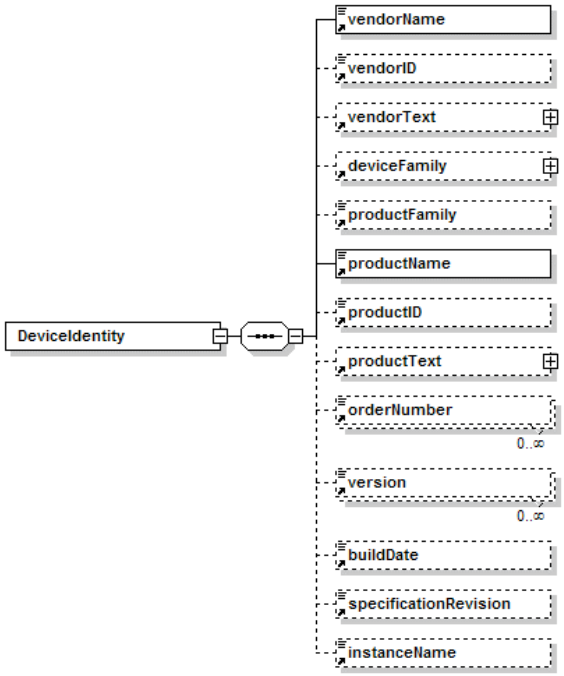

Figure 3 — CANopen DeviceIdentity element

Further details are given in 6.4.2.

5.1.3 Device manager

The DeviceManager element contains attributes and supports services that enable the monitoring of the device. Communication specific configuration data and mapping information is defined in the communication network specific part structured according to the schema specified in 6.5.

Figure 4 shows the structure of the CANopen DeviceManager element

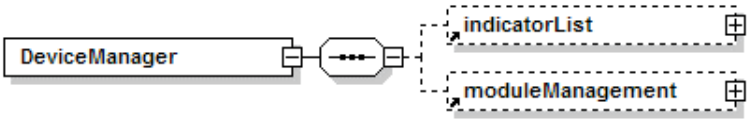

Figure 4 — CANopen DeviceManager element

Further details are given in 6.4.3.

5.1.4 Device function

The DeviceFunction element describes the intrinsic function of a device in terms of its technology. It contains network independent descriptions/definitions of the technological device functionality.

Figure 5 shows the structure of the CANopen DeviceFunction element.

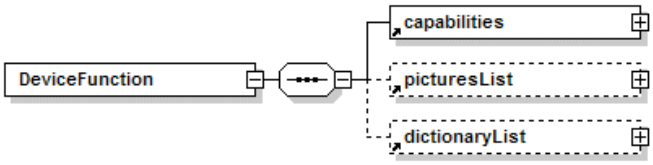

Figure 5 — CANopen DeviceFunction element

Further details are given in 6.4.3.3.

5.1.5 Application process

The ApplicationProcess element represents the set of services and parameters, which constitute the behavior and the interfaces of the device in terms of the application, independent of the device technology and the underlying communication networks and communication protocols.

Figure 6 shows the structure of the CANopen ApplicationProcess element.

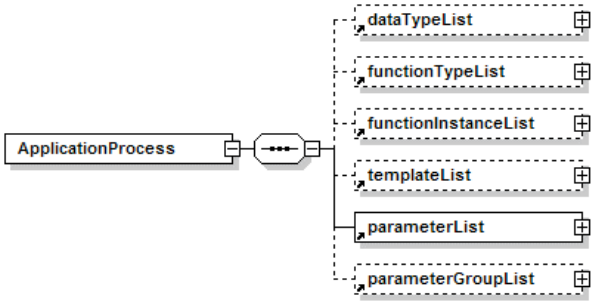

Figure 6 — CANopen ApplicationProcess element

Further details are given in 6.4.5.

5.2 Communication network profile

5.2.1 General

Figure 7 shows the element structure of the CANopen communication network profile. These elements are further decomposed and detailed in 6.5.

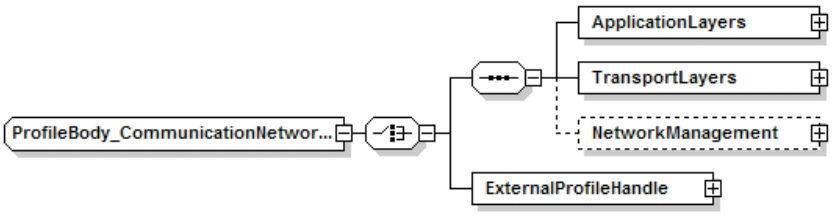

Figure 7 — CANopen communication network profile element

The XML schema representing the CANopen communication network profile is defined in Annex A.

NOTE The additional ExternalProfileHandle element shown in Figure 7 enables external non-XML data to be referenced. The ProfileIdentification, ProfileRevision, and ProfileLocation attributes of the ProfileHandle_DataType refer to the external non-XML data file. This can be used by legacy systems that are in the process of migrating to XML.

5.2.2 Application layers

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

Further details are given in 6.5.2.

5.2.3 Transport layers

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

Further details are given in 6.5.2.6.

5.2.4 Network management

The CANopen NetworkManagement element represents the network configuration and performance adjustment capabilities of the CANopen communication network integration model.

Further details are given in 6.5.4.

6 CANopen profile templates

6.1 Overview

The CANopen technology uses the concept of the multi-profile container specified in /ISO15745-1Amd/ for XML profile files. Therefore, CANopen profile templates are based on the alternate ISO15745ProfileContainer master profile template specified in amendment 1 of /ISO15745-1/.

Figure 8 shows the structure of a CANopen XML profile. Two types of ProfileBody are defined for CANopen profiles:

— ProfileBody_Device_CANopen and

— ProfileBody_CommunicationNetwork_CANopen.

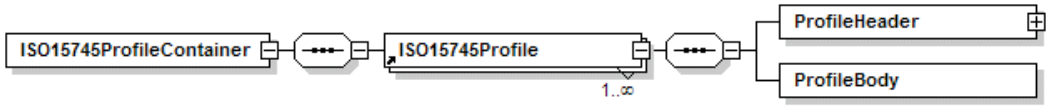

Figure 8 — CANopen master profile template

The ProfileTechnology name according to /ISO15745-1/ is CANopen.

6.2 General rules

6.2.1 Using unique IDs

An element can have the attribute uniqueID of type xsd:ID. The unique identifier therefore is forced to be unique in the whole XML file. An element that references the unique identifier contains a named attribute of type xsd:IDREF.

6.2.2 Language support

6.2.2.1 General

Device profiles complying with the XML schema described in this annex need a support of different languages, since tools are then able to use names out of the XML file in order to display them in their user interface. Communication parameters for example may be presented in the user interface of a tool.

The language support is implemented via the label group g_labels. Each name of an element, which would possibly be displayed and is therefore language dependent, is provided inside the schema as a g_labels element. Optionally, a URI may be added as an attribute to the label element.

EXAMPLE (for a given parameter name)

— German: Baud-Rate

— English: Baud rate

— French: Vitesse de transmission

6.2.2.2 Element g_labels

The group g_labels supports the introduction of a label (name) and a description in the context of the parent element (see Figure 9).

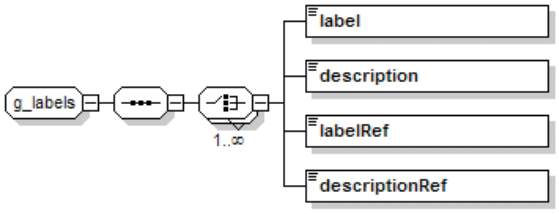

Figure 9 — Group g_labels

Every element, which needs a name or a description, shall select one or more suitable of the four elements to perform this task: the label, the description, the labelRef and the descriptionRef element.

1) The label element allows storage of the identifying name and the descriptive text inside the XML file itself. The label element has the attributes given in Table 1. The element may appear n times, once for each language. For identifying the language, the lang attribute is used.

Table 1 — Attributes of element label

Attribute	Data type	Use	Description
lang	xsd:language	required	Language used for the name

2) The description element allows storage of textual descriptions inside the XML file itself. The element may appear several times, once for each language.

Table 2 — Attributes of element description

Attribute	Data type	Use	Description
lang	xsd:language	required	Language used for the description
URI	xsd:anyURI	optional	Optional link to further descriptive information

3) The labelRef element allows storage of a reference to an external text resource file (see B.4).

The labelRef element provides a pointer via its attributes dictID and textID to a text entry in a separate text resource file. These text resource files are referenced in the dictionary sub-elements of the DeviceFunction element.

The labelRef element also may appear n times, to allow references to several dictionary entries, which contain links to files in different languages. The respective language is defined in the lang attribute of the dictionary element.

The labelRef element contains the attributes given in Table 3.

Table 3 — Attributes of element labelRef

Attribute	Data type	Use	Description
dictID	xsd:string	required	References a single dictionary element inside the dictionaryList element; the dictionary element contains a link to the external text resource file
textID	xsd:string	required	References a character sequence (xsd:string) inside the external text resource file.

4) The descriptionRef element allows storage of reference descriptive texts inside an external text resource file. The definitions from the labelRef element shall apply for the descriptionRef element.

Both the labelRef and the descriptionRef element may have a link to additional descriptive information as their content in the form of a URI.

6.2.2.3 Language identifier

For the multi-language support each label gets an attribute with the content of the language code. The language code corresponds to the content of the label element.

In order to verify which languages are supported in the XML file, the attribute supportedLanguages in the ProfileBody element lists the supported languages.

6.2.2.4 Attribute lang

The language identifier lang consists of a combination of a language code (as defined in /ISO639-2/) plus an optional dash character plus an optional country code (as defined in /ISO3166-1/). Lang is an attribute of the label element and the description element.

Some of the values for lang are given in Table 4.

Table 4 — Values of attribute lang

Language	value of lang
English (United States)	en-us
German (Standard)	de
French (Standard)	fr
Spanish (Standard)	es
Italian (Standard)	It
Portuguese (Brazil)	pt-br

6.2.2.5 Attribute supportedLanguages

The supportedLanguages attribute of the ProfileBody element identifies supported languages and consists of a list of language codes plus optional country codes.

EXAMPLE

supportedLanguages=”en-us de fr es”

6.2.2.6 URIs

A general mechanism allows specification of a URI in the context of a description element, a labelRef element and a descriptionRef element.

EXAMPLE This is used in the context of a vendor label, parameter label or services label.

6.3 ProfileHeader

To facilitate the identification of a profile, the profile header of the device profile as well as the communication network profile shall comply with the model shown in Figure 10, which is directly inherited from /ISO15745-1/.

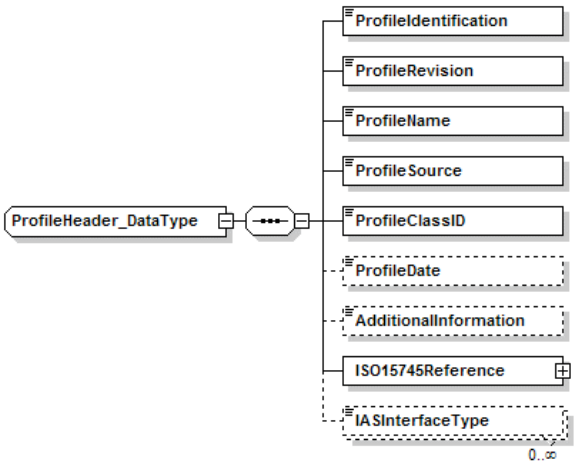

Figure 10 — ProfileHeader element

The ProfileHeader element is composed of the following elements:

— the ProfileIdentification element identifies the current profile,

— the ProfileRevision element identifies the current profile revision,

— the ProfileName element contains a descriptive English name of the current profile. In case that more than one ProfileBody element are present within a device profile, it is suggested that the value of the ProfileName element should be the concatenation of the values of the productName elements inside the respective DeviceIdentity elements,

— the ProfileSource element identifies the validator of the current profile,

— the ProfileClassID element identifies the class of the current profile according to /ISO15745-1/,

— the ISO15745Reference element states the ISO 15745 part, edition and technology, to which the description conforms.

6.4 Device profile template description

6.4.1 ProfileBody_Device_CANopen

For the device profile the ProfileBody contains the DeviceIdentity, the DeviceManager, the DeviceFunction and the ApplicationProcess elements shown in Figure 2.

The ProfileBody element contains the description

— of a single device (e.g. a proximity sensor or an electromechanical limit switch), or of a more complex one (e.g. a circuit breaker with up to 2500 parameters, more than 100 functions),

— or of a part of a device also called "module" in the PLC world (e.g. part of an I/O controller or of an electrical protection unit).

The ProfileBody element contains the attributes given in Table 5.

Table 5 — Attributes of element ProfileBody

Attribute	Data type	Use	Description
formatName	xsd:string	fixed	Format identifier
formatVersion	xsd:string	fixed	Format version identifier
fileName	xsd:string	required	Name of the file with extension without path
fileCreator	xsd:string	required	Person creating the file
fileCreationDate	xsd:date	required	Date of file creation
fileCreationTime	xsd:time	optional	Time of file creation
fileModifiedBy	xsd:string	optional	Person modifying the file
fileModificationDate	xsd.date	optional	Date of last file change
fileModificationTime	xsd:time	optional	Time of last file change
fileVersion	xsd:string	required	Vendor specific version of the file
supportedLanguages	xsd:NMTOKENS	optional	List of supported languages
deviceClass	xsd:NMTOKEN	optional	Classification of the device profile; valid values: — compact – the profile is complete and unique for a device or device family — modular – a bounded set of profiles exist according to the modular functionalities combined in a specific device occurrence — configurable – open configurable device that needs an external configurator to create the profile of one instance

6.4.2 DeviceIdentity

6.4.2.1 General

The DeviceIdentity element (see Figure 3) contains elements, which are independent of the network and of the process. It describes the identity of a single device or of a group of devices.

Table 6 specifies the attribute readOnly, which is attached to the vendorName, vendorID, vendorText, deviceFamily, productFamily, productName, productID, productText, orderNumber, version, specificationRevision and instanceName elements.

Table 6 — Attribute of element vendorName

Attribute	Data type	Use	Description
readOnly	xsd:boolean	default	Indicates whether the value is read-only for a user: false, true (default)

6.4.2.2 vendorName

The vendorName element identifies the name or the brand name of the vendor of the device.

6.4.2.3 vendorID

The vendorID element identifies the vendor. This information has to be filled in when the device described is recognized and validated by a consortium.

NOTE Consortia specific device families and vendor identifiers are linked.

6.4.2.4 vendorText

The vendorText element allows the vendor to provide additional company information, like address or hotline number. The g_labels group offers the possibility to include a vendor URI inside the vendorText element.

6.4.2.5 deviceFamily

The deviceFamily element states the family of the device.

EXAMPLE

Examples for device families are:

- Variable Speed Drive
- Circuit Breaker
- Pressure Sensor

6.4.2.6 productFamily

The productFamily element states a vendor specific affiliation of the device type to a certain set of devices inside a family. The list of valid productFamily values is system, tool or consortia specific.

NOTE Consortia specific device families and vendor identifiers are linked.

6.4.2.7 productName

The productName element states a vendor specific designation or name of the device type.

6.4.2.8 productID

The productID element states a vendor specific unique identification for the device type described.

6.4.2.9 productText

The productText element allows the vendor to provide a short textual description of the device type.

6.4.2.10 orderNumber

The orderNumber element is used to store the single order number of a given device or the set of different order numbers of the products of a device family, depending upon whether the device profile describes a device or a device family.

6.4.2.11 version

The version element is used to store different types of version information. Multiple version elements are possible.

The version element has the attributes given in Table 7.

Table 7 — Attributes of element version

Attribute	Data type	Use	Description
versionType	xsd:NMTOKEN	required	Type of version: — SW – Software — FW – Firmware — HW – Hardware
readOnly	xsd:boolean	default	Indicates whether the value is read-only for a user: false, true (default)

6.4.2.12 buildDate

The buildDate element specifies the build date of the software unit.

6.4.2.13 specificationRevision

The specificationRevision element contains the revision of the specification, to which this device conforms.

6.4.2.14 instanceName

This element contains the instance name of the device.

6.4.3 DeviceManager

6.4.3.1 General

The DeviceManager element defines the list of indicators provided by the device type, if any, and an interface to the module management, if the device supports connected modules and as such qualifies as a modular device.

6.4.3.2 indicatorList / LEDList

6.4.3.2.1 General

Figure 11 specifies the number and type of indicators, which are provided by a device type.

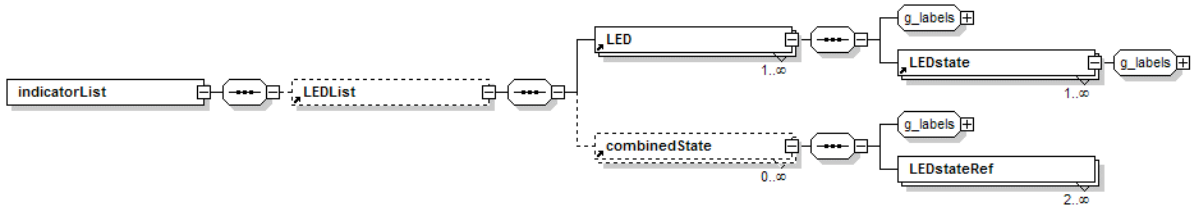

Figure 11 — indicatorList / LEDList

6.4.3.2.2 LED

The LED element describes the features of a single LED of the device type. A detailed feature description may be provided through the g_labels group.

Further properties of the LED are represented as attributes of the LED element given in Table 8.

Table 8 — Attributes of element LED

Attribute	Data type	Use	Description
LEDcolors	xsd:string	required	Colours of the LED; valid values are monocolor and bicolor
LEDtype	xsd:string	optional	Rough classification of the supervised item or functionality; valid values are IO, device and communication

In addition to the descriptive parts introduced above, the LED element contains one to many LEDstate elements, which define the device states signalled by the LED and the visual behaviour used for signalling the states.

The visual behaviour used for signalling the state is encoded as attribute values of the LEDstate element, as given in Table 9. Additionally a unique ID is allocated for the LED state.

Table 9 — Attributes of element LEDstate

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID for the LED state; may be referenced from an LEDstateRef element
state	xsd:string	required	State of the LED; possible attribute values: on, off, flashing
LEDcolor	xsd:string	required	Colours of the LED; valid values: green, red, amber
flashingPeriod	xsd:unsignedInt	optional	If state is flashing: flashing period of the LED in milliseconds
impulsWidth	xsd:unsignedByte	default	Width of the flashing impulse given in percent (%) of the flashing period; if the attribute impulsWidth is missing; the default value is 50 (%)
numberOfImpulses	xsd:unsignedByte	default	Number of impulses in case that more than one flashing impulse is inside one flashing period; if the attribute is present, the attribute impulsWidth shall be present also; if the attribute numberOfImpulses is missing, the default value is 1.

6.4.3.2.3 combinedState

The combinedState element allows the indication of device states which are signaled by more than one LED.

The description of the combined state is provided through the g_labels group.

The LED states participating in the signalling of the combined state are referenced by means of at least two LEDstateRef sub-elements of the combinedState element.

The reference to a LEDstate element is encoded as the attribute value of the single attribute of the LEDstateRef element (see Table 10).

Table 10 — Attribute of element LEDstateRef

Attribute	Data type	Use	Description
stateIDRef	xsd:IDREF	required	Unique ID of the referenced LEDstate element

6.4.3.3 moduleManagement

6.4.3.3.1 General

The optional moduleManagement element, if present, contains a choice of either a moduleInterfaceList element or a single moduleInterface element as shown in Figure 12.

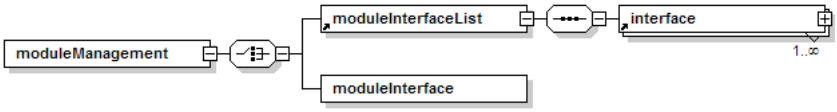

Figure 12 — moduleManagement

The moduleInterfaceList element (see 6.4.3.3.2) shall be present if the device implements the base of a modular device.

The single moduleInterface element shall be present if the device implements a single module. The single moduleInterface element shall contain the attributes given in Table 11.

Table 11 — Attributes of element moduleInterface

Attribute	Data type	Use	Description
childID	xsd:NCName	required	Unique ID of the module (see NOTE)
type	xsd:NCName	required	Type of the module
maxCount	xsd:nonNegativeInteger	optional	Maximum number of multiples of this module to be connected to the modular device. The value 0 shall indicate no limitation.
NOTE To guarantee uniqueness this ID consists of the 8 hexadecimal characters of the vendor-ID and a manufacturer specific part.			

6.4.3.3.2 moduleInterfaceList

6.4.3.3.2.1 General

The moduleInterfaceList element, if present, contains a sequence of one to many interface elements.

