EPSG Draft Standard 311

Ethernet POWERLINK

XML Device Description

Version 1.2.1

B&R Industrial Automation GmbH

POWERLINK-Office
B&R Straße 1
5142 Eggelsberg
Austria

powerlink.office@br-automation.com
www.br-automation.com/en/technologies/powerlink/

Disclaimer

Patent notice

History

Vers.	Date	Author(s)		Description
0.0.1	2006.03.20	F.Matejka	B&R	Document started
0.0.2	2006.04.13	F.Matejka	B&R	First public release of the document
0.0.3	2006.05.09	F.Matejka	B&R	Corrected typos. Removed access types rwr and rww (these are not supported by EPL).
0.0.4	2006.06.02	F.Matejka	B&R	Corrected typos. Added descriptions in table “Attributes of element dynamicChannel”. Feedback of Mr. Gedenk.
0.0.5	2006.06.13	F.Matejka	B&R	Editorial changes. Feedback of Ms. Böttger.
0.0.6	2006.06.13	F.Matejka	B&R	Included changes from the CiA WD 311 CANopen XML specification Version 0.4. Main change is the additional text resource file. Two appendices added. Referenced new schema version 0.8.
0.0.7	2006.07.24	F.Matejka	B&R	Corrected 2 typos in Tab. 54 and 55.
0.0.8	2006.11.16	F.Matejka	B&R	Added cover pages. Changed status from white paper to draft standard proposal (approved by TWG ready for 2 month commenting period). File name changed to reflect document status.
0.0.9	2007.03.01	F.Matejka	B&R	Changes as agreed at TWG Meeting 28.02.2007. Considered changes of CiA document from V0.4 to V0.5.2. Included the schema files in the document (like in the CiA document).
0.0.10	2007.03.26	F.Matejka	B&R	Synchronized with CiA DS 311 Version 1.0 (minor editorial changes). Version was not distributed.
1.0.0	2007.10.25	F.Matejka	B&R	Changed status to draft standard. Changed contact address.
1.0.1	2012.09.18	S. Kirchmayer	B&R	Modular IOs
1.0.2	2013.08.20	S. Kirchmayer	B&R	Changes from TWG meeting in 3/13
1.0.2	2014.04.15	S. Kirchmayer	B&R	Status changed to DSP, new office address
1.1.0	2015.12.04	S. Kirchmayer, B. Avramov, C. Ruecker	B&R	Schema listings deleted, Object firmwareList subindex 0 ro bitOffset added References updated connectorList added classificationList added
1.1.1	2016-06-22	S. Kirchmayer, B. Avramov, C. Ruecker	B&R	Modular device profiles added modulManagement added
1.2.0	2016-09-15	S. Kirchmayer	B&R	Status changed to DS
1.2.1	2023-06-02	S. Kirchmayer	B&R	© B&R due to dissolution of EPSG

Content

1 Introduction

This specification defines the device description and device configuration file format for EPSG DS 301, Ethernet POWERLINK Communication Profile Specification [7].

The format is based on the corresponding CANopen specification, "CiA DS 311 CANopen device description Version 1.0". Therefore many parts of the CANopen specification are mirrored in this document.

Alongside this document XML schema files are provided. The schema version 0.15 is described within this document.

The XML schema files are:

- Powerlink_Main.xsd
- CommonElements.xsd
- CommonElements_Modular.xsd
- ISO15745ProfileContainer.xsdProfileBody_Device_Powerlink.xsd
- ProfileBody_CommunicationNetwork_Powerlink.xsd
- ProfileBody_Device_Powerlink_Modular_Head.xsd
- ProfileBody_Device_Powerlink_Modular_Child.xsd
- ProfileBody_CommunicationNetwork_Powerlink.xsd
- ProfileBody_CommunicationNetwork_Powerlink_Modular_Head.xsd
- ProfileBody_CommunicationNetwork_Powerlink_Modular_Child.xsd
- TextResource.xsd

2 Scope

This specification defines the elements and rules for describing device profiles and communication network profiles for devices used in POWERLINK 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.

3 References

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

[2] ISO 15745-1:2003/Amd1:2007, Industrial automation systems and integration – Open systems application integration framework – Part 1: Generic reference description (proposed Amendment 1)

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

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

[5] IEC 61131-3, Programmable controllers – Part 3: Programming languages

[6] ISO 1000, SI units and recommendations for the use of their multiples and of certain other units

[7] EPSG Draft Standard 301 (EPSG DS 301), Ethernet POWERLINK, Communication Profile Specification

[8] EPSG Draft Standard 302-F (EPSG DS 302-F), Ethernet POWERLINK, Part F: Modular Device

4 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).

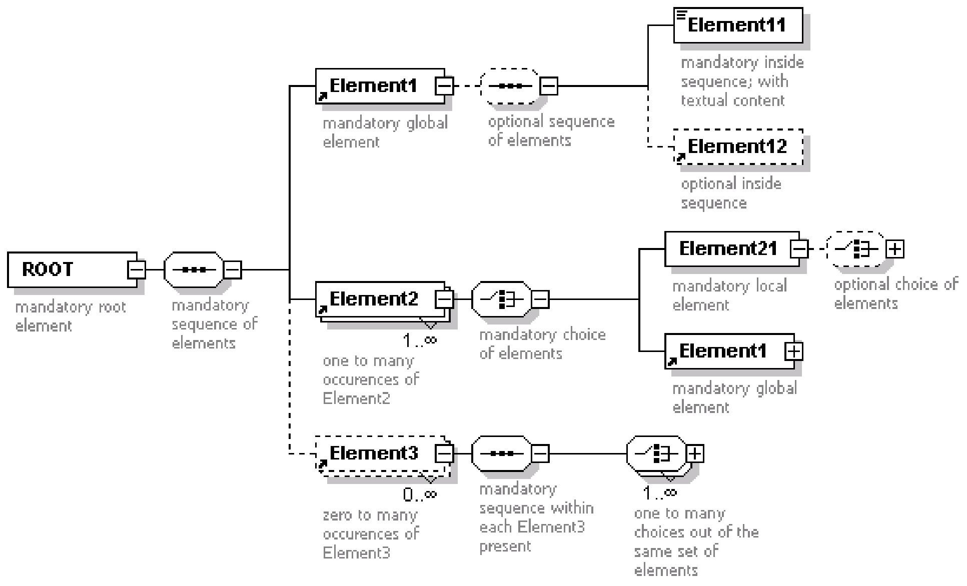
Element11
mandatory inside sequence; with textual content
Element1
mandatory global element
optional sequence of elements
Element12
optional inside sequence
ROOT
mandatory root element
mandatory sequence of elements
Element21
mandatory local element
optional choice of elements
Element2
1..∞
one to many occurences of Element2
mandatory choice of elements
Element1
mandatory global element
Element3
0..∞
zero to many occurences of Element3
mandatory sequence within each Element3 present
1..∞
one to many choices out of the same set of elements

Fig. 1. Notational Conventions

5 Abbreviations

HMI	Human Machine Interface
ISO	International Organization for Standardization
LED	Light Emitting Diode
OSI	Open Systems Interconnection
URI	Uniform Resource Identifier
XML	Extensible Markup Language

6 POWERLINK profiles

6.1 Device profile

6.1.1 General

Fig. 2 shows the element structure of a POWERLINK device profile.

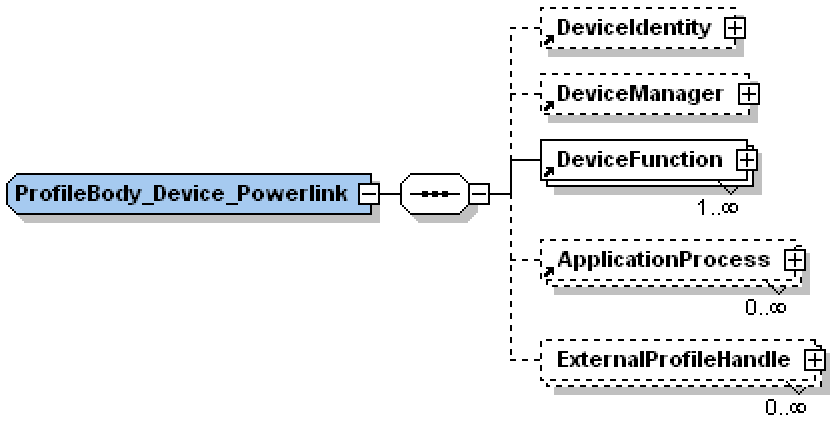

Fig. 2. device profile element

NOTE 1 The additional ExternalProfileHandle element shown in Fig. 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.

6.1.2 Device identity

The DeviceIdentity element contains attributes that are independent of the network and of the process, and which uniquely identify the device (for further information see 7.4.4).

Fig. 3 shows the structure of the DeviceIdentity element.

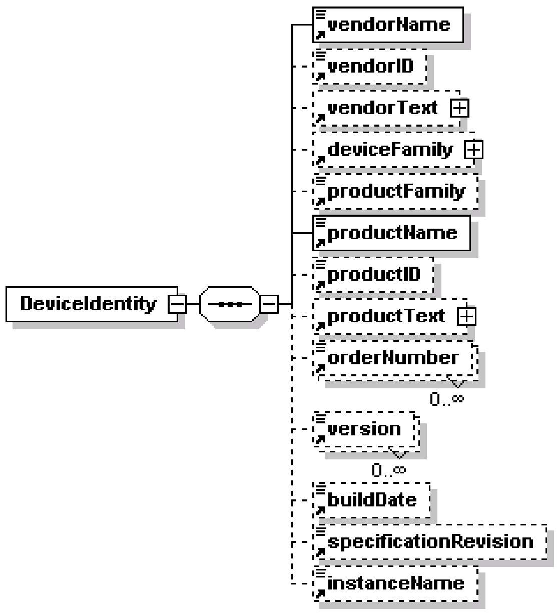

Fig. 3. DeviceIdentity element

6.1.3 Device manager

The DeviceManager element contains attributes and supports services that enable the monitoring of the device.