6.4.3.3.2.2 interface

6.4.3.3.2.2.1 General

The mandatory interface element (see Figure 13) is used to indicate the properties of one interface provided by a modular device. These properties include a file path (see 6.4.3.3.2.2.2) to the XML device description files defining the modules to be connected to the modular device, the module types (see 6.4.3.3.2.2.3) of the modules to be connected to the modular device, and may be the actual connected modules (see 6.4.3.3.2.2.4).

The interface element may contain the element group g_labels with one or more subelements and contains the attributes defined in Table 17.

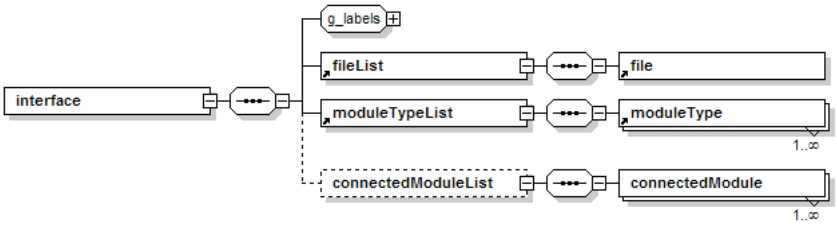

Figure 13 — interface

Table 12 — Attributes of element interface

Attribute	Data type	Use	Description
unique	xsd:ID	required	Unique ID of the interface
maxChilds	xsd:positiveInteger	required	Maxmimum number of modules to be connected
unusedSlots	xsd:Boolean	required	Empty slots allowed; valid values: — true — false
multipleChilds	xsd:Boolean	required	Multiple modules of the same type allowed; valid values: — true — false

6.4.3.3.2.2.2 fileList / File

The mandatory fileList element contains a sequence of one to many file elements.

The mandatory file element contains a file path referenced by the URI attribute (see Table 13) attribute to search for additional XML device description files containing definitions of the modules.

Table 13 — Attributes of element file

Attribute	Data type	Use	Description
URI	xsd:anyURI	required	File path to XML device description files (see NOTE)
NOTE The file path may contain wildcard characters. The wildcard character “?” would be recognized as a single wildcard character. The wildcard character “*” would be recognized as one to many wildcard characters.			

6.4.3.3.2.2.3 moduleTypeList / moduleType

The mandatory moduleTypeList element contains a sequence of one to many moduleType elements.

The mandatory moduleType element contains a reference to a module type of a module allowed to be conntected to the modular device. The according module type is defined in the additional XML device description files referenced by the fileList element (see 6.4.3.3.2.2.2). The module type is defined by the type attribute (see Table 14).

Table 14 — Attributes of element moduleType

Attribute	Data type	Use	Description
type	xsd:NCName	required	Reference to an allowed module type

6.4.3.3.2.2.4 connectedModuleList / connectedModule

The optional connectedModuleList element contains a sequence of one to many connectedModule elements. The connectedModuleList element is used for configured modular devices.

The mandatory connectedModule element contains the childIDRef attribute that is a reference to a connected module. The attributes of the connectedModule element are defined in Table 15.

Table 15 — Attributes of element connectedModule

Attribute	Data type	Use	Description
childIDRef	xsd:NCName	required	Reference to an allowed module type

6.4.4 DeviceFunction

6.4.4.1 General

The DeviceFunction element defines the catalogue view of the device, presented as a set of capabilities listing both device characteristics and compliance with various standards.

The sub-elements of the DeviceFunction element shown in Figure 5 are detailed in the following subclauses.

6.4.4.2 capabilities

6.4.4.2.1 General

The mandatory capabilities element describes all functionalities, their characteristics, and the important parameters of the device, that need to be known by tools which use the device profile to select products with the same or similar properties.

The capabilities element describes device features in a purely textual form. It contains a sequence of one to many characteristicsList elements and an optional standardComplianceList element.

6.4.4.2.2 characteristicsList

6.4.4.2.2.1 General

The characteristicsList element is a collection of characteristics. The element shall contain at least one characteristic sub-element. The characteristics inside a list may be associated with a category, which can be expressed as textual content of the g_labels sub-element of the optional category sub-element of the characteristicsList element.

6.4.4.2.2.2 characteristic

The characteristic element describes a single characteristic of a device. It contains a mandatory characteristicName element and one to many characteristicContent elements.

6.4.4.2.2.3 characteristicName

The mandatory characteristicName element denotes a major technical characteristic of the device. The vocabulary used in the device data sheet is recommended for the names of characteristics.

EXAMPLES

"Maximum operational voltage", "Overload protection", "Electrical durability".

6.4.4.2.2.4 characteristicContent

This mandatory element contains a value for the characteristic. Multiple values may be expressed by using multiple characteristicContent elements.

EXAMPLE

An example of a single value for "Maximum operational voltage" is 680V.

6.4.4.2.3 standardComplianceList

The standardComplianceList element is a collection of compliantWith elements. The element itself is optional; if it exists, it shall contain at least one compliantWith sub-element.

The compliantWith sub-element has attributes, which state the compliance of the device with an international or company internal standard. The content of type g_labels of this element may contain remarks concerning that standard.

The name or number of the standard is provided through the required name attribute of the compliantWith element. The second, default valued range attribute of the compliantWith element defines the range of applicability of the standard as given in Table 16.

Table 16 — Attributes of element compliantWith

Attribute	Data type	Use	Description
name	xsd:string	required	Name or number of the standard
range	xsd:NMTOKEN	default	The two possible enumerated values of the attribute are international (default) or internal

6.4.4.3 picturesList

The picturesList element offers the possibility to link pictures to the device profile. It contains one to many picture sub-elements, whose caption is provided via a g_labels sub-element.

Table 17 defines attributes of the picture sub-element: an optional number of the picture, and the mandatory link to an external resource containing the graphical information.

Table 17 — Attributes of element picture

Attribute	Data type	Use	Description
URI	xsd:anyURI	required	Link to the external resource
number	xsd:unsignedInt	optional	Number of the picture

6.4.4.4 dictionaryList

The optional dictionaryList element offers the possibility to include links to external text resource files to the device profile. It contains one to many dictionary elements, where each one contains one file sub-element.

A mandatory lang attribute of type xsd:language defines the language used in the file which is linked to the dictionary element (see Table 18). A mandatory dictID attribute of type xsd:string holds the unique identification of the dictionary which is referenced from the attribute dictID of a labelRef element as given in Table 3.

Table 18 — Attributes of element dictionary

Attribute	Data type	Use	Description
lang	xsd:language	required	Language used for the file belonging to a dictionary
dictID	xsd:string	required	Identification of the dictionary

A file sub-element contains a single mandatory attribute given in Table 19.

Table 19 — Attribute of element file

Attribute	Data type	Use	Description
URI	xsd:anyURI	required	Link to the respective file

6.4.5 ApplicationProcess

6.4.5.1 General

The ApplicationProcess element represents the set of services and parameters, which constitute the behavior, and the interfaces of the device in terms of the application, independent of the device technology and the underlying communication networks and communication protocols.

The sub-elements of the ApplicationProcess element in Figure 6 provide a generic approach for the description of arbitrary, flat or hierarchically structured functions of a device.

Functions are modeled as function types, which are instantiated within the device or - if hierarchical structures are needed - inside function types. The interface variables of these function instances, which may be of simple or complex data type, are associated with the parameters of the device by building a reference from the parameter to the respective interface variable of the function instance, in flat as well as in hierarchical structures.

The ApplicationProcess element contains up to six lists of items (see Figure 6):

— two optional lists which define data types and function types;

— one optional list which defines the function instances on device level (possibly including connections between instances);

— one optional list which defines templates for sub-elements of device parameters or for sub-elements of allowed device parameter values;

— one required list which defines the device parameters and

— one optional list which defines parameter groups (combinations of parameters for specific purposes).

The elements inside these lists are described in the following sub clauses.

6.4.5.2 dataTypeList

6.4.5.2.1 General

The optional dataTypeList element is present if complex data types like arrays or data structures are needed inside variable declarations or parameter specifications of the device profile.

If present, the dataTypeList element shown in Figure 14 contains a sequence of one to many elements out of the choice of:

— an array element,

— a struct element,

— an enum element, or

— a derived element.

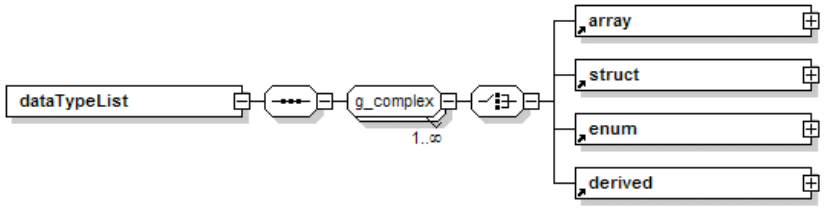

Figure 14 — dataTypeList

6.4.5.2.2 Common elements

6.4.5.2.2.1 Group g_simple

The group g_simple contains a choice of elements, whose names represent the names of all simple data types allowed in the definition of variables or parameters inside a device profile. The simple data types conform to the elementary data types defined in /IEC61131-3/; the data types BITSTRING and CHAR (=STRING[1]) are added.

These elements are introduced inside a group to allow their placement directly as a sub-element of the array element (or of the varDeclaration element, see 6.4.5.2.4.2, or other elements).

6.4.5.2.3 array

6.4.5.2.3.1 General

The array element (see Figure 15) serves to describe an array data type, which may be referenced from an interface variable of a function type, from another array type definition, from a component variable inside the definition of a structured data type, or from a parameter specification.

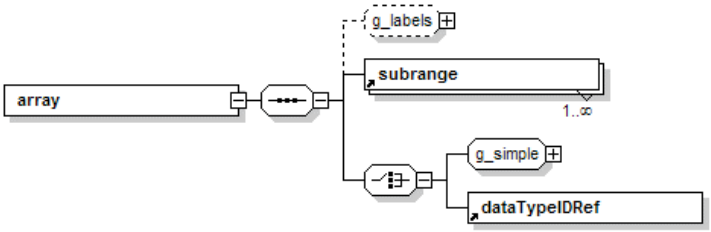

Figure 15 — array

The array element may contain the element group g_labels with one or more subelements. The array element contains at least one subrange element and either an element describing a simple data type out of the group g_simple, or an element dataTypeIDRef, which references one of the defined complex data types within the dataTypeList element.

For multi-dimensional arrays, several subrange elements will be present. In this case, the first subrange element in the sequence defines the subrange for the leftmost array index, and the last subrange element in the sequence defines the subrange for the rightmost array index.

The array element contains the attributes given in Table 20.

Table 20 — Attributes of element array

Attribute	Data type	Use	Description
name	xsd:string	required	Data type name of the array type
uniqueID	xsd:ID	required	Unique ID of the array type

6.4.5.2.3.2 subrange

The subrange element defines the lower and the upper limit of an array index for one dimension of the array. This element has no sub-elements. The limit values of type xsd:long are contained in the two attributes of the subrange element given in Table 21.

Table 21 — Attributes of element subRange

Attribute	Data type	Use	Description
lowerLimit	xsd:long	required	Lower limit of the subrange
upperLimit	xsd:long	required	Upper limit of the subrange

6.4.5.2.4 struct

6.4.5.2.4.1 General

The struct element (see Figure 16) serves to describe a structured data type, which may be referenced from an interface variable of a function type, from an array type definition, from a component variable inside the definition of another structured data type, or from a parameter specification.

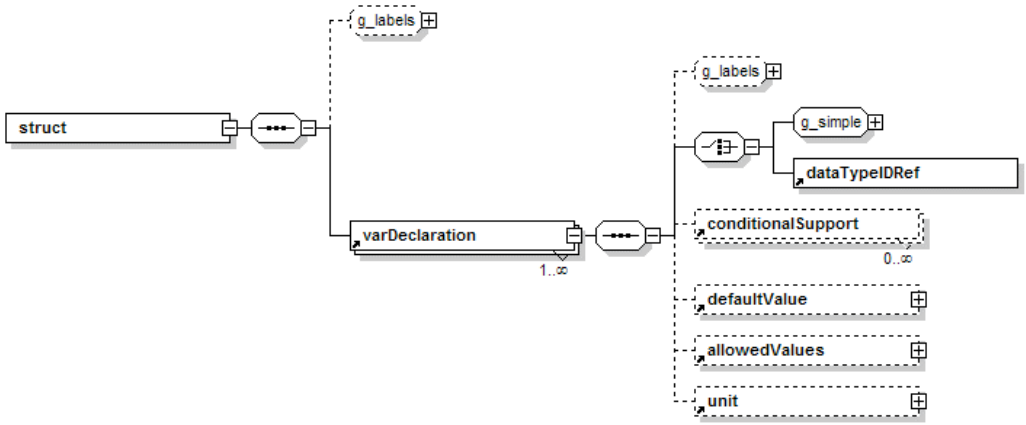

Figure 16 — struct

The struct element may contain the element group g_labels with one or more subelements. The struct element contains a sequence of one to many varDeclaration elements, which define the components of the structured data type.

The struct element shall contain the attributes given in Table 22.

Table 22 — Attributes of element struct

Attribute	Data type	Use	Description
name	xsd:string	required	Data type name of the structured data type
uniqueID	xsd:ID	required	Unique ID of the structured data type

6.4.5.2.4.2 varDeclaration

In the context of the definition of a structured data type, the varDeclaration element describes a single component variable (member) of the structure.

In the context of the definition of the interface of a function, the varDeclaration element describes a single interface variable of the function type.

The varDeclaration element may contain the element group g_labels with one or more subelements.

The data type of the component variable or interface variable is either defined by an element describing a simple data type out of the group g_simple, or by an element dataTypeIDRef, which references one of the defined complex data types within the dataTypeList element.

The varDeclaration element may contain the element conditionalSupport (see 6.4.5.7.2.2), the element defaultValue (see 6.4.5.7.2.5), the element allowedValues (see 6.4.5.7.2.7), and the element unit (see 6.4.5.7.2.8).

All further properties of the variable are contained in the attributes of the varDeclaration element, as given in Table 23.

Table 23 — Attributes of element varDeclaration

Attribute	Data type	Use	Description
name	xsd:string	required	Name of the interface variable or structure component
uniqueID	xsd:ID	required	Unique ID of the interface variable or structure component (see NOTE 1)
start	xsd:string	optional	First element, if the interface variable or structure component is of type anonymous ARRAY, BITSTRING, STRING or WSTRING (see NOTE 2)
size	xsd:string	optional	Number of elements, if the interface variable or structure component is of type anonymous ARRAY, BITSTRING, STRING or WSTRING (see NOTE 2)
signed	xsd:boolean	optional	Interpretaion as a signed values; valid values: — true — false (default)
offset	xsd:string	optional	Offset which is added to the value to form a scaled value: EngineeringValue = (value + offset) * multiplier; if not present, offset = 0 is assumed
multiplier	xsd:string	optional	Scaling factor by which the value is multiplied to form a scaled value: EngineeringValue = (value + offset) * multiplier; if not present, multiplier = 1 is assumed
initialValue	xsd:string	optional	Initial value of the interface variable or structure component (see NOTE 3)

NOTE 1 When creating the unique ID for a variable, it is essential that the ID is unique over all created IDs inside the XML source file. To allow equal names for component variables of different data structures and equal names for interface variables of function types, the ID of a variable should generally concatenate the type name of the structured data type or the function type with the variable name, to guarantee uniqueness.

NOTE 2 Anonymous types define the size of an array, bitstring or string directly in the variable declaration, and not through the reference to a named complex data type. In the case of an array, the data type of the variable gives the type of a single array element. In the case of a bitstring, the single array element is a single bit. In the case of a string, the single array element is a single-byte resp. double-byte character.

NOTE 3 If present, this attribute defines the initial (default) value of the interface variable of the function type. It is overwritten by a given default value of a parameter associated with the interface variable of the function instance.

6.4.5.2.5 enum

6.4.5.2.5.1 General

The enum element serves to describe an enumerated data type, which may be referenced from an interface variable of a function type, from an array type definition, from a component variable inside the definition of a structured data type, or from a parameter specification.

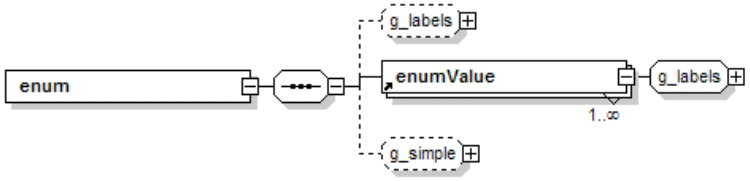

Figure 17 — enum

According to Figure 17, it contains a sequence. The sequence may contain the element group g_labels with one or more subelements, and one to many enumValue elements, which define the enumeration constants of the enumerated data type. The data type of the enumeration constants is optionally defined by an element describing a simple data type out of the group g_simple.

The enum element contains the attributes given in Table 24.

Table 24 — Attributes of element enum

Attribute	Data type	Use	Description
name	xsd:string	required	Type name of the enumerated data type
uniqueID	xsd:ID	required	Unique ID of the enumerated data type
size	xsd:string	optional	Optional number of enumerated values of the enumerated data type

6.4.5.2.5.2 enumValue

The enumValue element defines the name(s) and optionally a numerical value of a single enumeration constant. The name(s) are specified through the g_labels group, whereas the value is contained in the single value attribute of the enumValue element, as given in Table 25.

Table 25 — Attribute of element enumValue

Attribute	Data type	Use	Description
value	xsd:string	optional	Optional attribute: fixed numerical value for the enumeration constant, represented as a string of characters

6.4.5.2.6 derived

6.4.5.2.6.1 General

The derived element (see Figure 18) serves to derive a new data type from a given base type.

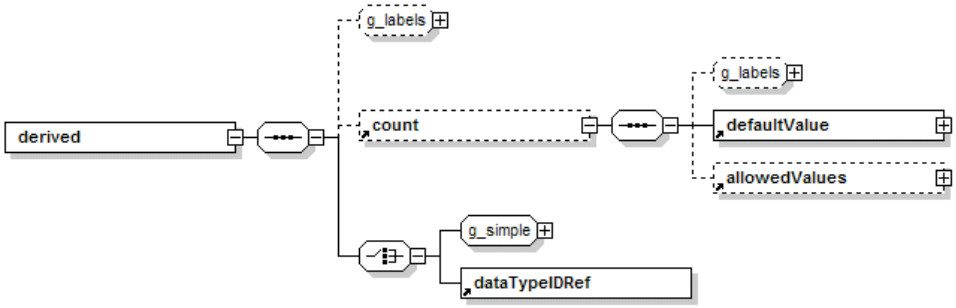

Figure 18 — derived

The derived element may contain the element group g_labels with one or more subelements, an optional count element and either an element describing a simple data type out of the group g_simple, or an element dataTypeIDRef, which references one of the defined complex data types within the dataTypeList element.

If the count element is missing, the derived type definition just introduces a new type name for the respective base type. If the count element is present, it defines the number of units of the respective base type used to build the derived type (e.g base type BITSTRING, count = 4 defines a derived type of size 4 bit).

The derived element contains the attributes given in Table 26.

Table 26 — Attributes of element derived

Attribute	Data type	Use	Description
name	xsd:string	required	Data type name of the derived type
uniqueID	xsd:ID	required	Unique ID of the derived type

6.4.5.2.6.2 count

The count element defines the number of used units of the base type of the derived type (see 6.4.5.2.6). Multilingual names and/or descriptions for the count element are provided through the group g_labels.

The count is described by:

— its attributes,

— the mandatory sub-element defaultValue and a possibly empty set of sub-elements g_labels and allowedValues.

The number of units is expressed as the value of the defaultValue attribute of the count element. The allowedValue attribute defines the range of values for the default value.

The sub-elements defaultValue and allowedValues are described in 6.4.5.7.2.5 and in 6.4.5.7.2.7.

The count element shall contain the attributes given in Table 27.

Table 27 — Attributes of element count

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the count
access	xsd:NMTOKEN	default	Defines which operations are valid for the count: — const – read access only; value is not changing — read – read access only (default value) — write – write access only — readWrite – both read and write access — noAccess – access denied

6.4.5.3 functionTypeList

If the optional ApplicationProcess element is present in the device profile, it may contain an optional functionTypeList element shown in Figure 19.

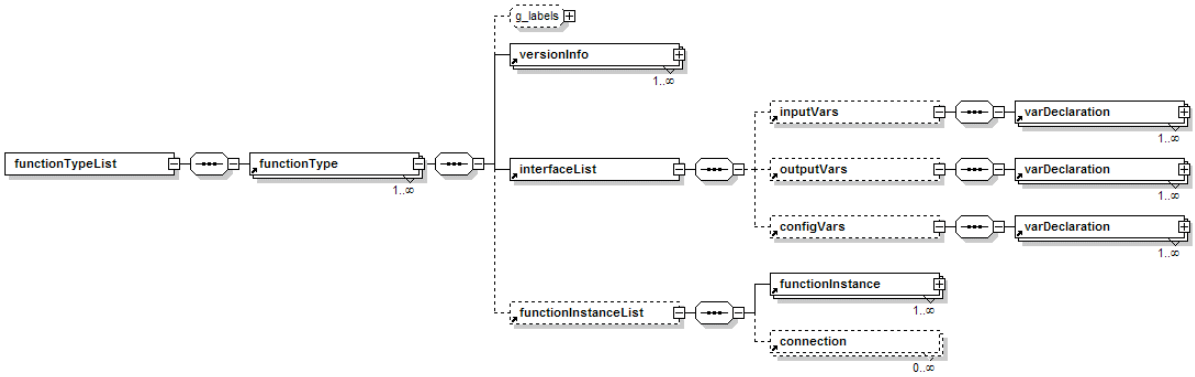

Figure 19 — functionTypeList

The functionTypeList represents a sequence of one to many functionType elements.