Fig. 4 shows the structure of the DeviceManager element

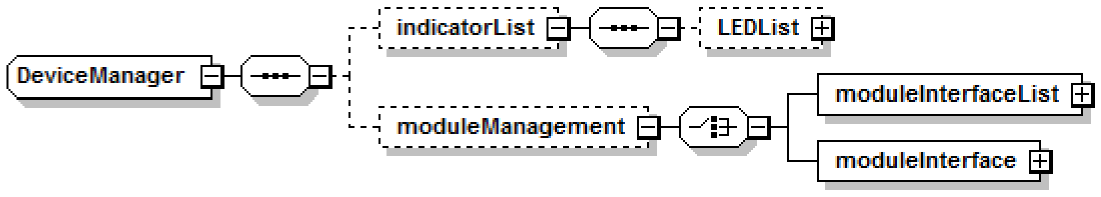

Fig. 4. DeviceManager element

6.1.4 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 (for further information see 7.4.6).

Fig. 5 shows the structure of the DeviceFunction element.

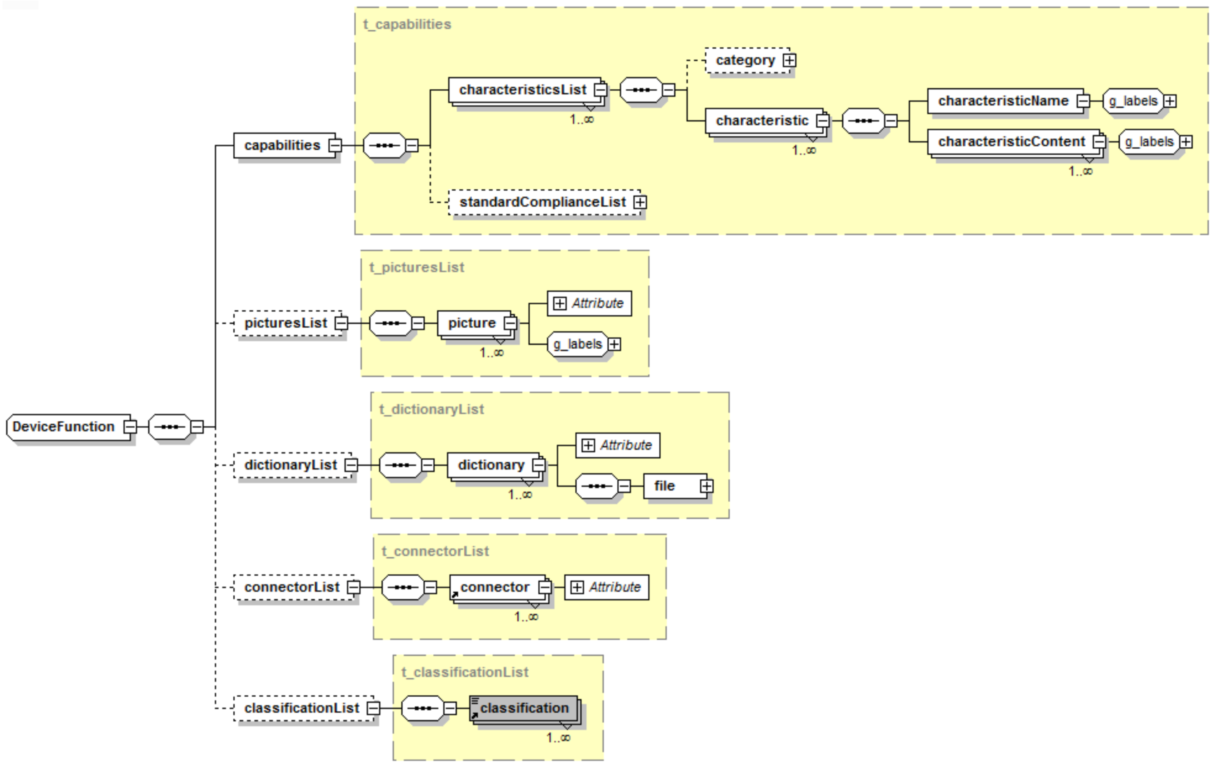

Fig. 5. DeviceFunction element

6.1.5 Application process

The ApplicationProcess element represents the set of services and parameters, which constitute the behaviour and the interfaces of the device in terms of the application, independent of the device technology and the underlying communication networks and communication protocols (see 7.4.6.5).

Fig. 6 shows the structure of the ApplicationProcess element.

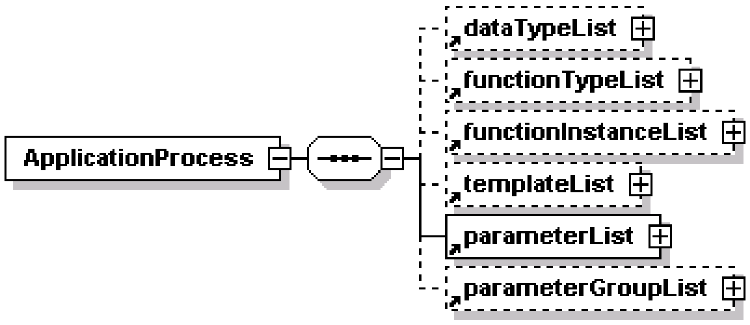

Fig. 6. ApplicationProcess element

6.2 Communication network profile

6.2.1 General

Fig. 7 shows the element structure of the communication network profile.

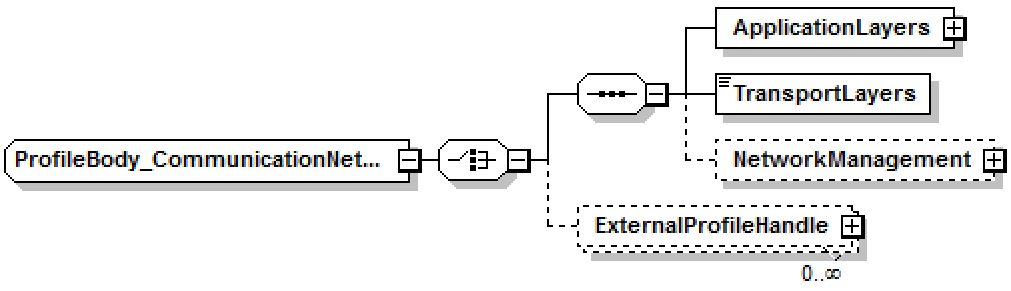

Fig. 7. Communication network profile element

NOTE: The additional ExternalProfileHandle element shown in Fig. 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.

6.2.2 Application layers

The ApplicationLayers element represents POWERLINK application layer, e.g. gives detailed information about a devices object dictionary (for further information see 7.5.4).

6.2.3 Transport layers

The TransportLayers element is present for compatibility to the CANopen XML device description. For POWERLINK this element is empty.

6.2.4 Network management

The NetworkManagement element is used to store specific device features and diagnostic capabilities (for further information see 7.5.6).

7 POWERLINK profile templates

7.1 Overview

POWERLINK is a hard real time communication system based on the Standard IEEE 802.3.

The POWERLINK technology uses the concept of the multi-profile container specified in ISO 15745 1:2005/Amd.1 for XML profile files. Therefore, POWERLINK profile templates are based on the alternate ISO15745ProfileContainer master profile template specified in amendment 1 of ISO 15745 1.

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

- ProfileBody_Device_Powerlink and
- ProfileBody_CommunicationNetwork_Powerlink.

Two types of ProfileBody are defined for POWERLINK modular head node profiles:

- ProfileBody_Device_Powerlink_Modular_Head and
- ProfileBody_CommunicationNetwork_Powerlink_Modular_Head.

Two types of ProfileBody are defined for POWERLINK modular child node profiles:

- ProfileBody_Device_Powerlink_Modular_Child and
- ProfileBody_CommunicationNetwork_Powerlink_Modular_Child.

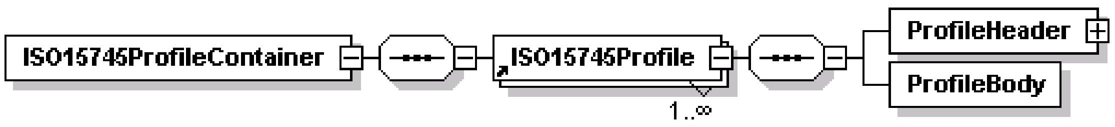

Fig. 8. Master profile template

The ProfileTechnology name according to ISO 15745-1 is POWERLINK.

7.2 General rules

7.2.1 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.

7.2.2 Language support

7.2.2.1 General

Device profiles complying with the XML schema described here 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

7.2.2.2 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 Fig. 9).

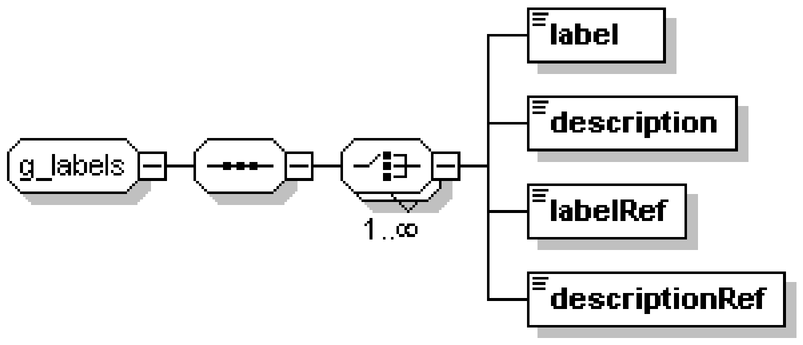

Fig. 9. Group g_labels

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 Tab. 1. The element may appear n times, once for each language. For identifying the language, the lang attribute is used.

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

Tab. 1 Attributes of element label

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

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

Tab. 2 Attributes of element label

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

The labelRef element provides a pointer via its attributes dictID and textID to a text entry in a separate text source file. These text source 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 Tab. 3.