Each of the functionType elements represents the type description of a device function, which is referenced from at least one instance of that function type inside a functionInstanceList element. References from more than one instance of the same function type are also possible.

The description of a function type contains all those objects and data which are common for all instances of a given function type.

EXAMPLE 1

Examples are the variable - or function parameter - objects that constitute the interface of the function (type respectively instance).

EXAMPLE 2

Other examples are instances contained within the body of a function in a hierarchically structured functional description. These instances, which are located within a functionInstanceList element inside the function type, reference other function types in the list of function types.

6.4.5.4 functionType

6.4.5.4.1 General

The functionType element may contain the element group g_labels with one or more subelements, and one to many versionInfo elements, a mandatory interfaceList element and an optional functionInstanceList element. The functionInstanceList element is only present within a functionType element, if the function is hierarchically structured.

The versionInfo element and the interfaceList element are described in the following subclauses.

Additionally, the functionType element shall contain the attributes given in Table 28.

Table 28 — Attributes of element functionType

Attribute	Data type	Use	Description
name	xsd:string	required	Type name of the function type
uniqueID	xsd:ID	required	Unique ID of the function type
package	xsd:string	optional	Optional textual association of the function type with a "package" or similar classification scheme - the usage of this attribute is left to the profile validator

6.4.5.4.2 versionInfo

The mandatory versionInfo element within the functionType element provides information on the versioning history of a function type (concerning the definition of the interface).

To keep track of the versioning history, the versionInfo element may be entered multiple times. The multiple entries shall be arranged within the functionType element in the following sequence:

a) the first entry represents the most recent version,

b) the second entry represents the immediately preceding version,

c) the last entry represents the first released version.

This element will be provided once at the creation of the description of the function type. New elements will only be added, if modifications of a function type are introduced, which lead to a modified version of the device profile.

The versionInfo element may contain one element group g_labels with one or more subelements.

The versionInfo element shall contain the attributes given in Table 29.

Table 29 — Attributes of element versionInfo

Attribute	Data type	Use	Description
organization	xsd:string	required	Name of the organisation maintaining the function type
version	xsd:string	required	Version identification in the versioning history; suggested format: "xx.yy" (xx,yy = 0..255)
author	xsd:string	required	Name of the person maintaining the function type
date	xsd:date	required	Date of this version

6.4.5.4.3 interfaceList

6.4.5.4.3.1 General

The mandatory interfaceList element within the functionType element provides the definition of the interface of the function type. Elements of the interface are:

— the input variables, and/or

— the output variables, and/or

— the configuration variables

of the function type.

Consequently the interfaceList element contains a sequence of three elements, where each element represents lists of one to many variable declarations encoded as varDeclaration elements (see 6.4.5.2.4.2):

— one optional element inputVars,

— one optional element outputVars, and

— one optional element configVars.

Neither the interfaceList nor the inputVars, outputVars or configVars elements have any attributes.

6.4.5.4.3.2 dataTypeIDRef

The dataTypeIDRef element serves to reference a complex data type inside the dataTypeList element (see 6.4.5.2), either from an interface variable of a function type, or from an array type definition, or from a component variable inside the definition of a structured data type.

The reference of type xsd:IDREF is provided as an attribute of the dataTypeIDRef element, as given in Table 30.

Table 30 — Attribute of element dataTypeIDRef

Attribute	Data type	Use	Description
uniqueIDRef	xsd:IDREF	required	Unique ID of the referenced data type

6.4.5.5 functionInstanceList

6.4.5.5.1 General

The optional functionInstanceList element, if present, contains a sequence of one to many functionInstance elements and zero to many connection elements.

At the application process level, the functionInstance elements represent the accessible application functions of the device type, independent of the network type or protocol. The connection elements represent connections - if any - between specific output and input variables of those function instances.

The functionInstanceList element also appears as an optional sub-element of the functionType element (see 6.4.5.4). Like at the application process level, the functionInstanceList element in that case contains a sequence of one to many functionInstance elements and zero to many connection elements.

The functionInstanceList element is only present within a functionType element, if a function is hierarchically structured. In this case the functionInstance elements represent the internal functions contained in the function type, and the connection elements the optional internal

connections. These functions and their optional connections would be instantiated together with the instantiation of the containing function type.

The functionInstanceList element does not have any attributes.

6.4.5.5.2 functionInstance

The mandatory functionInstance element may contain the element group g_labels with one or more subelements.

The functionInstance element shall contain the attributes given in Table 31.

Table 31 — Attributes of element functionInstance

Attribute	Data type	Use	Description
name	xsd:string	required	Name of the function instance
uniqueID	xsd:ID	required	Unique ID of the function instance (see NOTE)
typeIDRef	xsd:IDREF	required	Unique ID of the referenced function type
NOTE When creating the unique ID for a function instance, it is essential that the ID is unique over all created IDs inside the XML source file. To allow equal names for function instances inside different function types, the ID of a function instance should generally concatenate the name of the containing function type with the instance name, to guarantee uniqueness.			

6.4.5.5.3 connection

The optional connection element defines a connection between an output variable of a function instance and an input variable of another function instance. Inside function types, the connection may also be drawn between an input variable of the function type and an input variable of a contained function instance, or between an output variable of a contained function instance and an output variable of the function type. The connection element may appear zero to many times.

The connection element contains the attributes given in Table 32.

Table 32 — Attributes of element connection

Attribute	Data type	Use	Description
source	xsd:string	required	Starting point of connection
destination	xsd:string	required	Endpoint of connection
description	xsd:string	optional	Optional textual description of the connection

EXAMPLE

The values of the source and the destination attributes may be used to encode the starting point and the endpoint of a connection using the syntax <function_instance_name>'.'<variable_name>; example for the value of a source attribute: 'PowerMeasures.Frequency'. Connections to interface variables of a function type use the names of the interface variables only.

6.4.5.6 templateList

The various sub-elements of the optional templateList element are referenced from within parameter elements (see 6.4.5.7.2.1) or allowedValues elements (see 6.4.5.7.2.7), which use sets of sub-elements with the same attribute values in their descriptions. Intense usage of the template constructs may largely reduce the size of device profile XML files.

The templateList element, if present, contains a sequence of zero to many parameterTemplate elements and/or zero to many allowedValuesTemplate elements, as shown in Figure 20.

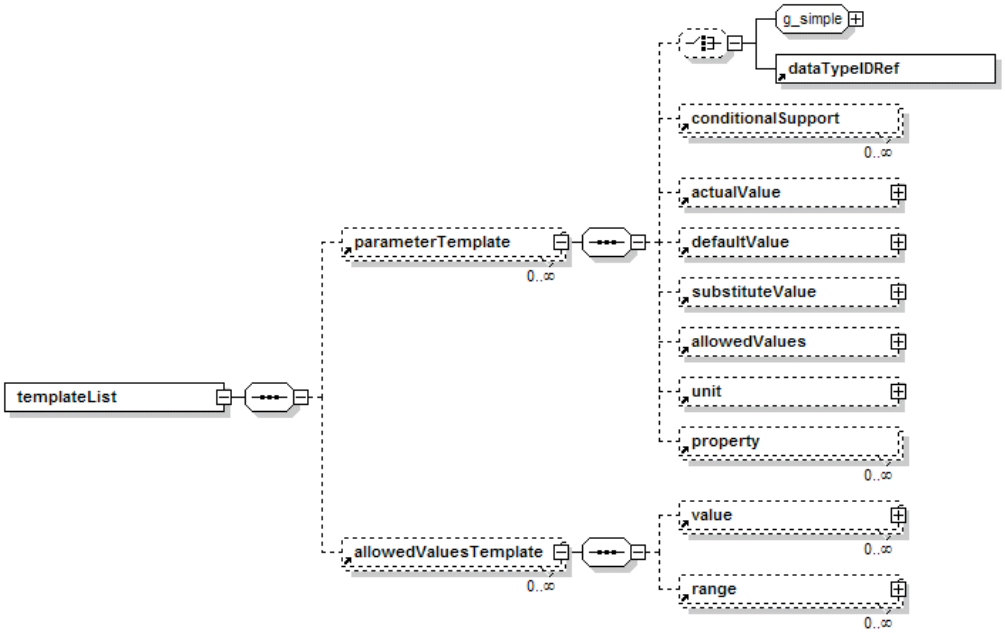

Figure 20 — templateList

The sub-elements of the parameterTemplate element are a subset of the sub-elements of the parameter element, as specified in 6.4.5.7.2, such that the descriptions of these sub-elements in the subclauses of 6.4.5.7.2 hold.

The parameterTemplate element shall contain the same attributes as the parameter element, as given in Table 34.

The sub-elements of the allowedValuesTemplate element are the same as the sub-elements of the allowedValues element, as specified in 6.4.5.7.2.7, such that the descriptions of these sub-elements in the subclauses of 6.4.5.7.2.7 also hold.

The single attribute of the allowedValuesTemplate element is given in Table 33.

Table 33 — Attribute of element allowedValuesTemplate

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the template for allowed values

If a parameter element or an allowedValues element referencing a template contains the same sub-element as the referenced template element, the value of the sub-element of the parameter element or of the allowedValues element overrides the value provided by the template.

6.4.5.7 parameterList

6.4.5.7.1 General

If the optional ApplicationProcess element is present in the device profile, it contains a mandatory parameterList element shown in Figure 21, which represents a sequence of one to many parameter elements.

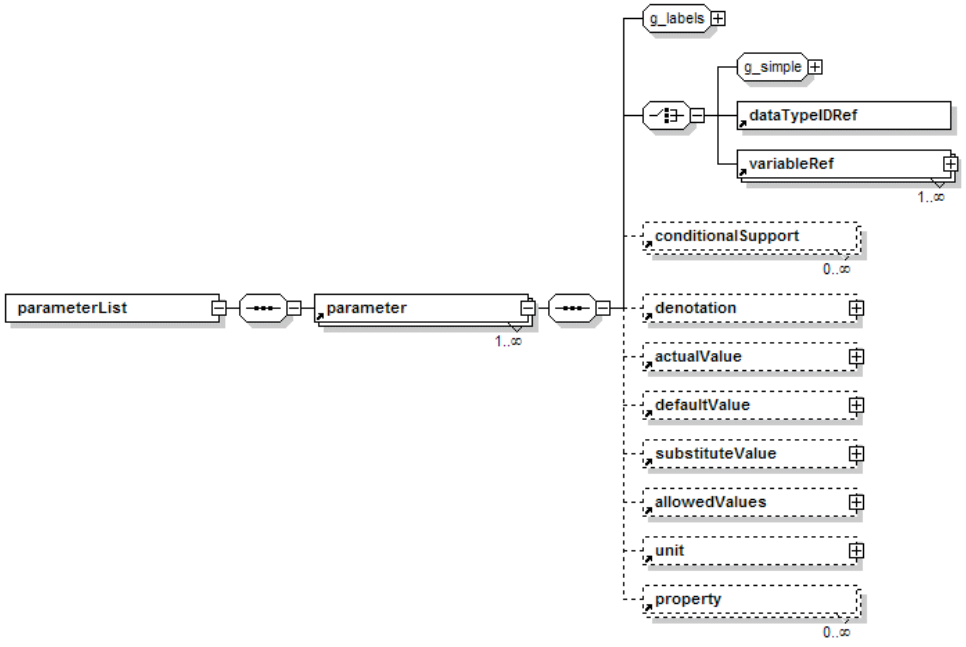

Figure 21 — parameterList

Each of the parameter elements represents a parameter of the device profile. Multilingual names and/or descriptions for the parameters are provided through the group g_labels.

The parameter is described by:

— its name(s) and description(s) (element g_labels),

— its attributes,

— a choice of three possible references:

  — a reference to a simple data type (element g_simple - see NOTE 1),

  — a reference to a complex data type (element dataTypeIDRef - see NOTE 1),

  — or a reference to one (or more) interface variable(s) of one (or more) function instance(s) (element variableRef - see NOTE 1 and NOTE 2),

— and a possibly empty set of sub-elements (conditionalSupport, denotation, actualValue, defaultValue, substituteValue, allowedValues, unit and property).

NOTE 1 All parameter elements of a given device profile shall either use references to data types only, or references to interface variables only.

NOTE 2 References to multiple variables are a special case: specific parameters may reference an output variable of one function instance and an input variable of another function instance at the same time. In this case the data types of the two variables shall be the same. The XML parser cannot check the equality of data types, this can only be checked by a supporting tool.

6.4.5.7.2 parameter

6.4.5.7.2.1 General

The parameter element shall contain the attributes given in Table 34.

NOTE The same attributes are also valid for the parameterTemplate element, as specified in 6.4.5.6.

Table 34 — Attributes of element parameter

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the parameter
access	xsd:NMTOKEN	default	Defines which operations are valid for the parameter: — const – read access only; the value is not changing — read – read access only (default value) — write – write access only — readWrite – both read and write access — readWriteInput – both read and write access, but represents process input data — readWriteOutput – both read and write access, but represents process output data — noAccess – access denied
accessList	xsd:NMTOKENS	optional	Defines a list of additionally allowed operations on the parameter: the valid values are the same as for the attribute access, but the parser will not check the usage of the correct values.
support	xsd:NMTOKEN	optional	Defines whether or not the parameter has to be implemented in the device; valid values: — mandatory – parameter implementation is required — optional – parameter implementation is possible but not required — conditional – parameter implementation is required if one or more other optional parameter(s) is (are) implemented; these parameters are specified using the sub-element conditionalSupport
persistent	xsd:boolean	default	Defines the behavior after power failure; valid values are false (default) and true
offset	xsd:string	optional	Offset which is added to an actual value to form a scaled value: EngineeringValue = (ParameterValue + offset) * multiplier; if not present, offset = 0 is assumed
multiplier	xsd:string	optional	Scaling factor by which an actual value is multiplied to form a scaled value: EngineeringValue = (ParameterValue + offset) * multiplier; if not present, multiplier = 1 is assumed
templateIDRef	xsd:IDREF	optional	Unique reference to a parameterTemplate element

6.4.5.7.2.2 conditionalSupport

One or more conditionalSupport elements are present only if the value of the support attribute of the parameter element is conditional. Each element refers to a single optional parameter. If at least one of those optional parameters is implemented, the conditional parameter has also to be implemented.

The element conditionalSupport shall contain the single attribute given in Table 35.

Table 35 — Attribute of element conditionalSupport

Attribute	Data type	Use	Description
paramIDRef	xsd:IDREF	required	Unique ID of the referenced optional parameter

6.4.5.7.2.3 denotation

The denotation element serves to hold application-specific, multilingual names of the parameter. The names are provided through the mandatory g_labels sub-element. It is also possible to add multilingual descriptive information. The element denotation does not have any attributes.

6.4.5.7.2.4 actualValue

The actualValue element serves to hold the actual value of the parameter. An optional g_labels sub-element may provide multilingual descriptive information for this value. The value itself is provided in the value attribute of the element actualValue. An offset and multiplier may also be provided.

The attributes of the element actualValue shall be as given in Table 36.

Table 36 — Attributes of element actualValue

Attribute	Data type	Use	Description
value	xsd:string	required	Actual value
offset	xsd:string	optional	Offset which is added to an actual value to form a scaled value: EngineeringValue = (value + offset) * multiplier; if not present, the respective value of the parameter element shall be used
multiplier	xsd:string	optional	Scaling factor by which an actual value is multiplied to form a scaled value: EngineeringValue = (value + offset) * multiplier; if not present, the respective value of the parameter element shall be used

6.4.5.7.2.5 defaultValue

The defaultValue element serves to hold the default value of the parameter. This value overwrites the default value, if any, of the interface variable of the function type associated with the parameter.

An optional g_labels sub-element may provide multilingual names and/or descriptive information for this value. The value itself is provided by the value attribute of the element defaultValue. An offset and multiplier may also be provided.

The attributes of the element defaultValue shall be as given in Table 36.

6.4.5.7.2.6 substituteValue

The substituteValue element defines a specific value of the parameter that is provided to the device application in certain device operating states (e.g. on device fault).

An optional g_labels sub-element may provide multilingual names and/or descriptive information for this value. The value itself is provided by the value attribute of the element substituteValue. An offset and multiplier may also be provided.

The attributes of the element substituteValue shall be as given in Table 36.

6.4.5.7.2.7 allowedValues

The allowedValues element defines a list of supported values and/or a single range or several ranges of supported values for the parameter (see Figure 21).

The list of supported values is represented by zero to many value sub-elements of the allowedValues element, whereas the ranges are represented by zero to many range sub-elements of the allowedValues element.

The value sub-element holds a single allowed value of the parameter. An optional g_labels sub-element may provide multilingual names and/or descriptive information for this value. The value itself is provided by the value attribute of the element value. An offset and multiplier may also be provided. The attributes of the element value shall be as given in Table 36.

The range sub-element contains two required sub-elements, namely the element minValue and the element maxValue, which define the limits of that range of allowed values. The optional step sub-element may be used to define equidistant intermediate values inside a range. The minValue, maxValue and step elements have the same structure and attributes as the value sub-element of the allowedValues element. Therefore the description of the value sub-element and Table 36 are also valid for these sub-elements.

6.4.5.7.2.8 unit

The unit element defines the engineering unit of a parameter (e.g. time, temperature, pressure, flow, acceleration, current, energy), as specified in /ISO1000/. An optional g_labels sub-element may provide multilingual names and/or descriptive information for the engineering unit.

The attributes of the element unit shall be as given in Table 37.

Table 37 — Attributes of element unit

Attribute	Data type	Use	Description
multiplier	xsd:string	required	Multiplier for engineering units of analog parameters
unitURI	xsd:anyURI	optional	Link to the respective unit definition in a file containing all engineering units (time, temperature, pressure, flow, acceleration, current, energy…), as standardized by /ISO1000/

6.4.5.7.2.9 variableRef

The variableRef element builds a reference to an interface variable of a function instance, or, if the variable is an array or a structure, possibly a reference to a member of the variable (array element or structure component).

In a hierarchically structured ApplicationProcess element, function instances can be located inside function instances of other function types. Therefore stepping through the tree can only access a specific instance in the functional tree, i.e. the specific instance shall be addressed

through a concatenation of instance names. To map this concatenation and allow the referencing of a member, the variableRef element contains:

— a sequence of one to many instanceIDRef elements, followed by

— a single mandatory variableIDRef element, and

— an optional memberRef element.

The variableRef element has the attribute given in Table 38.

Table 38 — Attribute of element variableRef

Attribute	Data type	Use	Description
position	xsd:unsignedByte	default	Defines the sequence of multiple mapped data items into a single parameter object; position=1 means starting the mapping at the lowest bit position; the number of bits is defined by the data type of the data item; subsequent data items are packed without gaps; default value: 1 (see NOTE)

NOTE Attribute can be omitted for a single mapped data item.

6.4.5.7.2.10 instanceIDRef

The instanceIDRef element serves to reference a function instance inside a functionInstanceList element, which may reside either on the level of the ApplicationProcess element or on the level of a functionType element.

The reference of type xsd:IDREF is provided as an attribute of the instanceIDRef element, as given in Table 39.

Table 39 — Attribute of element instanceIDRef

Attribute	Data type	Use	Description
uniqueIDRef	xsd:IDREF	required	Unique ID of the referenced function instance

6.4.5.7.2.11 variableIDRef

The variableIDRef element serves to reference an interface variable of a function type inside the functionTypeList element.

In a given variableRef element the instance of that function type is defined by the functionInstance element referenced by the instanceIDRef element, which is just preceding the variableIDRef element.

The reference of type xsd:IDREF is provided as an attribute of the variableIDRef element, as given in Table 40.

Table 40 — Attribute of element variableIDRef

Attribute	Data type	Use	Description
uniqueIDRef	xsd:IDREF	required	Unique ID of the referenced interface variable of a function type

6.4.5.7.2.12 memberRef

The optional memberRef element either references the respective component of an interface variable of structured data type (attribute uniqueIDRef is used), or the respective array element of an interface variable of array data type (attribute index is used). One of these two attributes shall be present if the memberRef element is present.

The memberRef element shall contain the attributes given in Table 41.