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 by pattern matching

Tab. 3 Attributes of element labelRef

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.

7.2.2.3 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.

7.2.2.4 Attribute lang

The language identifier lang 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). Lang is an attribute of the label element and the description element.

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

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

Tab. 4 Values of attribute lang

7.2.2.5 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”
```

7.2.2.6 URIs

A general mechanism allows of describing a URI in the context of a label element. The URI is implemented via an optional attribute URI.

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

7.3 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 Fig. 10, which is directly inherited from ISO 15745-1.

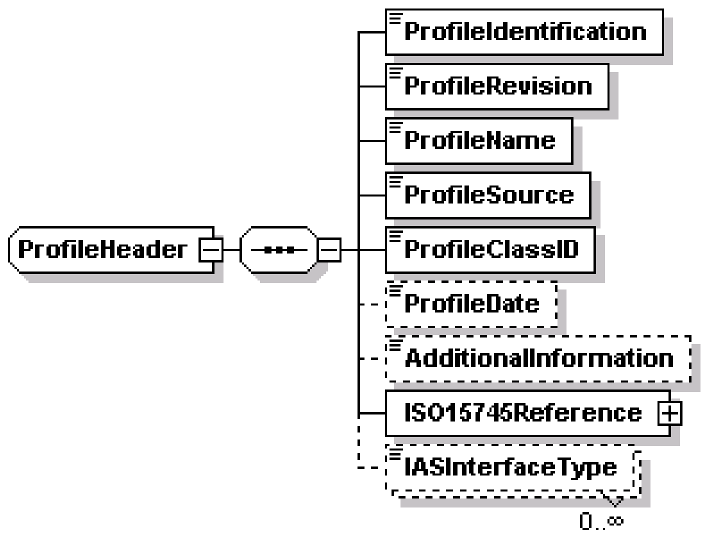

Fig. 10. ProfileHeader element

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 ISO 15745-1,
- the ISO15745Reference element states the ISO 15745 part, edition and technology, to which the description conforms.

7.4 Device profile template description

7.4.1 ProfileBody_Device_Powerlink

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

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 product or product family) configurable (open configurable product that needs an external configurator to create the profile of one instance)
specificationVersion	xsd:string	default	Describes the specification version of the device profile template. Default: 1.0.4

Tab. 5 Attributes of element ProfileBody

7.4.2 ProfileBody_Device_Powerlink_Modular_Head

For the device profile the ProfileBody contains the DeviceIdentity, the DeviceManager, the DeviceFunction and the ApplicationProcess elements shown in Fig. 2. The DeviceFunction element is extended with the moduleManagement for supporting the modular head device functionality. If a device implements a modular head node this profile template shall be used.

The ProfileBody_Device_Powerlink_Modular_Head element contains the attributes given in Tab. 5 and Tab. 6.

Attribute	Data type	Use	Description
specificationVersion	xsd:string	required	Describes the specification version of the device profile template. Fixed: 1.0.4

Tab. 6 Additional attribute of ProfileBody_Device_Powerlink_Modular_Head

7.4.3 ProfileBody_Device_Powerlink _Modular_Child

For the device profile the ProfileBody contains the DeviceIdentity, the DeviceManager, the DeviceFunction and the ApplicationProcess elements shown in Fig. 2. The DeviceFunction element is extended with the moduleManagement for supporting the modular child device functionality. If a device implements a modular child node this profile template shall be used.

The ProfileBody_Device_Powerlink_Modular_Head element contains the attributes given in Tab. 5 and Tab. 7.

Attribute	Data type	Use	Description
specificationVersion	xsd:string	required	Describes the specification version of the device profile template. Fixed: 1.0.4

Tab. 7 Additional attribute of ProfileBody_Device_Powerlink_Modular_Child

7.4.4 DeviceIdentity

7.4.4.1 General

The DeviceIdentity element (see Fig. 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.

Tab. 8 specifies the attribute readOnly, which is attached to the vendorName, vendorID, vendorText, deviceFamily, productFamily, productName, productID, productText, orderNumber, version, specificationRevision and instanceName elements.

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

Tab. 8 Attribute of element vendorName

7.4.4.2 vendorName

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

7.4.4.3 vendorID

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

NOTE Consortia specific product families and vendor identifiers are linked.

7.4.4.4 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.

7.4.4.5 deviceFamily

The deviceFamily element states the family of the device.

EXAMPLE

Examples for device families are:

- Variable Speed Drive
- Circuit Breaker
- Pressure Sensor

7.4.4.6 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.

7.4.4.7 productName

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

7.4.4.8 productID

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

7.4.4.9 productText

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

7.4.4.10 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 product or a device family.

7.4.4.11 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 Tab. 9.

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. (default: true)

Tab. 9 Attributes of element version

7.4.4.12 buildDate

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

7.4.4.13 specificationRevision

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

7.4.4.14 instanceName

This element contains the instance name of the device. The default value of the attribute readOnly is set to false.

7.4.5 DeviceManager

7.4.5.1 General

The DeviceManager element defines the list of indicators provided by the device type, if any.

7.4.5.2 indicatorList / LEDList

7.4.5.2.1 General

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

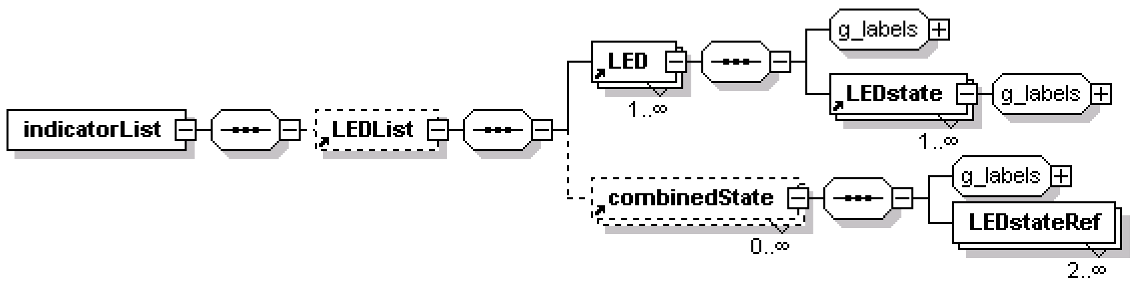

Fig. 11. indicatorList / LEDList

7.4.5.2.2 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 Tab. 10.

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

Tab. 10 Attributes of element LED

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 Tab. 11. Additionally a unique ID is allocated for the LED state.

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.

Tab. 11 Attributes of element LEDstate

7.4.5.2.3 combinedState

The combinedState element allows the indication of device states which are signalled 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 Tab. 12).

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

Tab. 12 Attribute of element LEDstateRef

7.4.5.3 moduleManagement (modular device only)

This element is only supported in the abstract profile types which include the modular device support ProfileBody_Device_Powerlink_Modular_Head and ProfileBody_Device_Powerlink_Modular_Child.

7.4.5.3.1 General

The moduleManagement element, if present, contains either a moduleInterfaceList element and an optional moduleInterface element or a single moduleInterface element as shown in Fig. 12 and Fig. 13.

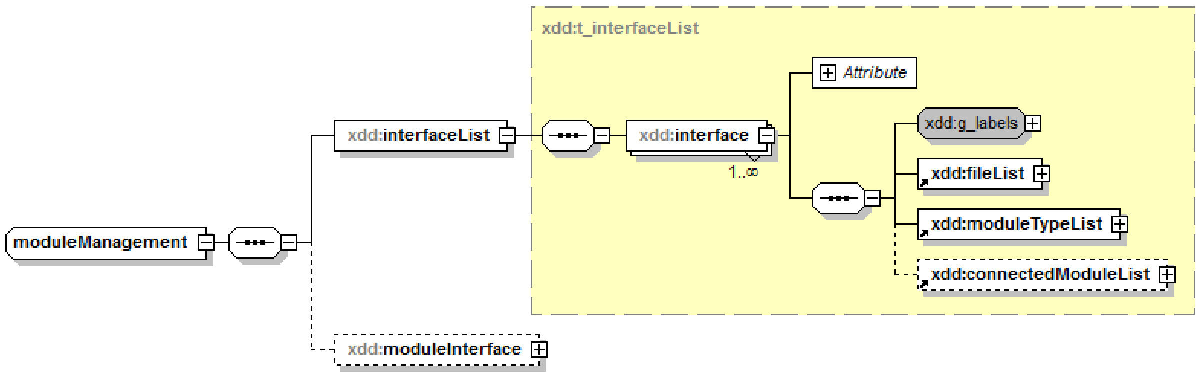

Fig. 12. moduleManagement element of a modular head device

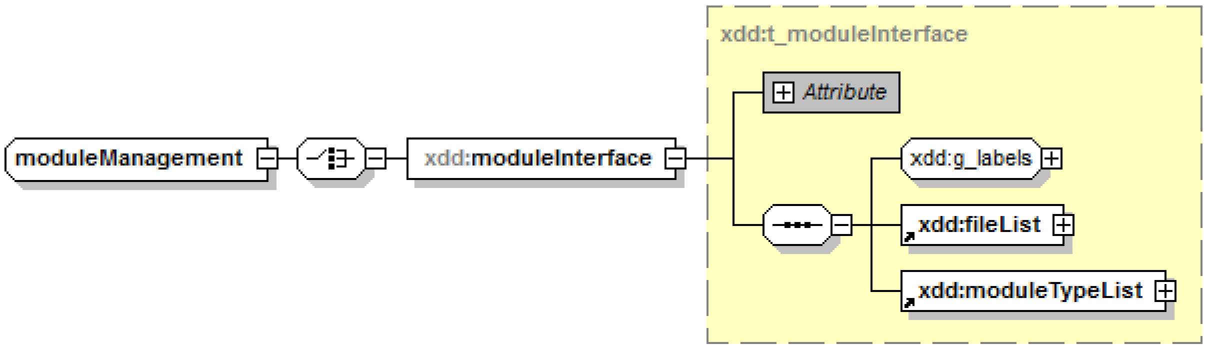

Fig. 13. moduleManagement element of a modular child device

The interfaceList element (see 7.4.5.3.2) shall be present if the device implements the root of a modular device. If the modular head device implements a module also the moduleInterface may be present.

The single moduleInterface element (see 7.4.5.3.3) shall be present if the device implements a single module and may be present if the device is a head node and a module simultaneously.

7.4.5.3.2 interfaceList

7.4.5.3.2.1 General

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

7.4.5.3.2.2 interface

7.4.5.3.2.2.1 General

The mandatory interface element (see Fig. 14) is used to indicate the properties of one interface provided by a modular device. These properties include a file path (see 7.4.5.3.2.2.2) to the XML device description files defining the modules to be connected to the modular device and may be the actual connected modules (see 7.4.5.3.2.2.4).

The interface element may contain the element group g_labels with one or more sub elements and contains the attributes defined in Tab. 13.

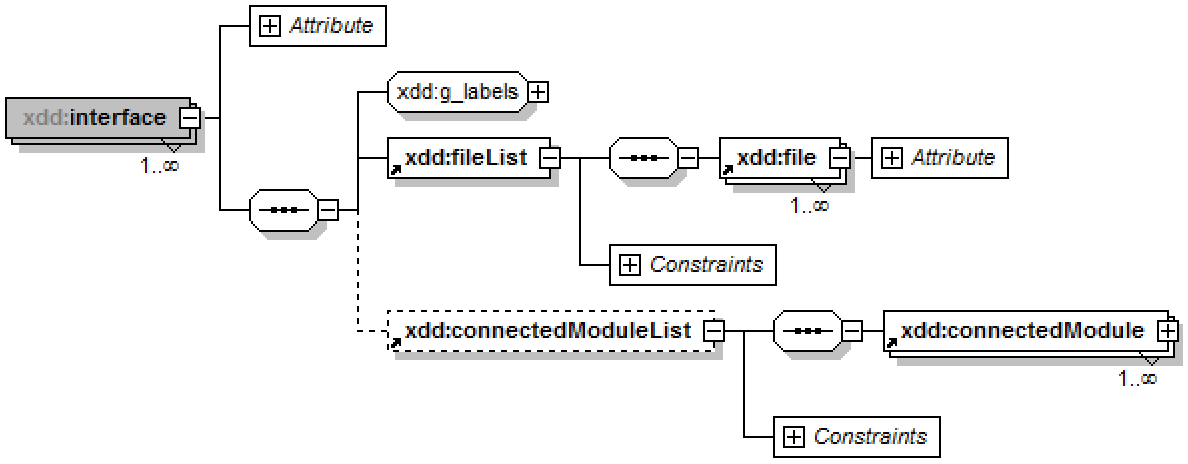

Fig. 14. interface element (modular head node)

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the interface.
type	xsd:NCName	required	Module type supported by the interface.
maxModules	xsd:positiveInteger	required	Maximum number of modules to be connected to the interface.
unusedSlots	xsd:boolean	required	Defines whether empty slots are allowed.
multipleModules	xsd:boolean	default	Defines if multiple modules of the same type are allowed. Default: true
moduleAddressing	xsd:NMTOKEN	required	Defines the addressing scheme for the node: • **manual** – The user shall provide an address for the module manually. • **position** – Modules shall receive an address which is equal with the position on the bus.
identList	xsd:hexBinary	optional	Index of the object (four hex digits) where identification data of child devices is stored. See App. 5 for details.
firmwareList	xsd:hexBinary	optional	Index of the object (four hex digits) where firmware data of child devices is stored. See App. 6 for details.

Tab. 13 Attributes of element interface

7.4.5.3.2.2.2 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 Tab. 14) to search for additional XML device description files containing definitions of the modules.

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

Tab. 14 Attributes of element file

7.4.5.3.2.2.3 moduleTypeList / moduleType

The moduleTypeList element contains a sequence of one to many moduleType elements. It is mandatory for a moduleInterface element and optional for an interface element.

A Type element contains a reference to the module type of a module allowed to be connected to the modular device and a unique id to be referenced from a connector element. The according module type is defined in the additional XML device description files referenced by the fileList element (see 7.4.5.3.2.2.2). The module type is defined by the type attribute (see Tab. 15).

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique id of the module type to be referenced by a connector attribute interfaceIDRef.
type	xsd:NCName	required	Reference to an allowed module type.

Tab. 15 Attributes of element moduleType

7.4.5.3.2.2.4 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 Tab. 16.

Attribute	Data type	Use	Description
childIDRef	xsd:string	required	Reference to the child id of a connected module.
position	xsd:positiveInteger	required	Defines the position of the module on the interface.
address	xsd:positiveInteger	optional	Defines the address of the module on the interface. Default: position

Tab. 16 Attributes of element connectedModule

7.4.5.3.3 moduleInterface

The moduleInterface element is used to indicate the properties of one module interface provided by a modular child device. These properties may include a file path (see 7.4.5.3.2.2.2) to the XML device description files defining the modules to be connected to the modular device and the module types (see 7.4.5.3.2.2.3) of the modules to be connected to the modular device,

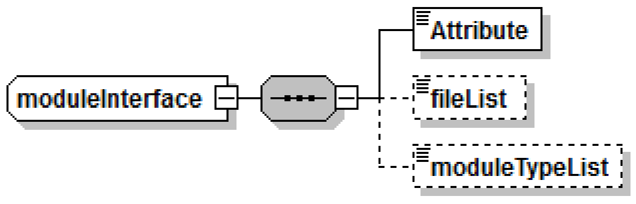

Fig. 15. moduleInterface element (modular child device)

The single moduleInterface element shall contain the attributes given in Tab. 17.

Attribute	Data type	Use	Description
childID	xsd:string	required	Unique ID of the module. (see NOTE)
type	xsd:NCName	required	Type of the module.
moduleAddressing	xsd:NMTOKEN	required	Defines if the module addressing on the bus: • **manual** – Module address shall be assigned manually by the user. Requires modular head interface attribute moduleAddressing to be manual. • **position** – Module address shall be assigned according to the position of the module. Modular head interface attribute moduleAddressing may only be manual or position. • **next** – Module address shall be assigned to the next address after its preceding module address. Requires modular head interface attribute moduleAddressing to be manual.
minPosition	xsd:positiveInteger	default	Minimum allowed position of the module on the interface bus. Default: 1.
maxPosition	xsd:positiveInteger	optional	Maximum allowed position of the module on the interface bus. Defaults to the maxModules attribute in the head interface.
minAddress	xsd:positiveInteger	default	Minimum allowed address of the module on the interface bus. Default: 1.
maxAddress	xsd:positiveInteger	optional	Maximum allowed address of the module on the interface bus. Defaults to the maxModules attribute in the head interface.
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.			

Tab. 17 Attributes of element moduleInterface

7.4.6 DeviceFunction

7.4.6.1 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 Fig. 5 are detailed in the following subclauses.

7.4.6.2 capabilities

7.4.6.2.1 General

The optional 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.

7.4.6.2.2 characteristicsList

7.4.6.2.2.1 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.

7.4.6.2.2.2 characteristic

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

7.4.6.2.2.3 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".

7.4.6.2.2.4 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.

7.4.6.2.2.5 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 Tab. 18.

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

Tab. 18 Attributes of element compliantWith

7.4.6.3 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.

Tab. 19 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.

Attribute	Data type	Use	Description
URI	xsd:anyURI	required	Link to the external resource