Table 41 — Attributes of element memberRef

Attribute	Data type	Use	Description
uniqueIDRef	xsd:IDREF	optional	Unique ID of the referenced component of a structured data type
index	xsd:long	optional	Index of the referenced array element

6.4.5.7.2.13 property

The property element is introduced as a generic element to allow the inclusion of values for additional specific properties into the description of a parameter.

The property element shall contain the attributes given in Table 42.

Table 42 — Attributes of element property

Attribute	Data type	Use	Description
name	xsd:string	required	Name of the property
value	xsd:string	required	Value of the property

6.4.5.8 parameterGroupList

6.4.5.8.1 General

The optional parameterGroupList element, if present, contains a sequence of one to many parameterGroup elements, as shown in Figure 22. Multilingual names and/or descriptions for the parameter groups are provided through the group g_labels.

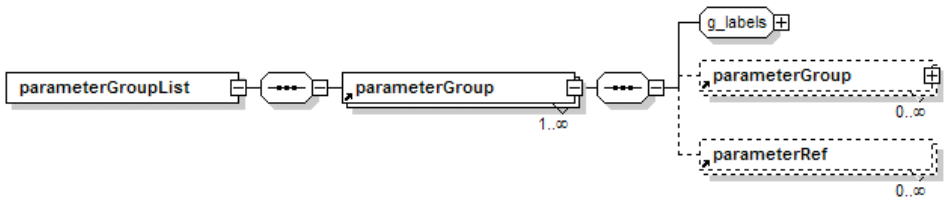

Figure 22 — parameterGroupList

6.4.5.8.2 parameterGroup

Each of the parameterGroup elements combines a set of parameters out of the parameterList element to build a group of parameters which serve a specific purpose, e.g. to prepare HMI views. This purpose is indicated by the value of the kindOfAccess attribute of the parameterGroup element. It is possible to define a hierarchy of parameter groups.

The respective parameters in the set are referenced through the corresponding number of parameterRef elements.

The parameterGroup element contains the attributes given in Table 43.

Table 43 — Attributes of element parameterGroup

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the parameter group
kindOfAccess	xsd:string	optional	Classifies the parameters of the parameter group.

6.4.5.8.3 parameterRef

The parameterRef element serves to reference a parameter element inside the parameterList element of the ApplicationProcess element.

The reference of type xsd:IDREF is provided as an attribute of the parameterRef element, as given in Table 44.

Table 44 — Attribute of element parameterRef

Attribute	Data type	Use	Description
uniqueIDRef	xsd:IDREF	required	Unique ID of the referenced parameter

6.5 Communication network profile template description

6.5.1 ProfileBody_CommunicationNetwork_CANopen

This third part of the device profile template defines the CANopen communication interface.

The ProfileBody_CommunicationNetwork_CANopen element contains the ApplicationLayers, the TransportLayers and the NetworkManagement sub-elements shown in Figure 7. It has the attributes given in Table 45.

Table 45 — Attributes of element ProfileBody

Attribute	Data type	Use	Description
formatName	xsd:string	fixed	Format identifier
formatVersion	xsd:string	fixed	Format version identifier
fileName	xsd:string	required	Name of the file with extension without path
fileCreator	xsd:string	required	Person creating the file
fileCreationDate	xsd:date	required	Date of file creation
fileCreationTime	xsd:time	optional	Time of file creation
fileModifiedBy	xsd:string	optional	Person modifying the file
fileModificationDate	xsd.date	optional	Date of last file change
fileModificationTime	xsd:time	optional	Time of last file change
fileVersion	xsd:string	required	Vendor specific version of the file
supportedLanguages	xsd:NMTOKENS	optional	List of supported languages

6.5.2 ApplicationLayers

6.5.2.1 General

Figure 23 shows the structure of the CANopen ApplicationLayers element.

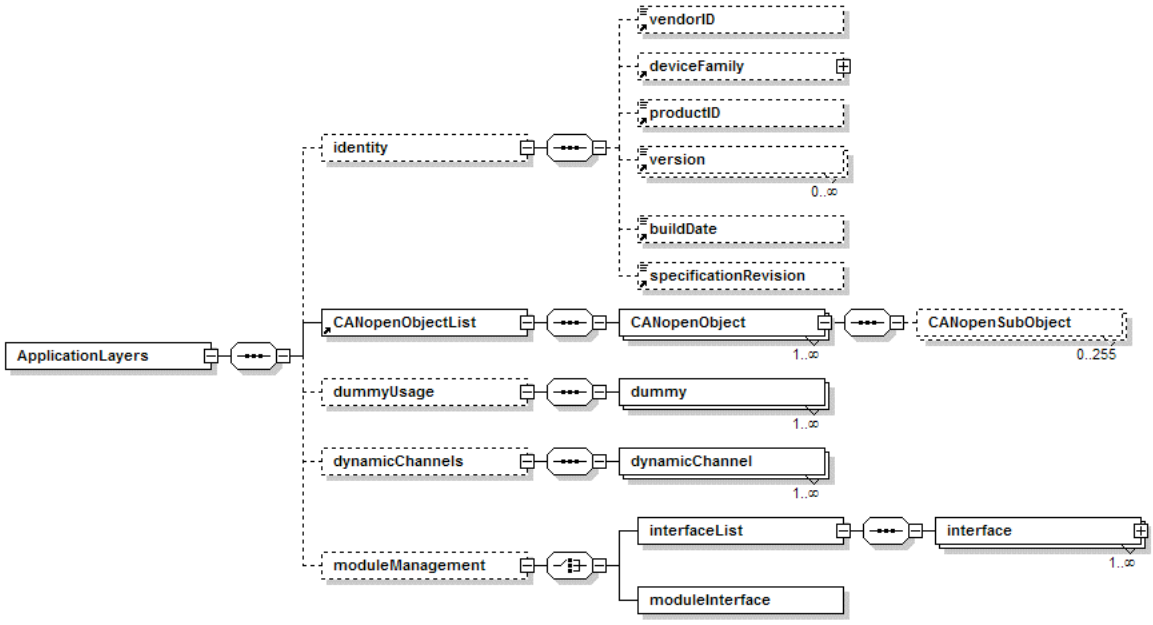

Figure 23 — CANopen ApplicationLayers element

The ApplicationLayers element has the attributes given in Table 46.

Table 46 — Attributes of element ApplicationLayers

Attribute	Data type	Use	Description
conformanceClass	xsd:string	optional	Conformance class of the device type (see NOTE)
communicationEntityType	xsd:NMTOKENS	default	Type of the communication entity - if several types are supported, their names shall be separated by a blank character, e.g. "master slave"; the type names shall be chosen out of the following enumeration of names: — slave (default) — master — server — client — interconnection – e.g. interconnection is gateway functionality. — peer – communication entity which acts as a client and a server
NOTE Using that attribute it is possible to classify the device according to the supported services of the communication protocol.			

6.5.2.2 identity

Since different communication profiles may require different identity information, an optional local identity sub-element may be used within an ApplicationLayers element. This identity element may contain a subset of the sub–elements of the DeviceIdentity element described in 5.1.2. All sub–element descriptions given there also apply for the sub-elements of this identity element.

6.5.2.3 CANopenObjectList

6.5.2.3.1 General

Figure 23 shows the structure of the CANopenObjectList element. The element contains one to many CANopenObject elements.

The CANopenObjectList element contains the optional attributes given in Table 47.

Table 47 — Attributes of element CANopenObjectList

Attribute	Data type	Use	Description
mandatoryObjects	xsd:unsignedInt	optional	Number of mandatory objects in the dictionary
optionalObjects	xsd:unsignedInt	optional	Number of optional objects in the dictionary
manufacturerObjects	xsd:unsignedInt	optional	Number of manufacturer-defined objects in the dictionary

6.5.2.3.2 CANopenObject

6.5.2.3.2.1 General

Figure 23 shows the structure of the CANopenObject element. The element contains zero to many CANopenSubObject elements. The CANopenObject element and the CANopenSubObject element map the functional part of the CANopen device profile to the CANopen communication network profile.

The CANopenObject element contains the attributes given in Table 48.

Table 48 — Attributes of element CANopenObject

Attribute	Data type	Use	Description
index	xsd:hexBinary	required	Index of the object (four hex digits)
name	xsd:string	optional	Name of the object
objectType	xsd:unsignedByte	required	CANopen object type
dataType	xsd:hexBinary	optional	CANopen data type (two hex digits)
lowLimit	xsd:string	optional	Low limit of the parameter value
highLimit	xsd:string	optional	High limit of the parameter value
accessType	xsd:string	optional	Access type of the object; valid values: — const – read access only; the value is not changing — ro – read access only — wo – write access only — rw – both read and write access
defaultValue	xsd:string	optional	Default value of the object
actualValue	xsd:string	optional	Actual value of the object
denotation	xsd:string	optional	Application specific name of the object
PDOmapping	xsd:NMTOKEN	optional	PDO mapping of the object; valid values: — no – not mappable — default – mapped by default — optional – optionally mapped — TPDO – may be mapped into TPDO only — RPDO – may be mapped into RPDO only

Attribute	Data type	Use	Description
objFlags	xsd:hexBinary	optional	Controls the behavior of tools (four hex digits) — bit 0: 0 – write on download allowed 1 – write on download not allowed — bit 1: 0 – read on upload allowed 1 – read on upload not allowed — bit 2: 0 – change of value takes effect immediatly 1 – change of value takes effect after reset — bit 3 to 31: reserved (0) NOTE This shall prevent unintended read or write access by generic tools in case a parameter implements a specific function. A read or write in such cases may trigger a specific function in the CANopen device.
uniqueIDRef	xsd:IDREF	optional	Unique ID of an appropriate element in the application process part referenced from this object; if the attribute uniqueIDRef is present, the attributes name, dataType, lowLimit, highLimit, accessType, defaultValue, actualValue, and denotation shall not be present.
subNumber	xsd:unsignedByte	optional	Number of sub-objects of the object
rangeSelector	xsd:string	optional	Reference of the index range to be used according to the range element

6.5.2.3.2.2 CANopenSubObject

The CANopenSubObject element has an empty content and contains the attributes given in Table 49.

Table 49 — Attributes of element CANopenSubObject

Attribute	Data type	Use	Description
subIndex	xsd:hexBinary	required	Subindex of the sub-object
name	xsd:string	optional	Name of the sub-object
objectType	xsd:unsignedByte	required	CANopen object type
dataType	xsd:hexBinary	optional	CANopen data type (two hex digits)
lowLimit	xsd:string	optional	Low limit of the parameter value
highLimit	xsd:string	optional	High limit of the parameter value
accessType	xsd:string	optional	Access type of the sub-object; valid values: — const – read access only; the value is not changing — ro – read access only — wo – write access only — rw – both read and write access
defaultValue	xsd:string	optional	Default value of the sub-object
actualValue	xsd:string	optional	Actual value of the sub-object
denotation	xsd:string	optional	Application specific name of the sub-object
PDOmapping	xsd:NMTOKEN	optional	PDO mapping of the sub-object; valid values: — no – not mappable — default – mapped by default — optional – optionally mapped — TPDO – may be mapped into TPDO only — RPDO – may be mapped into RPDO only

Attribute	Data type	Use	Description
objFlags	xsd:hexBinary	optional	Controls the behavior of tools (four hex digits) — bit 0: 0 – write on download allowed 1 – write on download not allowed — bit 1: 0 – read on upload allowed 1 – read on upload not allowed — bit 2: 0 – change of value takes effect immediatly 1 – change of value takes effect after reset — bit 3 to 31: reserved (0) NOTE This shall prevent unintended read or write access by generic tools in case a parameter implements a specific function. A read or write in such cases may trigger a specific function in the CANopen device.
uniqueIDRef	xsd:IDREF	optional	Unique ID of an appropriate element in the application process part referenced from this sub-object; if the attribute uniqueIDRef is present, the attributes name, dataType, lowLimit, highLimit, accessType, defaultValue, actualValue and denotation shall not be present.
rangeSelector	xsd:string	optional	Reference of the index range to be used according to the range element

6.5.2.4 dummyUsage

6.5.2.4.1 General

Figure 23 shows the structure of the dummyUsage element. The element contains one to many dummy elements.

6.5.2.4.2 dummy

The dummy element has no content. The element is used to enable and disable certain dummy entries for the dummy mapping.

The dummy element contains the attribute given in Table 50.

Table 50 — Attribute of element dummy

Attribute	Data type	Use	Description
entry	xsd:string	required	The string is constructed using the name of the dummy object, followed by the equal sign and then the value of either 0 for disabled mapping or 1 for enabled mapping; the allowed values are as follows: — Dummy0001=0 — Dummy0002=0 — Dummy0003=0 — Dummy0004=0 — Dummy0005=0 — Dummy0006=0 — Dummy0007=0 — Dummy0001=1 — Dummy0002=1 — Dummy0003=1 — Dummy0004=1 — Dummy0005=1 — Dummy0006=1 — Dummy0007=1

6.5.2.5 dynamicChannels

6.5.2.5.1 General

Figure 23 shows the structure of the dynamicChannels element. The element contains one to many dynamicChannel elements.

6.5.2.5.2 dynamicChannel

The dynamicChannel element is used to mark available channels that can be used to create a link between the data to be communicated in the CANopen network and the application program running on the device.

The dynamicChannel element contains the attributes given in Table 51.

Table 51 — Attributes of element dynamicChannel

Attribute	Data type	Use	Description
dataType	xsd:hexBinary	required	CANopen data type (two hex digits)
accessType	xsd:NMTOKEN	required	Access type of the object; valid values: — readOnly — writeOnly — readWriteOutput
startIndex	xsd:hexBinary	required	The index of the first object in the object dictionary used by the process image.
endIndex	xsd:hexBinary	required	The index of the last object in the object dictionary used by the process image.
maxNumber	xsd:unsignedInt	required	The maximum number of objects, which can be allocated in this segment.
addressOffset	xsd:hexBinary	required	The content is the offset of the segment inside the process image.

Attribute	Data type	Use	Description
bitAlignment	xsd:unsignedByte	optional	The content defines the bit alignment used. Most often this will be 1 (bit alignment), 8 (byte alignment), 16 (word alignment), or 32 (double word alignment).

6.5.2.6 moduleManagement

6.5.2.6.1 General

The optional moduleManagement element, if present, contains a choice of either an interfaceList element or a single moduleInterface element.

The interfaceList element (see 6.5.2.6.2) shall be present if the device implements the base of a module device.

The single moduleInterface element shall be present if the device implements a single module. The single moduleInterface element shall contain the attributes given in Table 52.

Table 52 — Attributes of element moduleInterface

Attribute	Data type	Use	Description
childIDRef	xsd:NCName	required	Unique ID of the referenced module in the device profile part
addressMin	xsd:positiveInteger	required	Minimum allowed address
addressMax	xsd:positiveInteger	optional	Maximum allowed address
addressing	xsd:NMTOKEN	optional	Addressing scheme of the module; valid values: — inherit (default) – the addressing scheme from the interface definition of the base of the modular device is inherited — auto – auto addressing by the modular device — manual – manual addressing by the user

6.5.2.6.2 interfaceList / interface

6.5.2.6.2.1 General

The optional interfaceList element, if present, contains a sequence of one to many interface elements.

The mandatory interface element is used to indicate the properties of one interface provided by a modulare device. These properties include the range of the CANopen indicies to be used. The interface element contains the attributes defined in Table 53.

Table 53 — Attributes of element interface

Attribute	Data type	Use	Description
uniqueIDRef	xsd:IDREF	required	Reference to the unique ID of the same interface definition as part of the device profile part.
addressing	xsd:NMTOKEN	required	Addressing scheme used for the CANopen indicies; valid values: — auto – address is assigned continuously by the modulare device — manual – address is assigned by the user

6.5.2.6.2.2 rangeList / range

The mandatory rangeList element contains a sequence of one to many range elements.

The mandatory range element is used to define a range of CANopen indicies to be used when creating new objects based on the connected modules to a modulare device. The range element contains the attributes defined in Table 54.

Table 54 — Attributes of element range

Attribute	Data type	Use	Description
name	xsd:string	required	Name of the range
moduleType	xsd:ID	required	Type of the module
baseIndex	xsd:hexBinary	required	First index to be used within the range
maxIndex	xsd:hexBinary	required	Last index to be used within the range
maxSubIndex	xsd:hexBinary	optional	Last sub-index to be used within the range, if sortMode is set to subIndex
sortMode	xsd:NMTOKEN	required	Sort mode according to either a new index or a new subindex is created for any new object; valid values: — index — subIndex
sortNumber	xsd:NMTOKEN	required	The rule how a new index or subindex is created; valid values: — continous – a new index or subindex is created following the other with respect of the sortStep — address – a new index or subindex is created depending on the address of the module
sortStep	xsd:hexBinary	optional	Step width for the next index or subindex; if not present it is set to 1

6.5.3 TransportLayers

6.5.3.1 General

Figure 24 shows the structure of the CANopen TransportLayers element. It contains a mandatory sub-element PhysicalLayer, which in turn contains a mandatory sub-element baudRate.

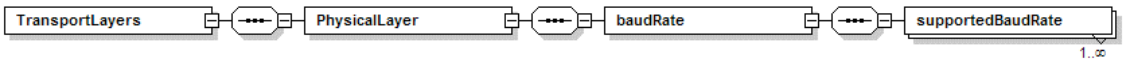

Figure 24 — CANopen TransportLayers element

The baudRate element in Figure 24 contains a sequence of supportedBaudRate sub-elements, which enumerate the bit rates supported by the device type.

A default value is specified for the single attribute of the baudRate element, as given in Table 55.

Table 55 — Attribute of element baudRate

Attribute	Data type	Use	Description
defaultValue	xsd:string	default	Default value for the bit rate of the device type - this value is preset to 250 Kbps; other valid values for the attribute defaultValue are: — 10 Kbps — 20 Kbps — 50 Kbps — 100 Kbps (see NOTE) — 125 Kbps — 250 Kbps — 500 Kbps — 800 Kbps — 1000 Kbps — auto-baudRate
NOTE This value is not officially defined by CAN in Automation specifications.			

The supportedBaudRate sub-element has an empty content and the single attribute given in Table 56.

Table 56 — Attribute of element supportedBaudRate

Attribute	Data type	Use	Description
value	xsd:string	required	One out of the set of bit rate values supported by the device type; valid values: — 10 Kbps — 20 Kbps — 50 Kbps — 100 Kbps (see NOTE) — 125 Kbps — 250 Kbps — 500 Kbps — 800 Kbps — 1000 Kbps — auto-baudRate
NOTE This value is not officially defined by CAN in Automation specifications.			

6.5.4 NetworkManagement

6.5.4.1 General

Figure 25 shows the structure of the CANopen NetworkManagement element. Further details are provided in the following subclauses.

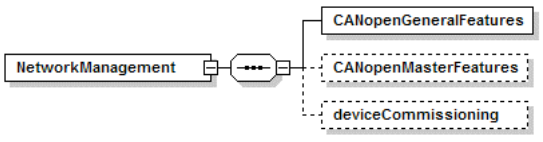

Figure 25 — CANopen NetworkManagement element

6.5.4.2 CANopenGeneralFeatures

The mandatory CANopenGeneralFeatures element in Figure 25 provides that information of a device type, which is relevant for slave functionality as well as for master functionality.

The CANopenGeneralFeatures element has an empty content; data is stored in the attributes of the element given in Table 57.

Table 57 — Attributes of element CANopenGeneralFeatures

Attribute	Data type	Use	Description
groupMessaging	xsd:boolean	default	States the support of multiplexed PDOs
dynamicChannels	xsd:unsignedByte	default	States the support of dynamic network variable generation; a value of 0 means that dynamic channels are not supported; any value different from 0 means that the corresponding number of channels is supported
selfStartingDevice	xsd:boolean	default	States the support of self-starting device functionality
SDORequestingDevice	xsd:boolean	default	States the support of requesting SDOs
granularity	xsd:unsignedByte	required	States the granularity of the PDO mapping supported by the CANopen device; a value different from 0 defines the bit size of the smallest unit of data that is supported to be mapped by the CANopen device; a value of 0 defines that the mapping is not modifiable
nrOfRxPDO	xsd:unsignedShort	default	Number of supported Receive PDOs
nrOfTxPDO	xsd:unsignedShort	default	Number of supported Transmit PDOs
bootUpSlave	xsd:boolean	default	States the support of boot-up slave functionality
layerSettingServiceSlave	xsd:boolean	default	States the support of layer setting services as a slave
NOTE All attributes of use default and type boolean in the table have the default value false; all attributes of use default and type unsignedByte and unsignedShort in the table have the default value 0.			

6.5.4.3 CANopenMasterFeatures

If present, the optional CANopenMasterFeatures element in Figure 25 provides that information of a device type, which is only relevant for master functionality.

The CANopenMasterFeatures element has an empty content; data is stored in the attributes of the element, which are given in Table 58.

Table 58 — Attributes of element CANopenMasterFeatures

Attribute	Data type	Use	Description
bootUpMaster	xsd:boolean	default	States the support of boot-up master functionality
flyingMaster	xsd:boolean	default	States the support of flying master functionality
SDOManager	xsd:boolean	default	Device type is able to act as an SDO manager
configurationManager	xsd:boolean	default	Device type is able to act as a configuration manager
layerSettingServiceMaster	xsd:boolean	default	States the support of layer setting services as a master
NOTE All attributes are of type boolean and have the default value false.			

6.5.4.4 deviceCommissioning

The deviceCommissioning element has an empty content.

The deviceCommissioning element contains the attributes given in Table 59.

Table 59 — Attributes of element deviceCommissioning

Attribute	Data type	Use	Description
nodeID	xsd:unsignedByte	required	The unique ID of the device.
nodeName	xsd:string	required	The name of the device.
actualBaudRate	xsd:string	required	Value of the actual baud rate
networkNumber	xsd:unsignedLong	required	The unique number of the network segment to which the device is connected.
networkName	xsd:string	required	The name of the network segment to which the device is connected.
CANopenManager	xsd:boolean	required	Device is able to act as a CANopen manager

Annex A — Normative

A.1 CANopen device profile template schemas

A.1.1 XML schema: ISO15745ProfileContainer.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.canopen.org/xml/1.1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.canopen.org/xml/1.1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0">
 <xsd:element name="ISO15745ProfileContainer">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element ref="ISO15745Profile" maxOccurs="unbounded"/>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="ISO15745Profile">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="ProfileHeader" type="ProfileHeader_DataType"/>
    <xsd:element name="ProfileBody" type="ProfileBody_DataType"/>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:annotation>
  <xsd:documentation>* HEADER SECTION *</xsd:documentation>
 </xsd:annotation>
 <xsd:complexType name="ProfileHeader_DataType">
  <xsd:sequence>
   <xsd:element name="ProfileIdentification" type="xsd:string">
    <xsd:annotation>
     <xsd:documentation>The ProfileIdentification element identifies the current
profile.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
   <xsd:element name="ProfileRevision" type="xsd:string">
    <xsd:annotation>
     <xsd:documentation>The ProfileRevision element identifies the current profile
revision.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
   <xsd:element name="ProfileName" type="xsd:string">
    <xsd:annotation>
     <xsd:documentation>The ProfileName element contains a descriptive English name
of the current profile. In case that more than one ProfileBody element are present
within a device profile, it is suggested that the value of the ProfileName element
should be the concatenation of the values of the productName elements inside the
respective DeviceIdentity elements.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
   <xsd:element name="ProfileSource" type="xsd:string">
    <xsd:annotation>
     <xsd:documentation>The ProfileSource element identifies the validator of the
current profile.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
   <xsd:element name="ProfileClassID" type="ProfileClassID_DataType">
    <xsd:annotation>
     <xsd:documentation>The ProfileClassID element identifies the class of the
current profile according to ISO 15745-1.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
   <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:annotation>
     <xsd:documentation>The ISO15745Reference element states the ISO 15745 part,
edition and technology, to which the description conforms.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
   <xsd:element name="IASInterfaceType" type="IASInterface_DataType" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
```



```
  <xsd:annotation>
    <xsd:documentation>* BODY SECTION *</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType name="ProfileBody_DataType" abstract="true"/>
  <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:annotation>
      <xsd:documentation>The ProfileIdentification, ProfileRevision, and ProfileLocation
attributes of the ProfileHandle_DataType refer to the external non-XML data file. This
can be used by legacy systems that are in the process of migrating to
XML.</xsd:documentation>
    </xsd:annotation>
    <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>
```

A.1.2 XML schema: CommonElements.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.canopen.org/xml/1.1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.canopen.org/xml/1.1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0">
  <!--################ common attribute group-->
  <xsd:attributeGroup name="ag_formatAndFile">
    <xsd:attribute name="formatName" type="xsd:string" fixed="CANopen">
      <xsd:annotation>
```



```
      <xsd:documentation>Format identifier that should be set to
"CANopen".</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="formatVersion" type="xsd:string" fixed="1.0">
     <xsd:annotation>
      <xsd:documentation>Format version identifier that should be set to
1.0.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileName" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>Name of the file with extension without
path.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileCreator" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>Person creating the file.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileCreationDate" type="xsd:date" use="required">
     <xsd:annotation>
      <xsd:documentation>Date of file creation.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileCreationTime" type="xsd:time" use="optional">
     <xsd:annotation>
      <xsd:documentation>Time of file creation.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileModifiedBy" type="xsd:string" use="optional">
     <xsd:annotation>
      <xsd:documentation>Person modifying the file.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileModificationDate" type="xsd:date" use="optional">
     <xsd:annotation>
      <xsd:documentation>Date of laste file change.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileModificationTime" type="xsd:time" use="optional">
     <xsd:annotation>
      <xsd:documentation>Time of last file change.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="fileVersion" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>Vendor specific version of the file.</xsd:documentation>
     </xsd:annotation>
   </xsd:attribute>
  </xsd:attributeGroup>
  <!--################ common groups-->
  <xsd:group name="g_labels">
    <xsd:annotation>
     <xsd:documentation>The group g_labels supports the introduction of a label (name)
and a description in the context of the parent element.
Every element, which needs a name or a description, shall select one or more suitable
of the four elements to perform this task: the label, the description, the labelRef and
the descriptionRef element.</xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
     <xsd:choice maxOccurs="unbounded">
      <xsd:element name="label">
       <xsd:annotation>
        <xsd:documentation>This element allows storage of the identifying name inside
the XML file itself.</xsd:documentation>
       </xsd:annotation>
       <xsd:complexType>
        <xsd:simpleContent>
         <xsd:extension base="xsd:string">
          <xsd:attribute name="lang" type="xsd:language" use="required">
           <xsd:annotation>
            <xsd:documentation>This attribute references the language used for the
identifying text. This attribute consists of a combination of a language code (as
defined in ISO 639-2) plus an optional dash character plus an optional country code (as
defined in ISO 3166-1), ex: en-us, de, fr.</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
```



```
          </xsd:extension>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="description">
      <xsd:annotation>
       <xsd:documentation>This element allows storage of descriptive information
inside the XML file itself.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
       <xsd:simpleContent>
        <xsd:extension base="xsd:string">
         <xsd:attribute name="lang" type="xsd:language" use="required">
          <xsd:annotation>
           <xsd:documentation>This attribute references the language used for the
descriptive information. This attribute consists of a combination of a language code
(as defined in ISO 639-2) plus an optional dash character plus an optional country code
(as defined in ISO 3166-1), ex: en-us, de, fr.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
         <xsd:attribute name="URI" type="xsd:anyURI" use="optional">
          <xsd:annotation>
           <xsd:documentation>Optional link to further descriptive
information.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
        </xsd:extension>
       </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="labelRef">
      <xsd:annotation>
       <xsd:documentation>This element allows storage of identifying names inside an
external text resource file.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
       <xsd:simpleContent>
        <xsd:extension base="xsd:anyURI">
         <xsd:attribute name="dictID" type="xsd:string" use="required">
          <xsd:annotation>
           <xsd:documentation>This attribute references a single element insie the
dictionaryList element. The dictionary element contains a link to the external text
resource file.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
         <xsd:attribute name="textID" type="xsd:string" use="required">
          <xsd:annotation>
           <xsd:documentation>This attribute references a chracter sequence inside
the external text resource file by pattern matching.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
        </xsd:extension>
       </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="descriptionRef">
      <xsd:annotation>
       <xsd:documentation>This element allows storage of reference descriptive texts
inside an external text resource file.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
       <xsd:simpleContent>
        <xsd:extension base="xsd:anyURI">
         <xsd:attribute name="dictID" type="xsd:string" use="required">
          <xsd:annotation>
           <xsd:documentation>This attribute references a single element insie the
dictionaryList element. The dictionary element contains a link to the external text
resource file.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
         <xsd:attribute name="textID" type="xsd:string" use="required">
          <xsd:annotation>
           <xsd:documentation>This attribute references a chracter sequence inside
the external text resource file by pattern matching.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
        </xsd:extension>
       </xsd:simpleContent>
      </xsd:complexType>
```



```
      </xsd:element>
    </xsd:choice>
  </xsd:sequence>
 </xsd:group>
 <xsd:group name="g_simple">
  <xsd:annotation>
   <xsd:documentation>This group contains a choice of elements, whose names represents
the names of all simple data types allowed in the definition of variables or parameters
inside a device profile. The simple data types conform to the elementary data types
defined in IEC 61131-3; the data types BITSTRING and CHAR (=STRING[1]) are
added.</xsd:documentation>
  </xsd:annotation>
  <xsd:choice>
   <xsd:element name="BOOL"/>
   <xsd:element name="BITSTRING"/>
   <xsd:element name="BYTE"/>
   <xsd:element name="CHAR"/>
   <xsd:element name="WORD"/>
   <xsd:element name="DWORD"/>
   <xsd:element name="LWORD"/>
   <xsd:element name="SINT"/>
   <xsd:element name="INT"/>
   <xsd:element name="DINT"/>
   <xsd:element name="LINT"/>
   <xsd:element name="USINT"/>
   <xsd:element name="UINT"/>
   <xsd:element name="UDINT"/>
   <xsd:element name="ULINT"/>
   <xsd:element name="REAL"/>
   <xsd:element name="LREAL"/>
   <xsd:element name="STRING"/>
   <xsd:element name="WSTRING"/>
  </xsd:choice>
 </xsd:group>
 <!--################ common elements-->
 <xsd:element name="vendorID">
  <xsd:annotation>
   <xsd:documentation>This element shall identify the vendor. This information has to
be filled in when the device described is recognized and validated by a consortium.
NOTE: Consortia specific device families and vendor identifiers are
linked.</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
   <xsd:simpleContent>
    <xsd:extension base="xsd:string">
     <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
    </xsd:extension>
   </xsd:simpleContent>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="deviceFamily">
  <xsd:annotation>
   <xsd:documentation>This element shall state the family of the
device.</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
   <xsd:group ref="g_labels"/>
   <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
    <xsd:annotation>
     <xsd:documentation>This attribute shall indicate whether the value is read-only
for a user: false, true (default).</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="productID">
  <xsd:annotation>
   <xsd:documentation>This element states a vendor specific unique identification for
the device type described.</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
   <xsd:simpleContent>
    <xsd:extension base="xsd:string">
     <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
```



```
        <xsd:annotation>
          <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:extension>
  </xsd:simpleContent>
 </xsd:complexType>
</xsd:element>
<xsd:element name="version">
 <xsd:annotation>
  <xsd:documentation>This element is used to store different types of version
information. Multiple version elements are possible.</xsd:documentation>
 </xsd:annotation>
 <xsd:complexType>
  <xsd:simpleContent>
   <xsd:extension base="xsd:string">
    <xsd:attribute name="versionType" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indiciate the type of the version;
valid values:
- SW - software
- FW - firmware
- HW - hardware</xsd:documentation>
     </xsd:annotation>
     <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
       <xsd:enumeration value="SW"/>
       <xsd:enumeration value="FW"/>
       <xsd:enumeration value="HW"/>
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:extension>
  </xsd:simpleContent>
 </xsd:complexType>
</xsd:element>
<xsd:element name="buildDate" type="xsd:date">
 <xsd:annotation>
  <xsd:documentation>This element specifies the build date of the software
unit.</xsd:documentation>
 </xsd:annotation>
</xsd:element>
<xsd:element name="specificationRevision">
 <xsd:annotation>
  <xsd:documentation>This element contains the revision of the specification, to
which this device conforms.</xsd:documentation>
 </xsd:annotation>
 <xsd:complexType>
  <xsd:simpleContent>
   <xsd:extension base="xsd:string">
    <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:extension>
  </xsd:simpleContent>
 </xsd:complexType>
</xsd:element>
</xsd:schema>
```