type	xsd:NMTOKEN	default	Defines the type of the given picture. • **frontPicture** – Is used to display the device in a configuration tool. Only one front picture shall be defined. • **icon** – Defines an icon for the device in a configuration tool. Only one icon shall be defined. • **additional** – Additional pictures linked to the device. • **none** – default.
number	xsd:unsignedInt	optional	Number of the picture

Tab. 19 Attributes of element picture

7.4.6.4 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 Tab. 20). 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 Tab. 3.

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

Tab. 20 Attributes of element dictionary

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

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

Tab. 21 Attribute of element file

7.4.6.5 connectorList

The optional connectorList element offers the possibility to describe the devices connectors. It contains one to many connector elements. The content of type g_labels of this element may contain remarks concerning the connector.

Attribute	Data type	Use	Description
id	xsd:string	required	Defines the id for the connector
posX	xsd:nonNegativeInteger	optional	Defines the X-axis number of pixel for the connector placement relative to the given device picture counted from the left upper corner.
posY	xsd:nonNegativeInteger	optional	Defines the Y-axis number of pixel for the connector placement relative to the given device picture counted from the left upper corner.
connectorType	xsd:string	default	Defines the fieldbus supported by the connector. Default is “POWERLINK”. Either connectorType or interfaceRef have to be present. The attribute is required for modular devices.
interfaceIDRef	xsd:string	optional	The connector may reference an interface or a module type given in the module management. This can be used to define the fieldbus supported by the connector. This attribute is required for modular child devices.
positioning	xsd:NMTOKEN	default	The positioning defines the docking point for devices or submodules. If positioning is remote the connector position is ignore. • **remote** – default. Devices are connected with a cable connection. • **localAbove** – Devices are connected directly above the device. • **localBelow** – Devices are connected directly below the device. • **localLeft** – Devices are connected directly on the left side of the device. • **localRight** – Devices are connected directly on the right side of the device.

Tab. 22 Attributes of element connector

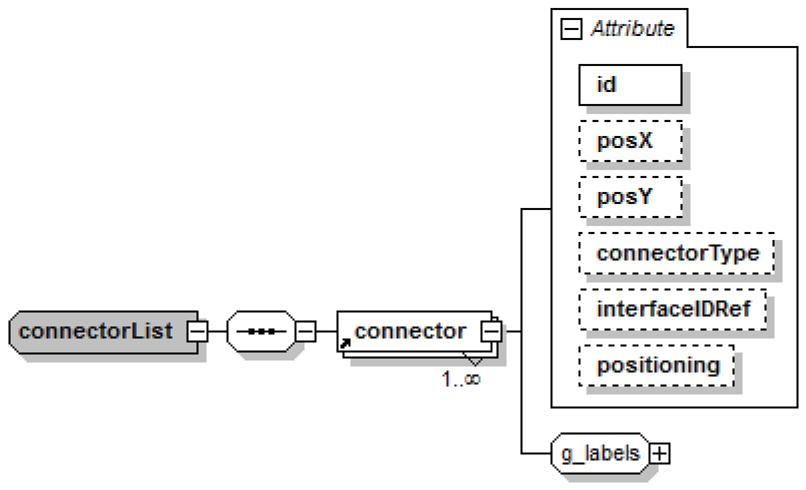

Fig. 16. connectorList

7.4.6.6 firmwareList

The optional firmwareList element offers the possibility to describe firmware files for the device revisions. It contains one to many firmware elements. The content of type g_labels of this element may contain remarks concerning the firmware.

Attribute	Data type	Use	Description
URI	xsd:anyURI	required	Link to the external firmware file.
deviceRevisionNumber	xsd:nonNegativeInteger	required	Defines the device revision for which the firmware is designed.
buildDate	xsd:dateTime	optional	Creation date of the firmware.

Tab. 23 Attributes of element firmware

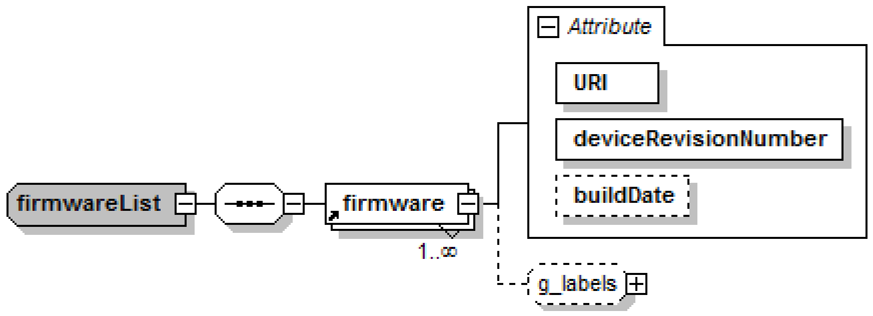

Fig. 17. firmwareList

7.4.6.7 classificationList

The optional classificationList element offers the possibility to classify the devices capabilities. It contains one to many classification elements.

Data type	Use	Description
xsd:NMTOKEN	required	Classification definition • **Controller** • **Industrial PC** • **Operator Interface** • **IO** • **Drive** • **Motor** • **Encoder** • **Vision System** • **Sensor** • **Temperature** • **Counter** • **Safe** • **Analog** • **Digital** • **Input** • **Output**

Tab. 24 Element classification

7.4.7 ApplicationProcess

7.4.7.1 General

The ApplicationProcess element represents the set of services and parameters, which constitute the behaviour, 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 Fig. 6 provide a generic approach for the description of arbitrary, flat or hierarchically structured functions of a device.

Functions are modelled 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 Fig. 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.

7.4.7.2 dataTypeList

7.4.7.2.1 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 Fig. 18 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.

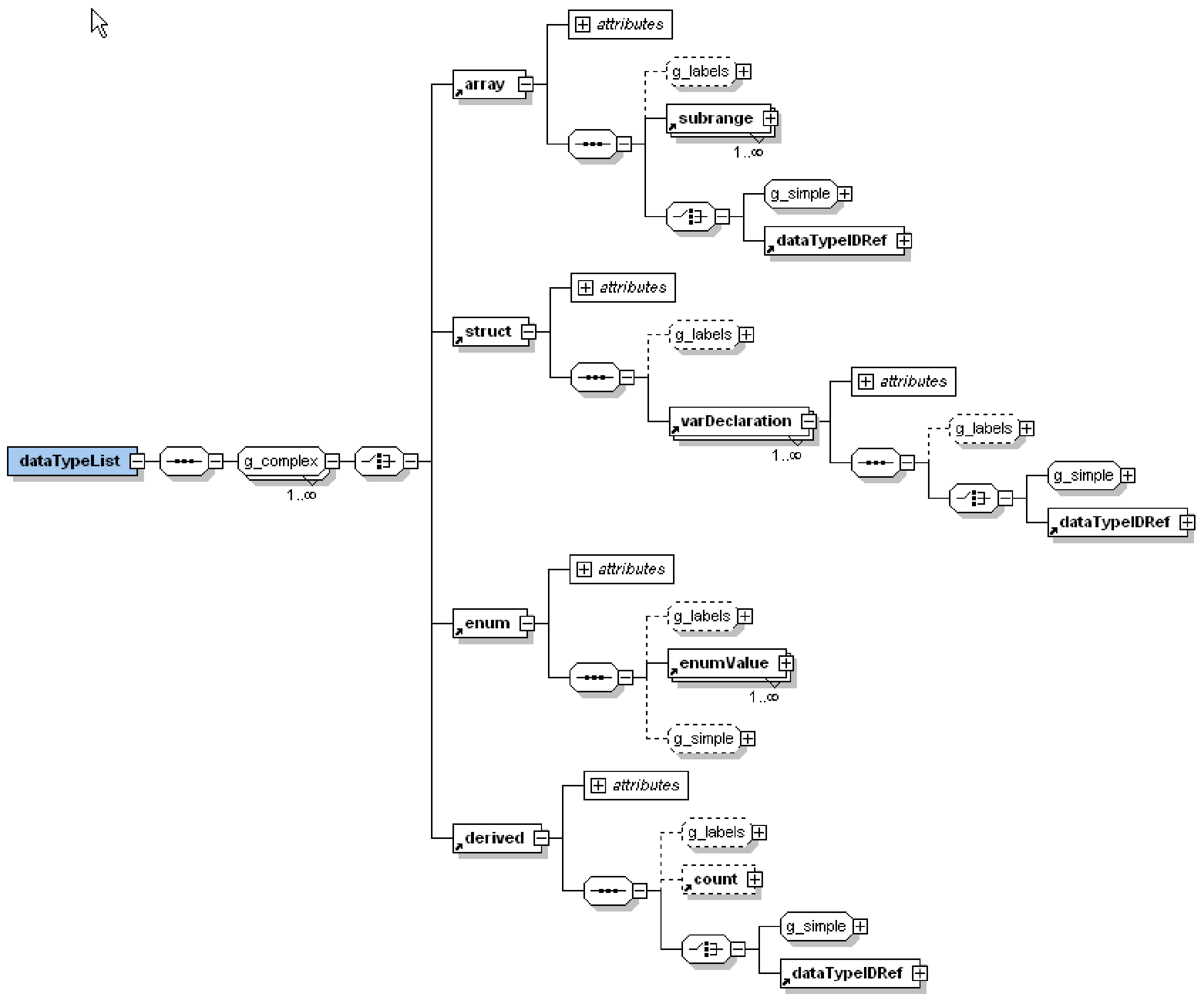

Fig. 18. dataTypeList

7.4.7.2.2 Common elements

7.4.7.2.2.1 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 IEC 61131-3; 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, or other elements).

7.4.7.2.3 array

7.4.7.2.3.1 General

The array 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 parameter specification.

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 Tab. 25.

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

Tab. 25 Attributes of element array

7.4.7.2.3.2 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 Tab. 26.

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

Tab. 26 Attributes of element subRange

7.4.7.2.4 struct

7.4.7.2.4.1 General

The struct 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.

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 Tab. 27.

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

Tab. 27 Attributes of element struct

7.4.7.2.4.2 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.

All further properties of the variable are contained in the attributes of the varDeclaration element, as given in Tab. 28.

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)
size	xsd:positiveInteger	default	Number of elements, if the interface variable or structure component is of type anonymous ARRAY, BITSTRING, STRING or WSTRING (see NOTE 2) Default: 1.
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.			

Tab. 28 Attributes of element varDeclaration

7.4.7.2.5 enum

7.4.7.2.5.1 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.

According to Fig. 18, 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 Tab. 29.

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:positiveInteger	default	Optional number of enumerated values of the enumerated data type. Default: 1.

Tab. 29 Attributes of element enum

7.4.7.2.5.2 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 Tab. 30.

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

Tab. 30 Attribute of element enumValue

7.4.7.2.6 derived

7.4.7.2.6.1 General

The derived element serves to derive a new data type from a given base type.

The derived element may contain the element group g_labels with one or more subelements and 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 Tab. 31.

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