A.1.3 XML schema: ProfileBody_Device_CANopen.xsd

This schema includes the schema “ISO15745ProfileContainer.xsd” in A.1.1 and the schema “CommonElements.xsd” in 0.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.canopen.org/xml/1.1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```



```
targetNamespace="http://www.canopen.org/xml/1.1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0">
  <xsd:include schemaLocation="ISO15745ProfileContainer.xsd"/>
  <xsd:include schemaLocation="CommonElements.xsd"/>
  <!--################ CANopen profile body device -->
  <xsd:complexType name="ProfileBody_Device_CANopen">
    <xsd:complexContent>
      <xsd:extension base="ProfileBody_DataType">
        <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:annotation>
              <xsd:documentation>The additional ExternalProfileHandle element enables
external non-XML data to be referenced.</xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        </xsd:sequence>
        <xsd:attributeGroup ref="ag_formatAndFile"/>
        <xsd:attribute name="supportedLanguages" type="xsd:NMTOKENS" use="optional">
          <xsd:annotation>
            <xsd:documentation>This attribute identifies supported languages and consists
of a list of language codes plus optional country codes. ex: supportedLanguages="en-us
de fr es"</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="deviceClass" use="optional">
          <xsd:annotation>
            <xsd:documentation>This attribute provides a classification of the device
profile; valid values:
- compact - the profile is complete and unique for a device or device family
- modular - a bounded set of profiles exist according to the modular functionalities
combined in a specific device occurrence
- configurable - open configurable device that needs an external configurator to create
the profile of one instance</xsd:documentation>
          </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
              <xsd:enumeration value="compact"/>
              <xsd:enumeration value="modular"/>
              <xsd:enumeration value="configurable"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!--################ device identity elements -->
  <xsd:element name="DeviceIdentity">
    <xsd:annotation>
      <xsd:documentation>The DeviceIdentity element contains attributes that are
independent of the network and of the process, and which uniquely identify the
device.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="vendorName"/>
        <xsd:element ref="vendorID" minOccurs="0"/>
        <xsd:element ref="vendorText" minOccurs="0"/>
        <xsd:element ref="deviceFamily" minOccurs="0"/>
        <xsd:element ref="productFamily" minOccurs="0"/>
        <xsd:element ref="productName"/>
        <xsd:element ref="productID" minOccurs="0"/>
        <xsd:element ref="productText" minOccurs="0"/>
        <xsd:element ref="orderNumber" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="version" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="buildDate" minOccurs="0"/>
        <xsd:element ref="specificationRevision" minOccurs="0"/>
        <xsd:element ref="instanceName" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="productFamily">
    <xsd:annotation>
```



```
    <xsd:documentation>This element states a vendor specific affiliation of the device
type to a certain set of devices inside a family. The list of valid productFamily
values is system, tool or consortia specific.
NOTE: Consortia specific device families and vendor identifiers are
linked.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:simpleContent>
     <xsd:extension base="xsd:string">
      <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
       <xsd:annotation>
        <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
     </xsd:extension>
    </xsd:simpleContent>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="instanceName">
   <xsd:annotation>
    <xsd:documentation>This element contains the instance name of the
device.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:simpleContent>
     <xsd:extension base="xsd:string">
      <xsd:attribute name="readOnly" type="xsd:boolean" default="false">
       <xsd:annotation>
        <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
     </xsd:extension>
    </xsd:simpleContent>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="orderNumber">
   <xsd:annotation>
    <xsd:documentation>This element is used to store the single order number of a given
device or the set of different order numbers of the products of a device family,
depending upon whether the device profile describes a device or a device
family.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:simpleContent>
     <xsd:extension base="xsd:string">
      <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
       <xsd:annotation>
        <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
     </xsd:extension>
    </xsd:simpleContent>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="productName">
   <xsd:annotation>
    <xsd:documentation>This element states a vendor specific designation or name of the
device type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:simpleContent>
     <xsd:extension base="xsd:string">
      <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
       <xsd:annotation>
        <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
     </xsd:extension>
    </xsd:simpleContent>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="productText">
   <xsd:annotation>
    <xsd:documentation>This element allows the vendor to provide a short textual
description of the device.</xsd:documentation>
```



```
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
    <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate whether the value is read-only
for a user: false, true (default).</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="vendorName">
   <xsd:annotation>
    <xsd:documentation>This element shall identify the name or the brand name of the
vendor of the device.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:simpleContent>
     <xsd:extension base="xsd:string">
      <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
       <xsd:annotation>
        <xsd:documentation>This attribute shall indicate whether the value is read-
only for a user: false, true (default).</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
     </xsd:extension>
    </xsd:simpleContent>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="vendorText">
   <xsd:annotation>
    <xsd:documentation>This element allows the vendor to provide additional company
information, like address or hotline number. The g_labels group offers the possibility
to include a vendor URI inside this element.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
    <xsd:attribute name="readOnly" type="xsd:boolean" default="true">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate whether the value is read-only
for a user: false, true (default).</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <!--################ device manager elements -->
  <xsd:element name="DeviceManager">
   <xsd:annotation>
    <xsd:documentation>The DeviceManager element contains attributes and supports
services that enable the monitoring the device.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="indicatorList" minOccurs="0"/>
     <xsd:element ref="moduleManagement" minOccurs="0"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <!--################ indicator list elements -->
  <xsd:element name="indicatorList">
   <xsd:annotation>
    <xsd:documentation>This element provides a list of indicators provided by the
device type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="LEDList" minOccurs="0"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="LEDList">
   <xsd:annotation>
    <xsd:documentation>This element  provides a list of LED indicators provided by the
device type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="LED" maxOccurs="unbounded"/>
     <xsd:element ref="combinedState" minOccurs="0" maxOccurs="unbounded"/>
```



```
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="LED">
   <xsd:annotation>
    <xsd:documentation>This element describes the features of a single LEDS of the
device type. A detailed feature description may be provided through the g_labels
group.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels"/>
     <xsd:element ref="LEDstate" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="LEDcolors" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the colors of the LED; valid
values:
- monocolor
- bicolor</xsd:documentation>
     </xsd:annotation>
     <xsd:simpleType>
      <xsd:restriction base="xsd:string">
       <xsd:enumeration value="monocolor"/>
       <xsd:enumeration value="bicolor"/>
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="LEDtype" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate a rough classification of the
supervised item of unctionality; valid values:
- IO
- device
- communication</xsd:documentation>
     </xsd:annotation>
     <xsd:simpleType>
      <xsd:restriction base="xsd:string">
       <xsd:enumeration value="IO"/>
       <xsd:enumeration value="device"/>
       <xsd:enumeration value="communication"/>
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="LEDstate">
   <xsd:annotation>
    <xsd:documentation>This element shall indicate the device states signalled by the
LED and the visual behavior used for signalling the states.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID for a LED state
and may be referenced from a LEDstateRef element.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="state" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the state of the LED; valid
values:
- on
- off
- flashing</xsd:documentation>
     </xsd:annotation>
     <xsd:simpleType>
      <xsd:restriction base="xsd:string">
       <xsd:enumeration value="on"/>
       <xsd:enumeration value="off"/>
       <xsd:enumeration value="flashing"/>
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="LEDcolor" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the color of the LED; valid
values:
```



```
- green
- red</xsd:documentation>
      </xsd:annotation>
      <xsd:simpleType>
       <xsd:restriction base="xsd:string">
        <xsd:enumeration value="green"/>
        <xsd:enumeration value="amber"/>
        <xsd:enumeration value="red"/>
       </xsd:restriction>
      </xsd:simpleType>
     </xsd:attribute>
     <xsd:attribute name="flashingPeriod" type="xsd:unsignedInt" use="optional">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the flashing period of the LED
in milliseconds, if the state is set to flashing.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
     <xsd:attribute name="impulsWidth" type="xsd:unsignedByte" default="50">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the width of the flashing
impulse given in percent (%) of the flashing period, if the state is set to flashing.
If this attribute is missing, the default value is set to 50.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
     <xsd:attribute name="numberOfImpulses" type="xsd:unsignedByte" default="1">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the number of impulses in case
that more than one flashing impulse is inside one flashing period, if the state is set
to flashing. If this attribute is present, the attribute impulsWidth shall be present
too. If this attribute is missing, then the default value is set to
1.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="combinedState">
    <xsd:annotation>
     <xsd:documentation>This element allows the indication of device states which are
signaled by more than one LED. The description of the combioned state is provided
through the g_labels group.
The LED states participating in the signalling of the combined state are referenced by
means of at least two LEDstateRef sub-elements of this element.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:sequence>
      <xsd:group ref="g_labels"/>
      <xsd:element name="LEDstateRef" minOccurs="2" maxOccurs="unbounded">
       <xsd:complexType>
        <xsd:attribute name="stateIDRef" type="xsd:IDREF" use="required">
         <xsd:annotation>
          <xsd:documentation>This attribute shall reference the unique ID of the
referenced LED state.</xsd:documentation>
         </xsd:annotation>
        </xsd:attribute>
       </xsd:complexType>
      </xsd:element>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <!--################ device function elements -->
   <xsd:element name="DeviceFunction">
    <xsd:annotation>
     <xsd:documentation>The DeviceFunction element describes the intrinsic function of a
device in terms of its technology. It contains network independent
descriptions/definitions of the technological device functionality.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element ref="capabilities"/>
      <xsd:element ref="picturesList" minOccurs="0"/>
      <xsd:element ref="dictionaryList" minOccurs="0"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="capabilities">
    <xsd:annotation>
     <xsd:documentation>This element describes all functionalities, their
characteristics, and the important parameters of the device, that need to be known by
```



```
tools which use the device profile to select products with the same or similar
properties.
This element describes device features in a purely textual form.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="characteristicsList" maxOccurs="unbounded"/>
     <xsd:element ref="standardComplianceList" minOccurs="0"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="characteristicsList">
   <xsd:annotation>
    <xsd:documentation>This element is a collection of
characteristics.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="category" minOccurs="0">
      <xsd:annotation>
       <xsd:documentation>This element may indicate a category associated with the
list of characterisitcs.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
       <xsd:group ref="g_labels"/>
      </xsd:complexType>
     </xsd:element>
     <xsd:element ref="characteristic" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="characteristic">
   <xsd:annotation>
    <xsd:documentation>This element describes a single characterisitc of a
device.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="characteristicName"/>
     <xsd:element ref="characteristicContent" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="characteristicContent">
   <xsd:annotation>
    <xsd:documentation>This element contain a value for the characteristic. Multiple
values may be expressed by using multiple characteristicContent elements.
EXAMPLE: An example of a single value for "Maximum operational voltage" is
680V.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="characteristicName">
   <xsd:annotation>
    <xsd:documentation>This element denotes a major technical characteristic of the
device. The vocabulary used in the device data sheet is recommended for the names of
characteristics.
EXAMPLE: "Maximum operational voltage", "Overload protection", "Electrical
durability"</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="standardComplianceList">
   <xsd:annotation>
    <xsd:documentation>This element is a collection of compliantWith sub-
elements.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="compliantWith" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="compliantWith">
   <xsd:annotation>
```



```
    <xsd:documentation>This element shall state the compliance of the device with an
international or company internal standard.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name or number of the
standard.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="range" default="international">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate if the type of the standard;
valied values:
- international
- internal</xsd:documentation>
     </xsd:annotation>
     <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
       <xsd:enumeration value="international"/>
       <xsd:enumeration value="internal"/>
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="picturesList">
   <xsd:annotation>
    <xsd:documentation>This element offers the possibility to link picturees to the
device profile.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="picture" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="picture">
   <xsd:annotation>
    <xsd:documentation>This element shall indicate a single picture linked with the
device profile.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
    <xsd:attribute name="URI" type="xsd:anyURI" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall  provide the link to the external
resource.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="number" type="xsd:unsignedInt" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute may indicate the number of the
picture.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="dictionaryList">
   <xsd:annotation>
    <xsd:documentation>This element offers the possibility to include links to external
text resource files to the device profile.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="dictionary" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="dictionary">
   <xsd:annotation>
    <xsd:documentation>This element shall indicate one link to an external text
resource file.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="file"/>
    </xsd:sequence>
```



```
    <xsd:attribute name="lang" type="xsd:language" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the language used for the file
belonging to a dictionary.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="dictID" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate a unique ID of the
dictionary.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="file">
   <xsd:annotation>
    <xsd:documentation>This element shall indicate a search path for the XDD-files that
may contain information on the connected modules.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="URI" type="xsd:anyURI" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate a relative or absolute file
path that may contain the wildcard characters "?" and "*".</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <!--################ application process elements -->
  <xsd:element name="ApplicationProcess">
   <xsd:annotation>
    <xsd:documentation>The ApplicationProcess element represents the set of services
and aprameters, which constitute the behavior and the interfaces of the device in terms
of the application, independent of the device technology and the underlying
communication networks and communication protocols.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="dataTypeList" minOccurs="0"/>
     <xsd:element ref="functionTypeList" minOccurs="0"/>
     <xsd:element ref="functionInstanceList" minOccurs="0"/>
     <xsd:element ref="templateList" minOccurs="0"/>
     <xsd:element ref="parameterList"/>
     <xsd:element ref="parameterGroupList" minOccurs="0"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="dataTypeList">
   <xsd:annotation>
    <xsd:documentation>This element is present if complex data types like arrays or
data structures are needed inside variable declarations or parameter specifications of
the device profile. It is also used to sub-structure data
components.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_complex" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="functionTypeList">
   <xsd:annotation>
    <xsd:documentation>This element contains a list of all defined function
types.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="functionType" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="functionType">
   <xsd:annotation>
    <xsd:documentation>This element represents the type description of a device
function, which is referenced from at least one instance of that function type.
References from more than one instance of the same function type are also possible.
The description of a function type contains all those objects and data which are common
for all instances of a given function type.</xsd:documentation>
   </xsd:annotation>
```



```
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:element ref="versionInfo" maxOccurs="unbounded"/>
     <xsd:element ref="interfaceList"/>
     <xsd:element ref="functionInstanceList" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name of the function
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the function
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="package" type="xsd:string" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute may indicate a textual association of the
function type with a "package" or similar classification scheme - the usage of this
attribute is left to the profile validator.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="versionInfo">
   <xsd:annotation>
    <xsd:documentation>This element shall provice information on the versioning history
of a function type (concerning the definition of the interface).
To keep track of the versioning history, this element may be entered multiple times in
sequence with the first elemt representing the most recent version and the last element
the first released version.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="organization" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name of the organization
maintaining the function type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="version" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the version identification in
the versioning history; suggested format: "xx.yy" (xx,yy = 0..255)</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="author" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name of the person
maintaining the function type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="date" type="xsd:date" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the date of this
version.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="interfaceList">
   <xsd:annotation>
    <xsd:documentation>This element provides the definition of the interface of the
function type including the input variables, output variables, and configuration
variables.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="inputVars" minOccurs="0"/>
     <xsd:element ref="outputVars" minOccurs="0"/>
     <xsd:element ref="configVars" minOccurs="0"/>
    </xsd:sequence>
   </xsd:complexType>
```



```
  </xsd:element>
  <xsd:element name="inputVars">
    <xsd:annotation>
      <xsd:documentation>This element defines a input variable that is part of the
interface of a function type.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="varDeclaration" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="outputVars">
    <xsd:annotation>
      <xsd:documentation>This element defines a output variable that is part of the
interface of a function type.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="varDeclaration" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="configVars">
    <xsd:annotation>
      <xsd:documentation>This element defines a configuration variable that is part of
the interface of a function type.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="varDeclaration" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="varDeclaration">
    <xsd:annotation>
      <xsd:documentation>In the context of the definition of a structured data type, this
element describes a single component variable (member) of the structure.
In the context of the definition of the interface of a function, this element describes
a single interface variable of the function type.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:group ref="g_labels" minOccurs="0"/>
        <xsd:choice>
          <xsd:group ref="g_simple"/>
          <xsd:element ref="dataTypeIDRef"/>
        </xsd:choice>
        <xsd:element ref="conditionalSupport" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="defaultValue" minOccurs="0"/>
        <xsd:element ref="allowedValues" minOccurs="0"/>
        <xsd:element ref="unit" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required">
        <xsd:annotation>
          <xsd:documentation>This attribute shall indicate the name of the interface
variable or structured component.</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
        <xsd:annotation>
          <xsd:documentation>This attribute shall indicate the unique ID of the interface
variable or strucure component.
NOTE: When creating the unique ID for a variable, it is essential that the ID is unique
over all created IDs inside the XML source file. To allow equal names for component
variables of different data structures and equal names for interface variables of
function types, the ID of a variable should generally concatenate the type name of the
structured data type or the function type with the variable name, to guarantee
uniqueness.</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="start" type="xsd:string" use="optional" default="0">
        <xsd:annotation>
          <xsd:documentation>This attribute may indicate the first element, if the
interface variable or structure component is of type anonymous ARRAY, BITSTRING, STRING
or WSTRING.
NOTE: Anonymous types define the first reference element of an array, bitstring or
string directly in the variable declaration, and not through the reference of a named
complex data type.</xsd:documentation>
```



```
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="size" type="xsd:string" use="optional" default="1">
      <xsd:annotation>
       <xsd:documentation>This attribute may indicate the number of elements, if the
interface variable or structure component is of type anonymous ARRAY, BITSTRING, STRING
or WSTRING.
NOTE: Anonymous types define the size of an array, bitstring or string directly in the
variable declaration, and not through the reference of a named complex data type. In
the case of an array, the data type of the variable gives the type of a single array
element. In the case of a bitsting, the single array element is a single bit. In the
case of a string, the single array element is a single-byte resp. double-byte
character.</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="signed" type="xsd:boolean" use="optional" default="false">
      <xsd:annotation>
       <xsd:documentation>This attribute may indicate if the component of the complex
data type is interpreted as a signed value or not; valid values:
- false (default)
- true</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="offset" type="xsd:string" use="optional" default="0">
      <xsd:annotation>
       <xsd:documentation>This attribute may indicates the offset which is added to the
value to form a scaled value: EngineeringValue = value + offset) *
multiplier.</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="multiplier" type="xsd:string" use="optional" default="1">
      <xsd:annotation>
       <xsd:documentation>This attribute may indicate the scaling factor by which the
value is multiplied to form a scaled value: EngineeringValue = value + offset) *
multiplier.</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="initialValue" type="xsd:string" use="optional">
      <xsd:annotation>
       <xsd:documentation>This attribute may indicate the initial value of the
interface variable or structure component.
NOTE: If present, this attribute defines the initial (default) value of the interface
variable of the function type. It is overwritten by a given default value of a
parameter associated with the interface variable of the function
instance.</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="dataTypeIDRef">
   <xsd:annotation>
    <xsd:documentation>This element serves to reference a complex data type, either
from an interface variable of a function type, or from an array type definition, or
from a component variable inside the definition of a structured data
type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="required">
      <xsd:annotation>
       <xsd:documentation>This attribute shall inidcate the unique ID of the referenced
data type.</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="functionInstanceList">
   <xsd:annotation>
    <xsd:documentation>This element contains thefrom function types instanciated
functions and the connections of the related variables of the interface to those
instanciated functions.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="functionInstance" maxOccurs="unbounded"/>
     <xsd:element ref="connection" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="functionInstance">
```



```
   <xsd:annotation>
    <xsd:documentation>This element represents an accessible application function of
the device type, independent of the network type or protocol.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name of the function
instance.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the function
instance.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="typeIDRef" type="xsd:IDREF" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the referenced
function type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="connection">
   <xsd:annotation>
    <xsd:documentation>This element represents connections between output and input
variables of the function instances.
This element defines a connection between an output variable of a function instance and
an input variable of another function instance. Inside function types, the connection
may also be drawn between an input variable of the function type and an input variable
of a contained function instance, or between an output variable of a contained function
instance and an output variable of the function type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="source" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate thew starting point of the
connection.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="destination" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the endpoint of the
connection.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="description" type="xsd:string" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute may indicate an additional textual description
of the connection.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="templateList">
   <xsd:annotation>
    <xsd:documentation>This element contains a list of parameter templates and
templates for allowed values.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="parameterTemplate" minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element ref="allowedValuesTemplate" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="parameterTemplate">
   <xsd:annotation>
    <xsd:documentation>This element shall define a template for parameter
definitions.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:choice minOccurs="0">
```



```
       <xsd:group ref="g_simple"/>
       <xsd:element ref="dataTypeIDRef"/>
      </xsd:choice>
      <xsd:element ref="conditionalSupport" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="actualValue" minOccurs="0"/>
      <xsd:element ref="defaultValue" minOccurs="0"/>
      <xsd:element ref="substituteValue" minOccurs="0"/>
      <xsd:element ref="allowedValues" minOccurs="0"/>
      <xsd:element ref="unit" minOccurs="0"/>
      <xsd:element ref="property" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="ag_parameter"/>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="allowedValuesTemplate">
   <xsd:annotation>
    <xsd:documentation>This element shall define a template of allowed values for later
use.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="value" minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element ref="range" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID for a single
template of allowed values.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="parameterList">
   <xsd:annotation>
    <xsd:documentation>This element contains a list of parameters.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="parameter" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="parameter">
   <xsd:annotation>
    <xsd:documentation>This element shall define a parameter.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels"/>
     <xsd:choice>
      <xsd:group ref="g_simple"/>
      <xsd:element ref="dataTypeIDRef"/>
      <xsd:element ref="variableRef" maxOccurs="unbounded"/>
     </xsd:choice>
     <xsd:element ref="conditionalSupport" minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element ref="denotation" minOccurs="0"/>
     <xsd:element ref="actualValue" minOccurs="0"/>
     <xsd:element ref="defaultValue" minOccurs="0"/>
     <xsd:element ref="substituteValue" minOccurs="0"/>
     <xsd:element ref="allowedValues" minOccurs="0"/>
     <xsd:element ref="unit" minOccurs="0"/>
     <xsd:element ref="property" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="ag_parameter"/>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="variableRef">
   <xsd:annotation>
    <xsd:documentation>This element buidls a reference to an interface variable of a
function instance, or, if the variable is an array or a structure, possibly a reference
to a member of the variable (array element or structure component).</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="instanceIDRef" maxOccurs="unbounded"/>
     <xsd:element ref="variableIDRef"/>
     <xsd:element ref="memberRef" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="position" type="xsd:unsignedByte" default="1">
```



```
      <xsd:annotation>
       <xsd:documentation>This attribute defines the sequence of multiple mapped data
items into a single parameter object.
Position=1 means starting the mapping at the lowest bit position. The number of bits is
defined by the data type of the data item. Subsequent data items are packed without
gaps.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="instanceIDRef">
   <xsd:annotation>
    <xsd:documentation>This element serves to reference a function instance., which may
reside on the level of the ApplicationProcess element or on the level of a functionType
element.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the referenced
function instance.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="variableIDRef">
   <xsd:annotation>
    <xsd:documentation>This element serves to reference an interface variable of a
function type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the referenced
interface variable of a function type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="memberRef">
   <xsd:annotation>
    <xsd:documentation>This element either references the respective component of an
interface variable of structured data type (attribute uniqueIDRef is used), or the
respective array element of an interface variable of array data type (attribute index
is used). One of these two attributes shall be present.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the referenced
component of a structured data type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="index" type="xsd:long" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the index of the referenced
array element.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="actualValue">
   <xsd:annotation>
    <xsd:documentation>This element serves to hold the actual value of the
parameter.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels" minOccurs="0"/>
    <xsd:attributeGroup ref="ag_value"/>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="conditionalSupport">
   <xsd:annotation>
    <xsd:documentation>This element refers to a single optional parameter. If at least
one of those optional parameters is implemented, the conditional parameter has also to
be implemented.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="paramIDRef" type="xsd:IDREF" use="required">
```



```
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the unique ID of the referenced
optional parameter.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="denotation">
    <xsd:annotation>
     <xsd:documentation>This element serves to hold application-specific, multilingual
names of the parameter.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:group ref="g_labels"/>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="defaultValue">
    <xsd:annotation>
     <xsd:documentation>This element serves to hold the default value.
The default value of a component of a datatype is overriden by the default value of a
template definition, of a parameter definition and of a interface variable of the
function type associated with the parameter in that order.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:attributeGroup ref="ag_value"/>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="substituteValue">
    <xsd:annotation>
     <xsd:documentation>This element defines a specific value of the parameter that is
provided to the device application in certain device operating states (e.g. on device
fault).</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:attributeGroup ref="ag_value"/>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="allowedValues">
    <xsd:annotation>
     <xsd:documentation>This element defines a list of supported values and/or single
range or several ranges of supported values for the component of a datatype, parameter
template definition, parameter definition, and interfcae variable of the function type
associated with the parameter. The allowed values are re-defined in that
order.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element ref="value" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="range" minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
     <xsd:attribute name="templateIDRef" type="xsd:IDREF" use="optional">
      <xsd:annotation>
       <xsd:documentation>This attribute may indicate the unique ID of a referenced
template of allowed values.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="value">
    <xsd:annotation>
     <xsd:documentation>Thsi element defines a single value within the allowed
values.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:attributeGroup ref="ag_value"/>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="range">
    <xsd:annotation>
     <xsd:documentation>This element defines a range of allowed values characterized by
the minimum and maximum value of the range and may be step width within the
range.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="minValue">
```



```
       <xsd:annotation>
        <xsd:documentation>This element shall indicate the minimum value of the
range.</xsd:documentation>
       </xsd:annotation>
       <xsd:complexType>
        <xsd:group ref="g_labels" minOccurs="0"/>
        <xsd:attributeGroup ref="ag_value"/>
       </xsd:complexType>
      </xsd:element>
      <xsd:element name="maxValue">
       <xsd:annotation>
        <xsd:documentation>This element shall indicate the maximum value of the
range.</xsd:documentation>
       </xsd:annotation>
       <xsd:complexType>
        <xsd:group ref="g_labels" minOccurs="0"/>
        <xsd:attributeGroup ref="ag_value"/>
       </xsd:complexType>
      </xsd:element>
      <xsd:element name="step" minOccurs="0">
       <xsd:annotation>
        <xsd:documentation>This element shall indicate the step width between the
minimum value and the maximum value. If this element is omitted, then the minimum
possible step width is assumed.</xsd:documentation>
       </xsd:annotation>
       <xsd:complexType>
        <xsd:group ref="g_labels" minOccurs="0"/>
        <xsd:attributeGroup ref="ag_value"/>
       </xsd:complexType>
      </xsd:element>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="unit">
    <xsd:annotation>
     <xsd:documentation>This element defines the engineering unit of a component of a
data type, parameter template definition, parameter definition, and interface variable
of the function type associated with the parameter as specified in ISO 1000 (e.g. time,
temperature, pressure, flow, acceleration, current, energy).</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:group ref="g_labels"/>
     <xsd:attribute name="multiplier" type="xsd:string" use="required">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the multiplier for engineering
units of analog parameters,</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
     <xsd:attribute name="unitURI" type="xsd:anyURI" use="optional">
      <xsd:annotation>
       <xsd:documentation>This attribute may link to the respective unit definition in
a file containing all engineering units as standardised in ISO 1000 (e.g. time,
temperature, pressure, flow, acceleration, current, energy, ...)</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="property">
    <xsd:annotation>
     <xsd:documentation>This element is introduced as a generic element to allow the
inclusion of values for additional specific properties into the description of a
parameter.</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
     <xsd:attribute name="name" type="xsd:string" use="required">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the name of the
property.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
     <xsd:attribute name="value" type="xsd:string" use="required">
      <xsd:annotation>
       <xsd:documentation>This attribute shall indicate the value of the
property.</xsd:documentation>
      </xsd:annotation>
     </xsd:attribute>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="parameterGroupList">
```



```
   <xsd:annotation>
    <xsd:documentation>This element contains a list of parameter
groups.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="parameterGroup" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="parameterGroup">
   <xsd:annotation>
    <xsd:documentation>This element combines a set of parameters to build a group of
parameters which serve a specific purpose, e.g. to prepare HMI
views.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels"/>
     <xsd:element ref="parameterGroup" minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element ref="parameterRef" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the parameter
group.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="kindOfAccess" type="xsd:string" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute may classify the parameters of the parameter
group.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="parameterRef">
   <xsd:annotation>
    <xsd:documentation>This element serves to reference a parameter
element.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the referenced
parameter.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <!--################ complex types -->
  <xsd:element name="array">
   <xsd:annotation>
    <xsd:documentation>This element serves to describe an array data type, which may be
referenced from an interface variable of a function type, from another array type
definition, from a component variable inside the definition of a structured data type,
or from a parameters specifcition.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:element ref="subrange" maxOccurs="unbounded"/>
     <xsd:choice>
      <xsd:group ref="g_simple"/>
      <xsd:element ref="dataTypeIDRef"/>
     </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the data type name of the array
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the array
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
```



```
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="subrange">
   <xsd:annotation>
    <xsd:documentation>This element defined the lower and upper limit of an array index
for one dimension of the array.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="lowerLimit" type="xsd:long" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the lower limit of the
subrange.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="upperLimit" type="xsd:long" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the upper limit of the
subrange.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="struct">
   <xsd:annotation>
    <xsd:documentation>This element serves to describe a structured data type, which
may be referenced from an interface variable of a function type, from an array type
definition, from a component variable inside the definition of another structured data
type, or from a parameter specification.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:element ref="varDeclaration" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the data type name of the
structured data type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribuet shall indicate the unique ID of the structured
data type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="enum">
   <xsd:annotation>
    <xsd:documentation>This element serves to describe an enumerated data type, which
may be referenced from an interface variable of a function type, from an array type
definition, from a component variable inside the definition of a structured data type,
or from a parameter specification.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:element ref="enumValue" maxOccurs="unbounded"/>
     <xsd:group ref="g_simple" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name of the enumerated data
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the enumerated
data type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="size" type="xsd:string" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute may indicate a  number of enumerated values of
the enumerated data type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
```



```
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="enumValue">
   <xsd:annotation>
    <xsd:documentation>This element defines a single enumeration
constant.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:group ref="g_labels"/>
    <xsd:attribute name="value" type="xsd:string" use="optional">
     <xsd:annotation>
      <xsd:documentation>This attribute may define the fixed numerical value for the
enumeration constant, represented as a string of characters.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="derived">
   <xsd:annotation>
    <xsd:documentation>This element serves to derive a new data type from a given base
data type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:element ref="count" minOccurs="0"/>
     <xsd:choice>
      <xsd:group ref="g_simple"/>
      <xsd:element ref="dataTypeIDRef"/>
     </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the name of the derived data
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the derived
data type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="count">
   <xsd:annotation>
    <xsd:documentation>This element defines the number of used units of the base type
of the derived type.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels" minOccurs="0"/>
     <xsd:element ref="defaultValue"/>
     <xsd:element ref="allowedValues" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the unique ID of the
count.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="access" default="read">
     <xsd:annotation>
      <xsd:documentation>This attribute may define the operations are valid for the
count; valid values:
- const - read access only; value is not changing
- read - read access only (default value)
- write - write access only
- readWrite - both read and write access
- noAccess - access denied</xsd:documentation>
     </xsd:annotation>
     <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
       <xsd:enumeration value="const"/>
       <xsd:enumeration value="read"/>
       <xsd:enumeration value="write"/>
       <xsd:enumeration value="readWrite"/>
       <xsd:enumeration value="noAccess"/>
```



```
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <!--################ groups-->
  <xsd:group name="g_complex">
   <xsd:choice>
    <xsd:element ref="array"/>
    <xsd:element ref="struct"/>
    <xsd:element ref="enum"/>
    <xsd:element ref="derived"/>
   </xsd:choice>
  </xsd:group>
  <!--################ attribute groups-->
  <xsd:attributeGroup name="ag_parameter">
   <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
    <xsd:annotation>
     <xsd:documentation>This attribute shall indicate the unique ID of the
parameter.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="access" default="read">
    <xsd:annotation>
     <xsd:documentation>This attribute shall indicate which operations are valid for
the parameter; valid values:
- const - read access only; the value is not changing
- read - read access only (default value)
- write - write access only
- readWrite - both read and write access
- readWriteInput - both read and write access, but represents process input data
- readWriteOutput - both read and write access, but represents process output data
- noAccess - access denied</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
     <xsd:restriction base="xsd:NMTOKEN">
      <xsd:enumeration value="const"/>
      <xsd:enumeration value="read"/>
      <xsd:enumeration value="write"/>
      <xsd:enumeration value="readWrite"/>
      <xsd:enumeration value="readWriteInput"/>
      <xsd:enumeration value="readWriteOutput"/>
      <xsd:enumeration value="noAccess"/>
     </xsd:restriction>
    </xsd:simpleType>
   </xsd:attribute>
   <xsd:attribute name="accessList" use="optional">
    <xsd:annotation>
     <xsd:documentation>This attribute may defines a list of additionally allowed
operations on the parameter. The valie values are the same as for the access attribute,
but the parser will not check the usage of the correct values.</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
     <xsd:restriction base="xsd:NMTOKENS"/>
    </xsd:simpleType>
   </xsd:attribute>
   <xsd:attribute name="support" use="optional">
    <xsd:annotation>
     <xsd:documentation>This attribute may defines whether or not the parameter has to
be implemented in the device; valid values:
- mandatory - parameter implementation required
- optional - parameter implementation is possible but not required
- conditional - parameter implementation is required if one or more optional
parameter(s) is (are) implemented. These parameters are specified using the element
conditionalSupport.</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
     <xsd:restriction base="xsd:NMTOKEN">
      <xsd:enumeration value="mandatory"/>
      <xsd:enumeration value="optional"/>
      <xsd:enumeration value="conditional"/>
     </xsd:restriction>
    </xsd:simpleType>
   </xsd:attribute>
   <xsd:attribute name="persistent" type="xsd:boolean" default="false">
    <xsd:annotation>
     <xsd:documentation>This attribute defines the behavior after power failure; valid
values:
- false (default)
```



```
- true</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="offset" type="xsd:string" use="optional" default="0">
    <xsd:annotation>
     <xsd:documentation>This attribute may indicates the offset which is added to an
actual value to form a scaled value: EngineeringValue = (ParameterValue + offset) *
multiplier.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="multiplier" type="xsd:string" use="optional" default="1">
    <xsd:annotation>
     <xsd:documentation>This attribute may indicate the scaling factor by which an
actual value is multiplied to form a scaled value: EngineeringValue = (ParameterValue +
offset) * multiplier.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="templateIDRef" type="xsd:IDREF" use="optional">
    <xsd:annotation>
     <xsd:documentation>This attribute may indicate the unique ID of a referenced
parameter template.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
  </xsd:attributeGroup>
  <xsd:attributeGroup name="ag_value">
   <xsd:attribute name="value" type="xsd:string" use="required">
    <xsd:annotation>
     <xsd:documentation>This attribute shall indicate the value.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="offset" type="xsd:string" use="optional">
    <xsd:annotation>
     <xsd:documentation>This attribute may indicate the offset, which is added to the
value for form a scaled value: EngineeringValue = (value + offset) * multiplier.
If not present, the respective value shall be used.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
   <xsd:attribute name="multiplier" type="xsd:string" use="optional">
    <xsd:annotation>
     <xsd:documentation>This attribute may indicate the scaling factor by which the
value is multiplied to form a scaled value: EngineeringValue = (value + offset) *
multiplier.
If not present, the respective value shall be used.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
  </xsd:attributeGroup>
  <xsd:element name="moduleManagement">
   <xsd:complexType>
    <xsd:choice>
     <xsd:element ref="moduleInterfaceList"/>
     <xsd:element name="moduleInterface">
      <xsd:annotation>
       <xsd:documentation>This element shall be by a single module to identify the
module and to describe its properties. </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
       <xsd:attribute name="childID" type="xsd:NCName" use="required">
        <xsd:annotation>
         <xsd:documentation>This attribute shall indicate the unique ID of a module.
To guarantee uniqueness this ID shall consists of the 8 hexadecimal characters of the
vendor-ID and a manufacturer specific part.</xsd:documentation>
        </xsd:annotation>
       </xsd:attribute>
       <xsd:attribute name="type" type="xsd:NCName" use="required">
        <xsd:annotation>
         <xsd:documentation>This attribute shall indicate the type of the
module.</xsd:documentation>
        </xsd:annotation>
       </xsd:attribute>
       <xsd:attribute name="maxCount" type="xsd:nonNegativeInteger" default="0">
        <xsd:annotation>
         <xsd:documentation>This attribute shall indicate the maximum number of
multiples of this module allowed to be connected to a modular device. The value of 0
shall indiacte no limitation.</xsd:documentation>
        </xsd:annotation>
       </xsd:attribute>
      </xsd:complexType>
     </xsd:element>
    </xsd:choice>
```



```
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="moduleInterfaceList">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="interface" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="interface">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:group ref="g_labels"/>
     <xsd:element ref="fileList"/>
     <xsd:element ref="moduleTypeList"/>
     <xsd:element name="connectedModuleList" minOccurs="0">
      <xsd:annotation>
       <xsd:documentation>This element shall contain a list of all connected
modules.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
       <xsd:sequence>
        <xsd:element name="connectedModule" maxOccurs="unbounded">
         <xsd:annotation>
          <xsd:documentation>Each connected module shal be indicated with this
element.</xsd:documentation>
         </xsd:annotation>
         <xsd:complexType>
          <xsd:attribute name="childIDRef" type="xsd:NCName" use="required">
           <xsd:annotation>
            <xsd:documentation>This attribute shall reference the connected
module.</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
         </xsd:complexType>
        </xsd:element>
       </xsd:sequence>
      </xsd:complexType>
     </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="uniqueID" type="xsd:ID" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall be the unique ID of the module
interface.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="maxChilds" type="xsd:positiveInteger" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate the maximum number of modules
to be conntected to that interface.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="unusedSlots" type="xsd:boolean" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate if empty slots are allowed when
modules are connected.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="multipleChilds" type="xsd:boolean" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall indicate if multiple modules of the same
type are allowed to be connected.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="moduleType">
   <xsd:annotation>
    <xsd:documentation>This element shall indicate a module type that can be connected
to this interface.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:attribute name="type" type="xsd:NCName" use="required">
     <xsd:annotation>
      <xsd:documentation>This attribute shall reference a supported module
type.</xsd:documentation>
     </xsd:annotation>
    </xsd:attribute>
   </xsd:complexType>
```



```
  </xsd:element>
  <xsd:element name="moduleTypeList">
   <xsd:annotation>
    <xsd:documentation>This element shall contain a list of supported module
types.</xsd:documentation>
   </xsd:annotation>
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="moduleType" maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="fileList">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element ref="file"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

A.1.4 XML schema: ProfileBody_CommunicationNetwork_CANopen.xsd

This schema includes the schema “ISO15745ProfileContainer.xsd” in A.1.1 and the schema “CommonElements.xsd” in A.1.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.canopen.org/xml/1.1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.canopen.org/xml/1.1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0">
 <xsd:include schemaLocation="ISO15745ProfileContainer.xsd"/>
 <xsd:include schemaLocation="CommonElements.xsd"/>
 <!-- CANopen profile body communication network-->
 <xsd:complexType name="ProfileBody_CommunicationNetwork_CANopen">
  <xsd:complexContent>
   <xsd:extension base="ProfileBody_DataType">
    <xsd:choice>
     <xsd:sequence>
      <!--################ CANopen application layers -->
      <!--################ CANopen transport layers -->
      <xsd:element name="ApplicationLayers">
       <xsd:annotation>
        <xsd:documentation>The CANopen ApplicationLayers element represents the
combined profiles for the upper 3 OSI layers of the CANopen communication network
integration model.</xsd:documentation>
       </xsd:annotation>
       <xsd:complexType>
        <xsd:sequence>
         <xsd:element name="identity" minOccurs="0">
          <xsd:complexType>
           <xsd:sequence>
            <xsd:element ref="vendorID" minOccurs="0"/>
            <xsd:element ref="deviceFamily" minOccurs="0"/>
            <xsd:element ref="productID" minOccurs="0"/>
            <xsd:element ref="version" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="buildDate" minOccurs="0"/>
            <xsd:element ref="specificationRevision" minOccurs="0"/>
           </xsd:sequence>
          </xsd:complexType>
         </xsd:element>
         <xsd:element ref="CANopenObjectList"/>
         <xsd:element name="dummyUsage" minOccurs="0">
          <xsd:complexType>
           <xsd:sequence>
            <xsd:element name="dummy" maxOccurs="unbounded">
             <xsd:annotation>
              <xsd:documentation>This element is used to enable and disable
certain dummy entries for the dummy mapping.</xsd:documentation>
             </xsd:annotation>
             <xsd:complexType>
              <xsd:attribute name="entry" use="required">
               <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                 <xsd:enumeration value="Dummy0001=0"/>
                 <xsd:enumeration value="Dummy0002=0"/>
                 <xsd:enumeration value="Dummy0003=0"/>
                 <xsd:enumeration value="Dummy0004=0"/>
```



```
                        <xsd:enumeration value="Dummy0005=0"/>
                        <xsd:enumeration value="Dummy0006=0"/>
                        <xsd:enumeration value="Dummy0007=0"/>
                        <xsd:enumeration value="Dummy0001=1"/>
                        <xsd:enumeration value="Dummy0002=1"/>
                        <xsd:enumeration value="Dummy0003=1"/>
                        <xsd:enumeration value="Dummy0004=1"/>
                        <xsd:enumeration value="Dummy0005=1"/>
                        <xsd:enumeration value="Dummy0006=1"/>
                        <xsd:enumeration value="Dummy0007=1"/>
                      </xsd:restriction>
                    </xsd:simpleType>
                  </xsd:attribute>
                </xsd:complexType>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="dynamicChannels" minOccurs="0">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="dynamicChannel" maxOccurs="unbounded">
                <xsd:annotation>
                  <xsd:documentation>This element is used to mark available channels
that can be used to create a link between the data to be communicated in the CANopen
network and the application program running on the device.</xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                  <xsd:attribute name="dataType" type="xsd:hexBinary" use="required">
                    <xsd:annotation>
                      <xsd:documentation>CANopen data type (two hex
digits).</xsd:documentation>
                    </xsd:annotation>
                  </xsd:attribute>
                  <xsd:attribute name="accessType" use="required">
                    <xsd:annotation>
                      <xsd:documentation>Access type of the object; valid values:
- readOnly
- writeOnly
- readWriteOutput</xsd:documentation>
                    </xsd:annotation>
                    <xsd:simpleType>
                      <xsd:restriction base="xsd:NMTOKEN">
                        <xsd:enumeration value="readOnly"/>
                        <xsd:enumeration value="writeOnly"/>
                        <xsd:enumeration value="readWriteOutput"/>
                      </xsd:restriction>
                    </xsd:simpleType>
                  </xsd:attribute>
                  <xsd:attribute name="startIndex" type="xsd:hexBinary"
use="required">
                    <xsd:annotation>
                      <xsd:documentation>The index of the first object in the object
dictionary used by the proccess image.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:attribute>
                  <xsd:attribute name="endIndex" type="xsd:hexBinary" use="required">
                    <xsd:annotation>
                      <xsd:documentation>The index of the last object in the object
dictionary used by the process image.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:attribute>
                  <xsd:attribute name="maxNumber" type="xsd:unsignedInt"
use="required">
                    <xsd:annotation>
                      <xsd:documentation>The maximum number of objects, which can be
allocated in this segment.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:attribute>
                  <xsd:attribute name="addressOffset" type="xsd:hexBinary"
use="required">
                    <xsd:annotation>
                      <xsd:documentation>The content is the offset of the segment
inside the process image.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:attribute>
                  <xsd:attribute name="bitAlignment" type="xsd:unsignedByte"
use="optional">
                    <xsd:annotation>
```



```
                    <xsd:documentation>The content defines the bit alignment used.
Most often this will be 1 (bit alignment), 8 (byte alignment), 16 (word alignment), or
32 (double word alignment).</xsd:documentation>
                  </xsd:annotation>
                </xsd:attribute>
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="moduleManagement" minOccurs="0">
        <xsd:complexType>
          <xsd:choice>
            <xsd:element name="interfaceList">
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="interface" maxOccurs="unbounded">
                    <xsd:complexType>
                      <xsd:sequence>
                        <xsd:element name="rangeList">
                          <xsd:complexType>
                            <xsd:sequence>
                              <xsd:element name="range" maxOccurs="unbounded">
                                    <xsd:complexType>
                                          <xsd:attribute name="name" type="xsd:string"
use="required"/>
                                          <xsd:attribute name="moduleType"
type="xsd:ID" use="required"/>
                                          <xsd:attribute name="baseIndex"
type="xsd:hexBinary" use="required">
                                                <xsd:annotation>
                                                      <xsd:documentation>This
attribute shall define the start index for the range.</xsd:documentation>
                                                </xsd:annotation>
                                          </xsd:attribute>
                                          <xsd:attribute name="maxIndex"
type="xsd:hexBinary" use="required">
                                                <xsd:annotation>
                                                      <xsd:documentation>This
attribute shall define the last possible index for the range.</xsd:documentation>
                                                </xsd:annotation>
                                          </xsd:attribute>
                                          <xsd:attribute name="maxSubIndex"
type="xsd:hexBinary" use="optional">
                                                <xsd:annotation>
                                                      <xsd:documentation>If the
object is of sortMode subIndex, then this attribute shall define the last possible sub-
index for the range before the next index value shall be used.</xsd:documentation>
                                                </xsd:annotation>
                                          </xsd:attribute>
                                          <xsd:attribute name="sortMode"
use="required">
                                                <xsd:annotation>
                                                      <xsd:documentation>This element
defines that the new (sub-)object shall be added with a new index or subindex; valid
values:
- index
- subIndex</xsd:documentation>
                                                </xsd:annotation>
                                                <xsd:simpleType>
                                                      <xsd:restriction
base="xsd:NMTOKEN">
                                                            <xsd:enumeration
value="index"/>
                                                            <xsd:enumeration
value="subIndex"/>
                                                      </xsd:restriction>
                                                </xsd:simpleType>
                                          </xsd:attribute>
                                          <xsd:attribute name="sortNumber"
use="required">
                                                <xsd:annotation>
                                                      <xsd:documentation>This element
defines that the index of the new (sub-)object; valid values:
- continous - next free index
- address - specific address</xsd:documentation>
                                                </xsd:annotation>
                                                <xsd:simpleType>
```



```
                                                    <xsd:restriction
base="xsd:NMTOKEN">
                                                        <xsd:enumeration
value="continuous"/>
                                                        <xsd:enumeration
value="address"/>
                                                    </xsd:restriction>
                                                </xsd:simpleType>
                                            </xsd:attribute>
                                            <xsd:attribute name="sortStep"
type="xsd:hexBinary" use="optional" default="1">
                                                <xsd:annotation>
                                                    <xsd:documentation>This shall
define the step width of the new (sub-)index, if sortNumber is set to
continous.</xsd:documentation>
                                                </xsd:annotation>
                                            </xsd:attribute>
                                        </xsd:complexType>
                                    </xsd:element>
                                   </xsd:sequence>
                                  </xsd:complexType>
                                 </xsd:element>
                                </xsd:sequence>
                                <xsd:attribute name="uniqueIDRef" type="xsd:IDREF"
use="required">
                                 <xsd:annotation>
                                  <xsd:documentation>This attribute shall reference the module
interface.</xsd:documentation>
                                 </xsd:annotation>
                                </xsd:attribute>
                                <xsd:attribute name="addressing" use="required">
                                 <xsd:annotation>
                                  <xsd:documentation>This attribute shall indicate if the
connected modules are addressed automatically or manually.</xsd:documentation>
                                 </xsd:annotation>
                                 <xsd:simpleType>
                                  <xsd:restriction base="xsd:NMTOKEN">
                                   <xsd:enumeration value="auto"/>
                                   <xsd:enumeration value="manual"/>
                                  </xsd:restriction>
                                 </xsd:simpleType>
                                </xsd:attribute>
                               </xsd:complexType>
                              </xsd:element>
                             </xsd:sequence>
                            </xsd:complexType>
                           </xsd:element>
                           <xsd:element name="moduleInterface">
                            <xsd:annotation>
                             <xsd:documentation>This element shall be by a single module to
identify the module and to describe its properties. </xsd:documentation>
                            </xsd:annotation>
                            <xsd:complexType>
                             <xsd:attribute name="childIDRef" type="xsd:NCName" use="required">
                              <xsd:annotation>
                               <xsd:documentation>This attribute shall indicate the unique ID of
the referenced module.</xsd:documentation>
                              </xsd:annotation>
                             </xsd:attribute>
                             <xsd:attribute name="addressMin" type="xsd:positiveInteger"
use="required">
                              <xsd:annotation>
                               <xsd:documentation>This attribute shall indicate the minimum
address the module is allowed to use.</xsd:documentation>
                              </xsd:annotation>
                             </xsd:attribute>
                             <xsd:attribute name="addressMax" type="xsd:positiveInteger"
use="optional">
                              <xsd:annotation>
                               <xsd:documentation>This attribute shall indicate the maximum
address the module is allowed to use.</xsd:documentation>
                              </xsd:annotation>
                             </xsd:attribute>
                             <xsd:attribute name="addressing" use="optional" default="inherit">
                              <xsd:annotation>
                               <xsd:documentation>This attribute shall indicate idicate the
addressing for the module. If present, it shall override the definition of the
interface.</xsd:documentation>
                              </xsd:annotation>
```



```
                    <xsd:simpleType>
                     <xsd:restriction base="xsd:NMTOKEN">
                      <xsd:enumeration value="inherit"/>
                      <xsd:enumeration value="auto"/>
                      <xsd:enumeration value="manual"/>
                     </xsd:restriction>
                    </xsd:simpleType>
                   </xsd:attribute>
                  </xsd:complexType>
                 </xsd:element>
                </xsd:choice>
               </xsd:complexType>
              </xsd:element>
             </xsd:sequence>
             <xsd:attribute name="conformanceClass" type="xsd:string" use="optional">
              <xsd:annotation>
               <xsd:documentation>Conformance class of the device
type.</xsd:documentation>
              </xsd:annotation>
             </xsd:attribute>
             <xsd:attribute name="communicationEntityType" type="xsd:NMTOKENS"
default="slave">
              <xsd:annotation>
               <xsd:documentation>Type of the communication entity - if several types
are supported, their names shall be separated by a blank character, e.g. "master
slave"; the type names shall be chosen out of the following enumarion of names:
- slave (default)
- master
- server
- client
- interconnection (e.g. interconnection is a gateway functionality)
- peer (communication entity, which acts as a client and server)</xsd:documentation>
              </xsd:annotation>
             </xsd:attribute>
            </xsd:complexType>
           </xsd:element>
           <xsd:element name="TransportLayers">
            <xsd:annotation>
             <xsd:documentation>The CANopen TransportLayers element represents the
combined profiles for the lower 4 OSI layers of the CANopen communication network
integration model.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
             <xsd:sequence>
              <xsd:element name="PhysicalLayer">
               <xsd:complexType>
                <xsd:sequence>
                 <xsd:element name="baudRate">
                  <xsd:complexType>
                   <xsd:sequence>
                    <xsd:element name="supportedBaudRate" maxOccurs="unbounded">
                     <xsd:complexType>
                      <xsd:attribute name="value" use="required">
                       <xsd:annotation>
                        <xsd:documentation>One out of the set of bit rate values
supported by the device type; valid values:
- 10 Kbps
- 20 Kbps
- 50 Kbps
- 100 Kbps (This value is not offically defined by CAN in Automation specifications.)
- 125 Kbps
- 250 Kbps
- 500 Kbps
- 800 Kbps
- 1000 Kbps
- auto-baudRate</xsd:documentation>
                       </xsd:annotation>
                       <xsd:simpleType>
                        <xsd:restriction base="xsd:string">
                         <xsd:enumeration value="10 Kbps"/>
                         <xsd:enumeration value="20 Kbps"/>
                         <xsd:enumeration value="50 Kbps"/>
                         <xsd:enumeration value="100 Kbps"/>
                         <xsd:enumeration value="125 Kbps"/>
                         <xsd:enumeration value="250 Kbps"/>
                         <xsd:enumeration value="500 Kbps"/>
                         <xsd:enumeration value="800 Kbps"/>
                         <xsd:enumeration value="1000 Kbps"/>
                         <xsd:enumeration value="auto-baudRate"/>
```



```
                        </xsd:restriction>
                       </xsd:simpleType>
                      </xsd:attribute>
                     </xsd:complexType>
                    </xsd:element>
                   </xsd:sequence>
                   <xsd:attribute name="defaultValue" default="250 Kbps">
                    <xsd:annotation>
                     <xsd:documentation>Default value for the bit rate of the device
type - this value is preset to 250 Kbps; other valid values are:
- 10 Kbps
- 20 Kbps
- 50 Kbps
- 100 Kbps (This value is not offically defined by CAN in Automation specifications.)
- 125 Kbps
- 250 Kbps
- 500 Kbps
- 800 Kbps
- 1000 Kbps
- auto-baudRate</xsd:documentation>
                    </xsd:annotation>
                    <xsd:simpleType>
                     <xsd:restriction base="xsd:string">
                      <xsd:enumeration value="10 Kbps"/>
                      <xsd:enumeration value="20 Kbps"/>
                      <xsd:enumeration value="50 Kbps"/>
                      <xsd:enumeration value="100 Kbps"/>
                      <xsd:enumeration value="125 Kbps"/>
                      <xsd:enumeration value="250 Kbps"/>
                      <xsd:enumeration value="500 Kbps"/>
                      <xsd:enumeration value="800 Kbps"/>
                      <xsd:enumeration value="1000 Kbps"/>
                      <xsd:enumeration value="auto-baudRate"/>
                     </xsd:restriction>
                    </xsd:simpleType>
                   </xsd:attribute>
                  </xsd:complexType>
                 </xsd:element>
                </xsd:sequence>
               </xsd:complexType>
              </xsd:element>
             </xsd:sequence>
            </xsd:complexType>
           </xsd:element>
           <!--################ CANopen network management -->
           <xsd:element name="NetworkManagement" minOccurs="0">
            <xsd:annotation>
             <xsd:documentation>The CANopen NetworkManagement element represents the
network configuration and performance adjustment capabilities of the CANopen
communication network integration model.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
             <xsd:sequence>
              <xsd:element name="CANopenGeneralFeatures">
               <xsd:complexType>
                <xsd:attribute name="groupMessaging" type="xsd:boolean"
default="false"/>
                <xsd:attribute name="dynamicChannels" type="xsd:unsignedByte"
default="0"/>
                <xsd:attribute name="selfStartingDevice" type="xsd:boolean"
default="false"/>
                <xsd:attribute name="SDORequestingDevice" type="xsd:boolean"
default="false"/>
                <xsd:attribute name="granularity" type="xsd:unsignedByte"
use="required"/>
                <xsd:attribute name="nrOfRxPDO" type="xsd:unsignedShort" default="0"/>
                <xsd:attribute name="nrOfTxPDO" type="xsd:unsignedShort" default="0"/>
                <xsd:attribute name="bootUpSlave" type="xsd:boolean" default="false"/>
                <xsd:attribute name="layerSettingServiceSlave" type="xsd:boolean"
default="false"/>
               </xsd:complexType>
              </xsd:element>
              <xsd:element name="CANopenMasterFeatures" minOccurs="0">
               <xsd:complexType>
                <xsd:attribute name="bootUpMaster" type="xsd:boolean" default="false"/>
                <xsd:attribute name="flyingMaster" type="xsd:boolean" default="false"/>
                <xsd:attribute name="SDOManager" type="xsd:boolean" default="false"/>
                <xsd:attribute name="configurationManager" type="xsd:boolean"
default="false"/>
```



```
              <xsd:attribute name="layerSettingServiceMaster" type="xsd:boolean"
default="false"/>
             </xsd:complexType>
            </xsd:element>
            <xsd:element name="deviceCommissioning" minOccurs="0">
             <xsd:complexType>
              <xsd:attribute name="nodeID" type="xsd:unsignedByte" use="required"/>
              <xsd:attribute name="nodeName" type="xsd:string" use="required"/>
              <xsd:attribute name="actualBaudRate" type="xsd:string" use="required"/>
              <xsd:attribute name="networkNumber" type="xsd:unsignedLong"
use="required"/>
              <xsd:attribute name="networkName" type="xsd:string" use="required"/>
              <xsd:attribute name="CANopenManager" type="xsd:boolean" use="required"/>
             </xsd:complexType>
            </xsd:element>
           </xsd:sequence>
          </xsd:complexType>
         </xsd:element>
        </xsd:sequence>
        <xsd:element name="ExternalProfileHandle" type="ProfileHandle_DataType">
         <xsd:annotation>
          <xsd:documentation>The additional ExternalProfileHandle element enables
external non-XML data to be referenced.</xsd:documentation>
         </xsd:annotation>
        </xsd:element>
       </xsd:choice>
       <xsd:attributeGroup ref="ag_formatAndFile"/>
       <xsd:attribute name="supportedLanguages" type="xsd:NMTOKENS" use="optional">
        <xsd:annotation>
         <xsd:documentation>List of supported languages.</xsd:documentation>
        </xsd:annotation>
       </xsd:attribute>
     </xsd:extension>
   </xsd:complexContent>
  </xsd:complexType>
  <!--################ CANopen object dictionary -->
  <xsd:element name="CANopenObjectList">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="CANopenObject" maxOccurs="unbounded">
      <xsd:complexType>
       <xsd:sequence>
        <xsd:element name="CANopenSubObject" minOccurs="0" maxOccurs="255">
         <xsd:complexType>
          <xsd:attribute name="subIndex" type="xsd:hexBinary" use="required">
           <xsd:annotation>
            <xsd:documentation>Subindex of the sub-object (two hex
digits).</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="name" type="xsd:string" use="required">
           <xsd:annotation>
            <xsd:documentation>Name of the sub-object.</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="objectType" type="xsd:unsignedByte" use="required">
           <xsd:annotation>
            <xsd:documentation>CANopen object type.</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="dataType" type="xsd:hexBinary" use="optional">
           <xsd:annotation>
            <xsd:documentation>CANopen data type (two hex
digits).</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="lowLimit" type="xsd:string" use="optional">
           <xsd:annotation>
            <xsd:documentation>Low limit of the parameter value.</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="highLimit" type="xsd:string" use="optional">
           <xsd:annotation>
            <xsd:documentation>High limit of the parameter
value.</xsd:documentation>
           </xsd:annotation>
          </xsd:attribute>
          <xsd:attribute name="accessType" use="optional">
           <xsd:annotation>
```



```
              <xsd:documentation>Access type of the sub-object; valid values:
- const - read access only; the value is not changing
- ro - read access only
- wo - write access only
- rw - both read and write access</xsd:documentation>
             </xsd:annotation>
             <xsd:simpleType>
              <xsd:restriction base="xsd:string">
               <xsd:enumeration value="ro"/>
               <xsd:enumeration value="wo"/>
               <xsd:enumeration value="rw"/>
               <xsd:enumeration value="const"/>
              </xsd:restriction>
             </xsd:simpleType>
            </xsd:attribute>
            <xsd:attribute name="defaultValue" type="xsd:string" use="optional">
             <xsd:annotation>
              <xsd:documentation>Default value of the sub-object.</xsd:documentation>
             </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="actualValue" type="xsd:string" use="optional">
             <xsd:annotation>
              <xsd:documentation>Actual value of the sub-object.</xsd:documentation>
             </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="denotation" type="xsd:string" use="optional">
             <xsd:annotation>
              <xsd:documentation>Application specific name of the sub-
object.</xsd:documentation>
             </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="PDOmapping" use="optional">
             <xsd:annotation>
              <xsd:documentation>PDO mapping of the sub-object; valid values:
- no - not mappable
- default - mapped by default
- optional - optionally mapped
- TPDO - may be mapped into TPDO only
- RPDO - may be mapped into TPDO only</xsd:documentation>
             </xsd:annotation>
             <xsd:simpleType>
              <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="no"/>
               <xsd:enumeration value="default"/>
               <xsd:enumeration value="optional"/>
               <xsd:enumeration value="TPDO"/>
               <xsd:enumeration value="RPDO"/>
              </xsd:restriction>
             </xsd:simpleType>
            </xsd:attribute>
            <xsd:attribute name="objFlags" type="xsd:hexBinary" use="optional">
             <xsd:annotation>
              <xsd:documentation>Controls the behavior of tools (four hex digits);
valid values:
- bit 0: 0 - write on download allowed; 1 - write on download not allowed
- bit 1: 0 - read on upload allowed; 1 - read on upload not allowed
- bit 2: 0 - change of value takes effect immediatly; 1 - change of value takes effect
after reset
- bit 3 to 31: reserved (0)
NOTE: This shall prevent unintended read or write access by generic tools in case a
parameter implements a specific function. A read or write in such cases may trigger a
specific function in the CANopen device.</xsd:documentation>
             </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="optional">
             <xsd:annotation>
              <xsd:documentation>Unique ID of an appropriate element in the
application process part referenced from this object; if the attribute is present, the
attributes dataType, lowLimit, highLimit, accessType, and defaultValue shall be defined
by the referenced element in the application process part.</xsd:documentation>
             </xsd:annotation>
            </xsd:attribute>
            <xsd:attribute name="rangeSelector" type="xsd:string" use="optional">
             <xsd:annotation>
              <xsd:documentation>If the sub-object is part of a modular device, this
references the index range as defined by the CANopenIndexRange element that should be
used for the index of the sub-object.</xsd:documentation>
             </xsd:annotation>
            </xsd:attribute>
```



```
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="index" type="xsd:hexBinary" use="required">
          <xsd:annotation>
            <xsd:documentation>Index of the object (four hex
digits).</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="name" type="xsd:string" use="required">
          <xsd:annotation>
            <xsd:documentation>Name of the object.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="objectType" type="xsd:unsignedByte" use="required">
          <xsd:annotation>
            <xsd:documentation>CANopen object type.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="dataType" type="xsd:hexBinary" use="optional">
          <xsd:annotation>
            <xsd:documentation>CANopen data type (two hex digits).</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="lowLimit" type="xsd:string" use="optional">
          <xsd:annotation>
            <xsd:documentation>Low limit of the parameter value.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="highLimit" type="xsd:string" use="optional">
          <xsd:annotation>
            <xsd:documentation>High limit of the parameter value.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="accessType" use="optional">
          <xsd:annotation>
            <xsd:documentation>Access type of the object; valid values:
- const - read access only; the value is not changing
- ro - read access only
- wo - write access only
- rw - both read and write access</xsd:documentation>
          </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="ro"/>
              <xsd:enumeration value="wo"/>
              <xsd:enumeration value="rw"/>
              <xsd:enumeration value="const"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="defaultValue" type="xsd:string" use="optional">
          <xsd:annotation>
            <xsd:documentation>Default value of the object.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="actualValue" type="xsd:string" use="optional">
          <xsd:annotation>
            <xsd:documentation>Actual value of the object.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="denotation" type="xsd:string" use="optional">
          <xsd:annotation>
            <xsd:documentation>Application specific name of the
object.</xsd:documentation>
          </xsd:annotation>
        </xsd:attribute>
        <xsd:attribute name="PDOmapping" use="optional">
          <xsd:annotation>
            <xsd:documentation>PDO mapping of the object; valid values:
- no - not mappable
- default - mapped by default
- optional - optionally mapped
- TPDO - may be mapped into TPDO only
- RPDO - may be mapped into TPDO only</xsd:documentation>
          </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
              <xsd:enumeration value="no"/>
```



```
            <xsd:enumeration value="default"/>
            <xsd:enumeration value="optional"/>
            <xsd:enumeration value="TPDO"/>
            <xsd:enumeration value="RPDO"/>
           </xsd:restriction>
          </xsd:simpleType>
         </xsd:attribute>
         <xsd:attribute name="objFlags" type="xsd:hexBinary" use="optional">
          <xsd:annotation>
           <xsd:documentation>Controls the behavior of tools (four hex digits); valid
values:
- bit 0: 0 - write on download allowed; 1 - write on download not allowed
- bit 1: 0 - read on upload allowed; 1 - read on upload not allowed
- bit 2: 0 - change of value takes effect immediatly; 1 - change of value takes effect
after reset
- bit 3 to 31: reserved (0)
NOTE: This shall prevent unintended read or write access by generic tools in case a
parameter implements a specific function. A read or write in such cases may trigger a
specific function in the CANopen device.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
         <xsd:attribute name="uniqueIDRef" type="xsd:IDREF" use="optional">
          <xsd:annotation>
           <xsd:documentation>Unique ID of an appropriate element in the application
process part referenced from this object; if the attribute is present, the attributes
dataType, lowLimit, highLimit, accessType, and defaultValue shall be defined by the
referenced element in the application process part.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
         <xsd:attribute name="subNumber" type="xsd:unsignedByte" use="optional">
          <xsd:annotation>
           <xsd:documentation>Number of sub-objects of the object.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
         <xsd:attribute name="rangeSelector" type="xsd:string" use="optional">
          <xsd:annotation>
           <xsd:documentation>If the object is part of a modular device, this
references the index range as defined by the CANopenIndexRange element that should be
used for the index of the object.</xsd:documentation>
          </xsd:annotation>
         </xsd:attribute>
        </xsd:complexType>
       </xsd:element>
      </xsd:sequence>
      <xsd:attribute name="mandatoryObjects" type="xsd:unsignedInt" use="optional">
       <xsd:annotation>
        <xsd:documentation>Number of mandatory objects in the
dictionary.</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="optionalObjects" type="xsd:unsignedInt" use="optional">
       <xsd:annotation>
        <xsd:documentation>Number of optional objects in the
dictionary.</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="manufacturerObjects" type="xsd:unsignedInt" use="optional">
       <xsd:annotation>
        <xsd:documentation>Number of manufacturer-specific objects in the
dictionary.</xsd:documentation>
       </xsd:annotation>
      </xsd:attribute>
    </xsd:complexType>
   </xsd:element>
</xsd:schema>
```