description	xsd:string	optional	Description of the derived type

Tab. 31 Attributes of element derived

7.4.7.2.6.2 count

The count element defines the number of used units of the base type of the derived type. 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 count element shall contain the attributes given in Tab. 32.

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

Tab. 32 Attributes of element count

7.4.7.3 functionTypeList

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

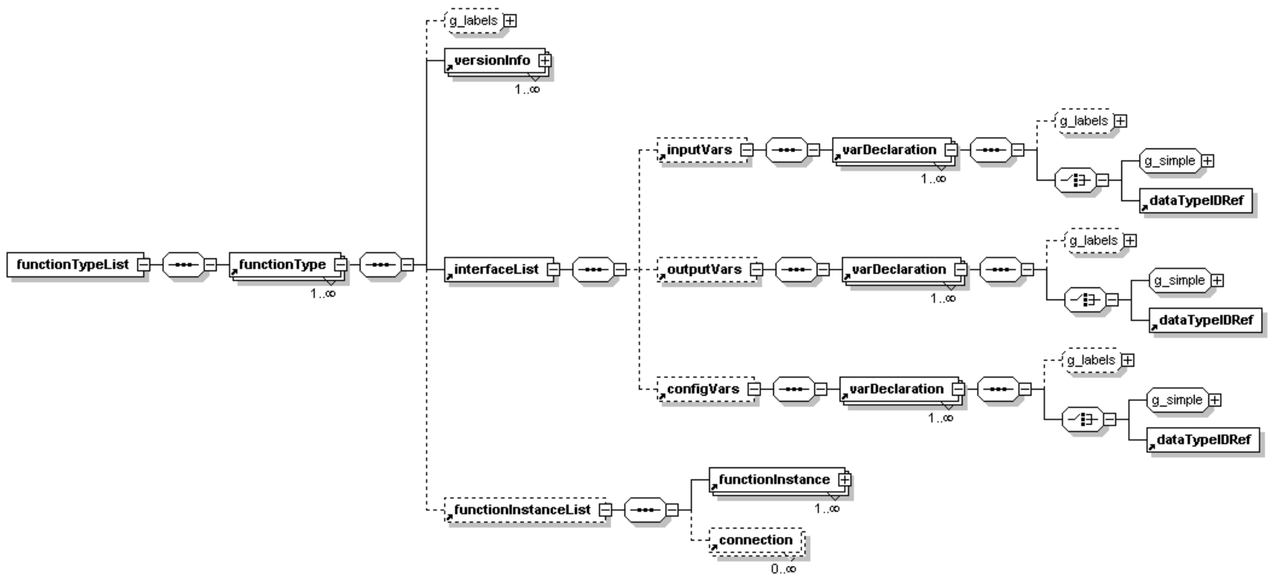

Fig. 19. functionTypeList

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.

7.4.7.4 functionType

7.4.7.4.1 General

The functionType element may contain the element group g_labels with one or more subelements and contains 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 Tab. 33.

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

Tab. 33 Attributes of element functionType

7.4.7.4.2 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 Tab. 34.

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

Tab. 34 Attributes of element versionInfo

7.4.7.4.3 interfaceList

7.4.7.4.3.1 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:

- 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.

7.4.7.4.3.2 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 type, the varDeclaration element describes a single interface variable of the function type.

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.

7.4.7.2.2.1 describes the group g_simple and 7.4.7.4.3.3 describes the dataTypeIDRef element.

All further properties of the variable are contained in the attributes of the varDeclaration element, as given in Tab. 35.

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
size	xsd:string	optional	Number of elements, if the interface variable or structure component is of type anonymous ARRAY, BITSTRING, STRING or WSTRING
initialValue	xsd:string	optional	Initial value of the interface variable or structure component

Tab. 35 Attributes of element varDeclaration

7.4.7.4.3.3 dataTypeIDRef

The dataTypeIDRef element serves to reference a complex data type inside the dataTypeList element (see 7.4.7.2), 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 Tab. 36.

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

Tab. 36 Attribute of element dataTypeIDRef

7.4.7.5 functionInstanceList

7.4.7.5.1 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. 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.

7.4.7.5.2 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 Tab. 37.

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.			

Tab. 37 Attributes of element functionInstance

7.4.7.5.3 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 Tab. 38.

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

Tab. 38 Attributes of element 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.

7.4.7.6 templateList

The various sub-elements of the optional templateList element are referenced from within parameter elements or allowedValues elements, 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 Fig. 20.

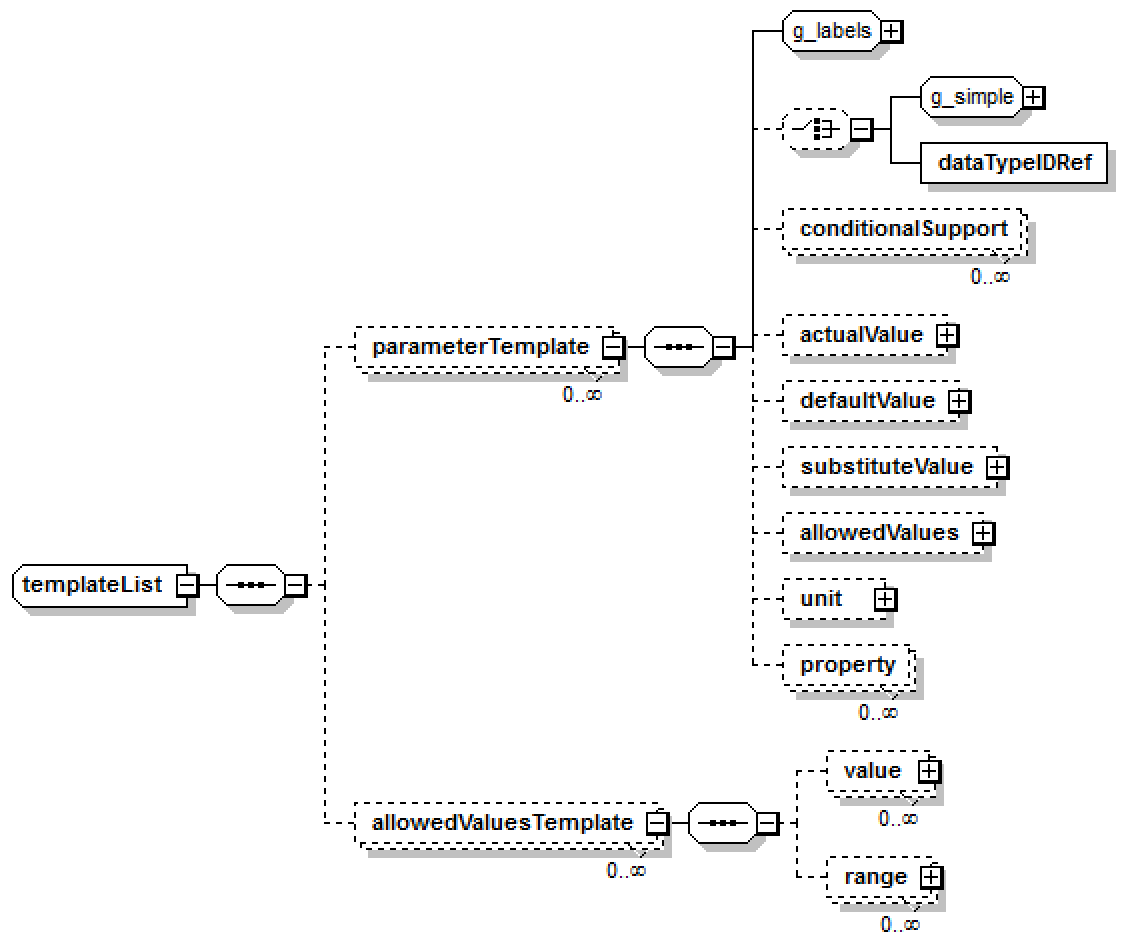

Fig. 20. templateList

The sub-elements of the parameterTemplate element are a subset of the sub-elements of the parameter element, as specified in 7.4.7.7.2, such that the descriptions of these sub-elements in the subclauses of 7.4.7.7.2 hold.

The parameterTemplate element shall contain the same attributes as the parameter element, as given in Tab. 40.

The sub-elements of the allowedValuesTemplate element are the same as the sub-elements of the allowedValues element, as specified in 7.4.7.7.2.7, such that the descriptions of these sub-elements in the subclauses of 7.4.7.7.2.7 also hold.

The single attribute of the allowedValuesTemplate element is given in Tab. 39.

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the template. Shall only be referenced by a Parameter element.

Tab. 39 Attribute of element allowedValuesTemplate

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.

7.4.7.7 parameterList

7.4.7.7.1 General

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

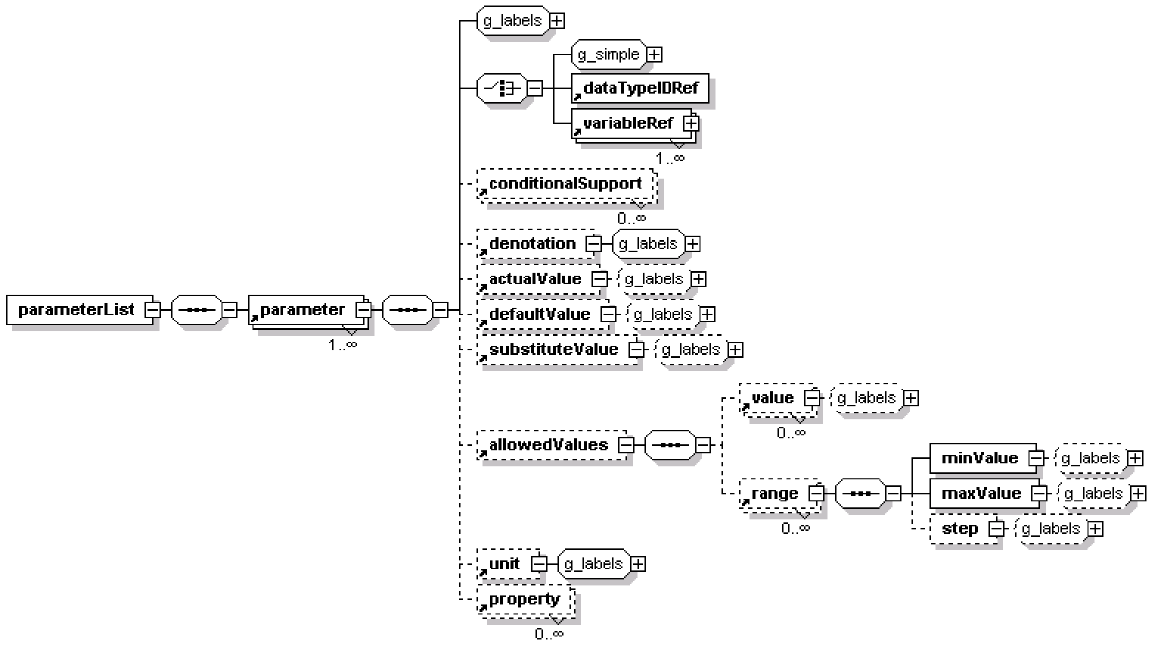

Fig. 21. parameterList

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.

7.4.7.7.2 parameter

7.4.7.7.2.1 General

The parameter element shall contain the attributes given in Tab. 40.

NOTE The same attributes are also valid for the parameterTemplate element, as specified in 7.4.7.6.

Attribute	Data type	Use	Description
uniqueID	xsd:ID	required	Unique ID of the parameter
access	xsd:NMTOKEN	optional	Defines which tool operations are valid for the parameter: • const - read access only; the value is not changing • read - read access only 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 behaviour 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

Tab. 40 Attributes of element parameter

7.4.7.7.2.2 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 Tab. 41.

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

Tab. 41 Attribute of element conditionalSupport

7.4.7.7.2.3 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.

7.4.7.7.2.4 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 Tab. 42.

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

Tab. 42 Attributes of element actualValue

7.4.7.7.2.5 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 Tab. 42.

7.4.7.7.2.6 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 Tab. 42.

7.4.7.7.2.7 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 Fig. 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 Tab. 42.

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 Tab. 42 are also valid for these sub-elements.

7.4.7.7.2.8 unit

The unit element defines the engineering unit of a parameter (e.g. time, temperature, pressure, flow, acceleration, current, energy), as specified in ISO 1000. 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 Tab. 43.

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 ISO 1000

Tab. 43 Attributes of element unit

7.4.7.7.2.9 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 Tab. 44.

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.			

Tab. 44 Attribute of element variableRef

7.4.7.7.2.10 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 Tab. 45.

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

Tab. 45 Attribute of element instanceIDRef

7.4.7.7.2.11 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 Tab. 46.

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

Tab. 46 Attribute of element variableIDRef

7.4.7.7.2.12 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 Tab. 47.