Annex B — Normative

B.1 Value syntax

B.1.1 General

The values given are formatted as `xsd:string` in terms of XML. Values representing other data types than string values, e.g. numerical values, shall be given in accordance to the definition of /IEC61131-3/.

NOTE The definition of hexadecimal values is different from the definition of /IEC61131-3/. The definition is according to ANSI C.

boolean_value	::= 'true' | 'false'
real_value	::= integer_value '.' unsigned_value [exponent]
exponent	::= ('E' | 'e') integer_value
integer_value	::= ['+' | '–'] unsigned_value
unsigned_value	:= digit { digit }
hexadecimal_value	::= '0' 'x' hex_digit { hex_digit }
hex_word	::= hex_byte hex_byte
hex_byte	::= hex_digit hex_digit
hex_digit	::= digit | hex_letter
hex_letter	::= 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'A' | 'B' | 'C' | 'D' | 'E' | 'F'
digit	::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'

B.1.2 Value syntax for DOMAIN and OCTET_STRING

Values of the CANopen defined data type DOMAIN or OCTET_STRING shall be represented as sequence of hex_byte or shall reference a file as uniform resource identifier (URI). The URI schema shall be "file", for example "file://data.bin" if the referenced file name is "data.bin" and resides at the same location like the XDP, XDD or XDC file.

B.1.3 Value syntax for VISIBLE_STRING

Values of the CANopen defined data type VISIBLE_STRING shall be represented as a regular string according to /CiA301/.

B.1.4 Value syntax for UNICODE_STRING

Values of the CANopen defined data type UNICODE_STRING shall be represented as a sequence of hex_word.

B.2 XML file content convention

The XML file content shall be encoded in UTF8.

B.3 XML file naming convention

The XML schema defined in this document applies to three different XML files

— the profile definition file,

— the device description file, and

— the device configuration file.

The profile definition file is an XML representation of a CANopen framework, CANopen device profile or CANopen application profile. The extension shall be "XPD".

The device description file is an XML file describing the device type of a CANopen device. The extension shall be "XDD". The name of the XML file shall contain the vendor-ID of the CANopen device in the form of 8 hexadecimal digits. The vendor-ID may be used at any position in the name. Underscores shall be used to separate the vendor-ID from the remaining parts of the name, e.g. if the vendor-ID is 12345678_h the vendor-ID appears in the XML file name as "12345678_name.XDD", "name1_12345678_name2.XDD", or "name_12345678.XDD".

The device configuration file is an XML file describing a configured CANopen device. The extension shall be "XDC".

No other characters than the following named characters are allowed in the XML file. The file name shall not be case sensitive.

The XML file name of the text resource file shall be built using the same rules as defined for the XML file name of the CANopen device description.

filename_XPD	::=	character { character | '_' | '-' } '.' ('XPD' | 'xpd')
filename_XDD	::=	[prefix] vendor_id [postfix] '.' ('XDD' | 'xdd')
filename_XDC	::=	character { character | '_' | '-' } '.' ('XDC' | 'xdc')
filename_textresource	::=	[prefix] vendor_id [postfix] '.' character { character }
prefix	::=	character { character | '-' } '_'
postfix	::=	'_' { character | '-' } character
vendor_id	::=	8 * hex_digit
character	::=	letter | digit
letter	::=	lowercase_letter | uppercase_letter
lowercase_letter	::=	'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'
uppercase_letter	::=	'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'
hex_digit	::=	digit | hex_letter
digit	::=	'0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'

hex_letter ::= 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'A' | 'B' | 'C' | 'D' | 'E' | 'F'

B.4 Text resource

B.4.1 General

A text resource file contains exactly one textResource element. The textResource element has a lang attribute, which designates the language of all text entries within the text resource file.

Figure B.1 shows the structure of the textResource element.

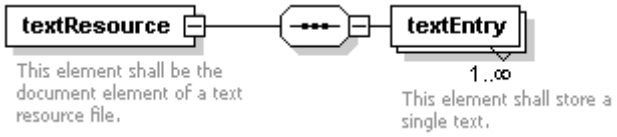

Figure B.1 — textResource element

The textResource element has the attributes given in Table B.1.

Table B.1 — Attributes of element textResource

Attribute	Data type	Use	Description
lang	xsd:language	required	Lanugage of all text entries within the text resource file

B.4.2 textEntry

The textEntry contains the text data as content.

The textEntry element has the attributes given in Table B.2.

Table B.2 — Attributes of element textEntry

Attribute	Data type	Use	Description
textID	xsd:string	required	Text identifier of the text entry

B.4.3 Text resource schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.canopen.org/xml/1.1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.canopen.org/xml/1.1" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
 <xsd:element name="textResource">
  <xsd:annotation>
   <xsd:documentation>This element shall be the document element of a text resource
file.</xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="textEntry" maxOccurs="unbounded">
     <xsd:annotation>
      <xsd:documentation>This element shall store a single text.</xsd:documentation>
     </xsd:annotation>
     <xsd:complexType>
      <xsd:simpleContent>
       <xsd:extension base="xsd:string">
        <xsd:attribute name="textID" type="xsd:string" use="required">
         <xsd:annotation>
          <xsd:documentation>The unique identifier of the text entry that is
referenced by the according entry in the CANopen XML file.</xsd:documentation>
         </xsd:annotation>
        </xsd:attribute>
       </xsd:extension>
      </xsd:simpleContent>
     </xsd:complexType>
    </xsd:element>
   </xsd:sequence>
   <xsd:attribute name="lang" type="xsd:language" use="required">
    <xsd:annotation>
     <xsd:documentation>Defines the language used in the text resource
file.</xsd:documentation>
    </xsd:annotation>
   </xsd:attribute>
  </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