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

Tab. 47 Attributes of element memberRef

7.4.7.7.2.13 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 Tab. 48.

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

Tab. 48 Attributes of element property

7.4.7.8 parameterGroupList

7.4.7.8.1 General

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

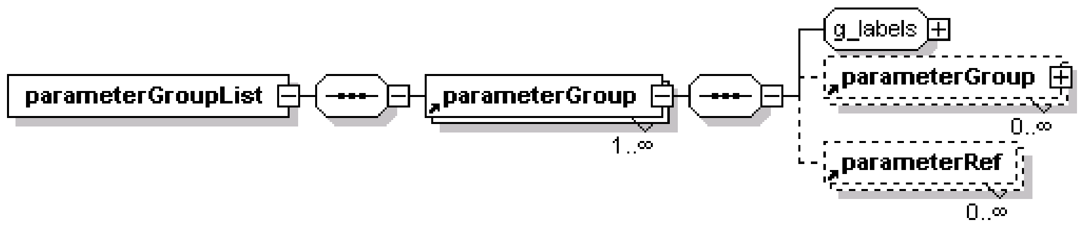

Fig. 22. parameterGroupList

7.4.7.8.2 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 Tab. 49.

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.
configParameter	xsd:boolean	default	An engineering tool shall consider the group as configuration parameters if the respective visible attribute is true. Default: false.
groupLevelVisible	xsd:boolean	default	An engineering tool shall not display a new hierarchy level for this group. Default: false.
conditionalUniqueIDRef	xsd:IDREF	optional	Referenced parameter for conditional processing of this group.
conditionalValue	xsd:string	optional	Value for comparison with referenced parameter. If the referenced parameter does not have the given value, the group shall be ignored by an engineering tool.
bitOffset	xsd:nonNegativeInteger	default	Only relevant if directly or indirectly referenced by an Object element from communication network profile. Value for the bit offset defined position in the parent element. Default: 0

Tab. 49 Attributes of element parameterGroup

7.4.7.8.3 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 Tab. 50.

Attribute	**Data type**	**Use**	**Description**
uniqueIDRef	xsd:IDREF	required	Unique ID of the referenced parameter
actualValue	xsd:string	optional	Actual value of parameterRef. Can be used to control further conditional groups.
visible	xsd:boolean	default	Defines the visibility of the parameter in an engineering tool. Default: false
locked	xsd:boolean	default	Disable user input in an engineering tool . Default: false
bitOffset	xsd:nonNegativeInteger	default	Only relevant if indirectly referenced by an Object element from communication network profile. Value for the bit offset defined position in parent parameterGroup element. If bitOffset is implemented, the parent parameterGroup also shall implement bitOffset. Default: 0

Tab. 50 Attribute of element parameterRef

7.5 Communication network profile template description

7.5.1 ProfileBody_CommunicationNetwork_Powerlink

This part of the device profile template defines the POWERLINK communication interface.

The ProfileBody_CommunicationNetwork_POWERLINK element contains the ApplicationLayers, the TransportLayers and the NetworkManagement sub-elements shown in Fig. 7. It has the attributes given in Tab. 51.

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
specificationVersion	xsd:string	default	Describes the specification version of the device profile template. Default: 1.0.4

Tab. 51 Attributes of element ProfileBody

7.5.2 ProfileBody_CommunicationNetwork_Powerlink _Modular_Head

This part of the device profile template defines the POWERLINK communication interface for a modular head node. If a device implements a modular head node this profile template shall be used.

The ProfileBody_CommunicationNetwork_Powerlink_Modular_Head element contains the ApplicationLayers, the TransportLayers and the NetworkManagement sub-elements shown in Fig. 7. It has the attributes given in Tab. 51 and Tab. 52. Additionally it includes a moduleManagement element in the ApplicationLayers element to support the modular head node device functionality.

Attribute	Data type	Use	Description
specificationVersion	xsd:string	required	Describes the specification version of the device profile template. Fixed: 1.0.4

Tab. 52 Additional attribute of ProfileBody_CommunicationNetwork_Powerlink_Modular_Head

7.5.3 ProfileBody_CommunicationNetwork_Powerlink _Modular_Child

This part of the device profile template defines the POWERLINK communication interface for a modular child node. If a device implements a modular child node this profile template shall be used.

The ProfileBody_CommunicationNetwork_Powerlink_Modular_Child element contains the ApplicationLayers, the TransportLayers and the NetworkManagement sub-elements shown in Fig. 7. It has the attributes given in Tab. 51 and Tab. 53. Additionally it includes a moduleManagement element in the ApplicationLayers element to support the modular child node device functionality.

Attribute	Data type	Use	Description
specificationVersion	xsd:string	required	Describes the specification version of the device profile template. Fixed: 1.0.4

Tab. 53 Additional attribute of ProfileBody_CommunicationNetwork_Powerlink_Modular_Child

7.5.4 ApplicationLayers

7.5.4.1 General

Fig. 23 shows the structure of the ApplicationLayers element.

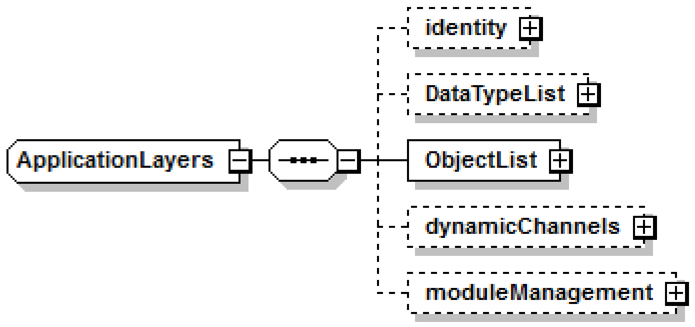

Fig. 23. POWERLINK ApplicationLayers element

The ApplicationLayers element has the attributes given in following Table.

Attribute	Data type	Use	Description
conformanceClass	xsd:string	optional	Conformance class of the device type (see NOTE)
communicationEntityType	xsd:NMTOKENS	default	Shall not be used. Provided for compatibility to CANopen. default: slave
NOTE Using that attribute it is possible to classify the device according to the supported services of the communication protocol.			

Tab. 54 Attributes of element ApplicationLayers

7.5.4.2 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, shown in Fig. 24. All sub–element descriptions given there also apply for the sub-elements of this identity element.

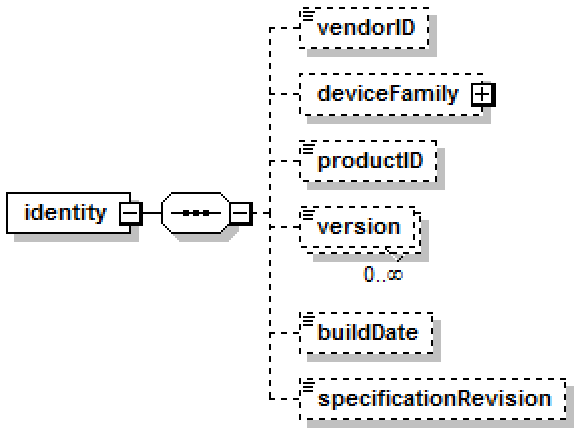

Fig. 24. identity element with sub-elements

7.5.4.3 DataTypeList

Data types in ObjectList are given in hexadecimal number (defined in the POWERLINK Specification). The element DataTypeList stores a hex to data type relation to ease tool implementation. Modular child devices may not implement the DataTypeList element. The following Figure shows the element DataTypeList and its sub elements (not all data types shown)

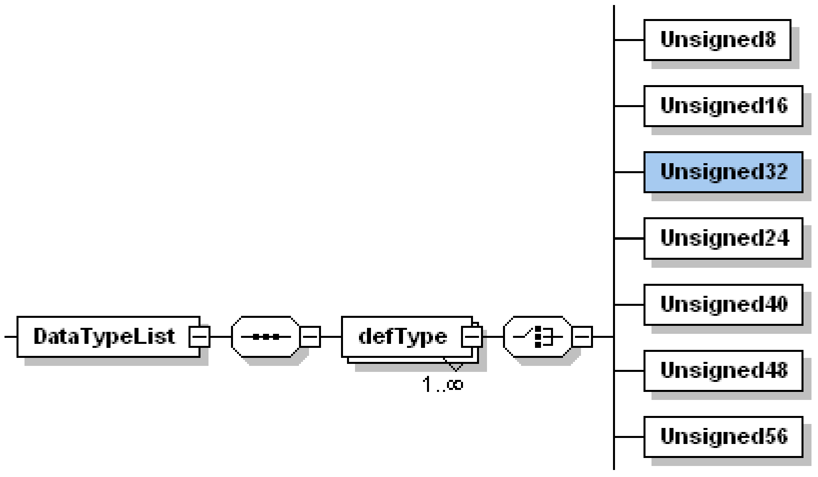

Fig. 25. DataTypeList element

For each data type DataTypeList contains a sub element defType. The defType element contains the attributes given in following Table.

Attribute	Data type	Use	Description
dataType	xsd:hexBinary	required	Gives the hexadecimal code for the specified data type (codes are defined in the POWERLINK specification). This code is then used in the dataType attributes of the elements Object and SubObject.

Tab. 55 Attributes of element defType

Therefore a device description file may contain the following fragment, with multiple defType sub-elements:

```
<DataTypeList>
        <defType dataType="hexadecimal number">
                <data type/>
        </defType>
</DataTypeList>
```

Thereby one defType sub-element for each hexadecimal number of Tab. 56is mandatory.

Hexadecimal representation	Data type
0001	Boolean
0002	Integer8
0003	Integer16
0004	Integer32
0005	Unsigned8
0006	Unsigned16
0007	Unsigned32
0008	Real32
0009	Visible_String
0010	Integer24
0011	Real64
0012	Integer40
0013	Integer48
0014	Integer56
0015	Integer64
000A	Octet_String
000B	Unicode_String
000C	Time_of_Day
000D	Time_Diff
000F	Domain
0016	Unsigned24
0018	Unsigned40
0019	Unsigned48
001A	Unsigned56
001B	Unsigned64
0401	MAC_ADDRESS
0402	IP_ADDRESS
0403	NETTIME

Tab. 56 Hexadecimal representation of the data types

7.5.4.4 ObjectList

Following figure shows the structure of the ObjectList element. The element contains one to many Object elements.

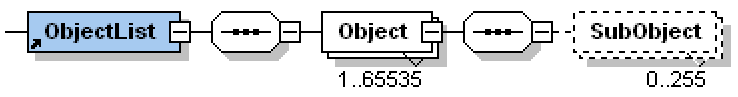

Fig. 26. ObjectList element

The ObjectList element contains the attributes given in following Table.

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

Tab. 57 Attributes of element ObjectList

If these attributes are used the objects given in ObjectList shall appear in order: all mandatory objects first followed by optional objects and ended with the manufacturer objects.

7.5.4.4.1 Object

The element contains zero to many SubObject elements. The Object element and the SubObject element map the functional part of the POWERLINK device profile to the POWERLINK communication network profile.

The Object element contains the attributes given in following Table.

Attribute	**Data type**	**Use**	**Description**
index	xsd:hexBinary	Required (optional for modular device)	Index of the object (four hex digits). A modular device may define an index offset with this attribute. This offset must not be higher than the sortStep of the referenced range in the head node.
name	xsd:string	required	Name of the object
objectType	xsd:unsignedByte	required	POWERLINK object type
dataType	xsd:hexBinary	optional	POWERLINK data type (four hex digits). See element DataTypeList.
lowLimit	xsd:string	optional	Low limit of the parameter value
highLimit	xsd:string	optional	High limit of the parameter value
accessType	xsd:NMTOKEN	optional	Access type of the object; valid values: • const – read only access; the value is not changing • ro – read only access • wo – write only access • rw – both read and write access
defaultValue	xsd:string	optional	Default value of the object
actualValue	xsd:string	optional	Actual value of the object. Tools store calculated values or user input in this attribute. Usually device description files (.xdd) do not contain this attribute, it is added by a tool after initial loading of a .xdd file. If the attribute actualValue is used the file has to be stored as a .xdc (XML device configuration file). However it might be necessary[1] to add the attribute already in the .xdd to define preset values of objects, which shall be part of the .xdc, even without user input.
denotation	xsd:string	optional	Application specific name of the object. Used by tools for application specific object naming (e.g. pressure, velocity). If the attribute denotation is used the file has to be stored as a .xdc (XML device configuration file).
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
objFlags	xsd:hexBinary	optional	Controls the behaviour 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

[1] *NOTE: By this it is possible to preselect a certain value, if the value is not equal to the default value. E.g. selection of a specific temperature sensor*

			• bit 2: 0 – change of value takes effect immediately 1 – change of value takes effect after reset • bit 3 to 31: reserved (0) NOTE 1 Bit 0 and bit 1 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 POWERLINK device. NOTE 2 Bit 2 gives information whether the change of an object value is immediately recognized by a device (and leads to a different behaviour) or is deferred to the next reset of the device.
uniqueIDRef	xsd:IDREF	optional	Unique ID of an appropriate element in the application process part referenced from this object; if the attribute is present and is referencing a parameter, the attributes dataType, lowLimit, highLimit, accessType, and defaultValue may be defined by the referenced element in the application process part.
subNumber	xsd:unsignedByte	optional	Number of sub-objects of the object
rangeSelector (modular device only)	xsd:string	required (optional for modular head device)	Reference the index range to be used according to the range element in the head node.

Tab. 58 Attributes of element Object

7.5.4.4.1.1 SubObject

The SubObject element has an empty content and contains the attributes given in following Table.

Attribute	**Data type**	**Use**	**Description**
subIndex	xsd:hexBinary	required	Subindex of the sub-object (two hex digits)
name	xsd:string	required	Name of the sub-object
objectType	xsd:unsignedByte	required	POWERLINK object type
dataType	xsd:hexBinary	optional	POWERLINK data type (four hex digits). See element DataTypeList.
lowLimit	xsd:string	optional	Low limit of the parameter value
highLimit	xsd:string	optional	High limit of the parameter value
accessType	xsd:NMTOKEN	optional	Access type of the sub-object; valid values: • const – read only access; the value is not changing • ro – read only access • wo – write only access • 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. Tools store calculated values or user input in this attribute. Usually device description files (.xdd) do not contain this attribute, it is added by a tool after initial loading of a .xdd file. If the attribute actualValue is used the file has to be stored as a .xdc (XML device configuration file). However it might be necessary[2] to add the attribute already in the .xdd to define preset values of objects, which shall be part of the .xdc, even without user input.
denotation	xsd:string	optional	Application specific name of the sub-object. Used by tools for application specific object naming (e.g. pressure, velocity). If the attribute denotation is used the file has to be stored as a .xdc (XML device configuration file).
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
objFlags	xsd:hexBinary	optional	Controls the behaviour 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 immediately 1 – change of value takes effect after reset

[2] *NOTE: By this it is possible to preselect a certain value, if the value is not equal to the default value. E.g. selection of a specific temperature sensor*

			• bit 3 to 31: reserved (0) NOTE 1 Bit 0 and bit 1 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 POWERLINK device. NOTE 2 Bit 2 gives information whether the change of an object value is immediately recognized by a device (and leads to a different behaviour) or is deferred to the next reset of the device.
uniqueIDRef	xsd:IDREF	optional	Unique ID of an appropriate element in the application process part referenced from this object; if the attribute is present and is referencing a parameter, the attributes dataType, lowLimit, highLimit, accessType, and defaultValue may be defined by the referenced element in the application process part

Tab. 59 Attributes of element SubObject

7.5.4.5 dynamicChannels

The optional dynamicChannels element contains one to many dynamicChannel elements.

7.5.4.5.1 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 POWERLINK network and the application program running on the device.

The dynamicChannel element contains the attributes given in following Table.

Attribute	**Data type**	**Use**	**Description**
dataType	xsd:hexBinary	required	POWERLINK 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 dictionary used by the process image.
endIndex	xsd:hexBinary	required	The index of the last 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.
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).

Tab. 60 Attributes of element dynamicChannel

7.5.4.6 moduleManagement (modular device only)

This element is only supported in the abstract profile type which includes the modular device support for modular head devices ProfileBody_CommunicationNetwork_Powerlink_Modular_Head.

7.5.4.6.1 General

The moduleManagement element, contains the interfaceList element. The structure of the moduleManagement element is given in Fig. 27.

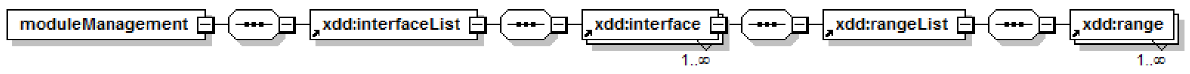

Fig. 27. moduleManagement element of a modular head node

7.5.4.6.2 interfaceList / interface

7.5.4.6.2.1 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 modular device. These properties include the range of the indices to be used. The interface element contains the attributes defined in Tab. 61.

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.

Tab. 61 Attributes of element interface

7.5.4.6.2.2 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 indices to be used when creating new objects based on the connected modules to a modular device. The range element contains the attributes defined in Tab. 62.

Attribute	Data type	Use	Description
name	xsd:string	required	Name of the range.
baseIndex	xsd:hexBinary	required	First index to be used within the range.
maxIndex	xsd:hexBinary	optional	Last index to be used within the range. If sortMode is subindex and no subindex overflow is allowed, this attribute may be omitted as it is equal to the baseIndex.
maxSubIndex	xsd:hexBinary	required	Last subindex 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 or subobject; valid values: • index • subindex
sortNumber	xsd: NMTOKEN	required	The rule how a new index or subindex is created; valid values: • **continuous** – a new index or subindex is created continuously for module objects. • **address** –A new index or subindex is created depending on the address of the module. A new index is calculated baseIndex plus address minus 1.
sortStep	xsd:positiveInteger	default	Step width for the next module. Default: 1.
PDOmapping	xsd:NMTOKEN	optional	PDO mapping of the objects and sub-objects; 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

Tab. 62 Attributes of element range

Examples for the usage of sortNumber and sortStep are given in Fig. 28.

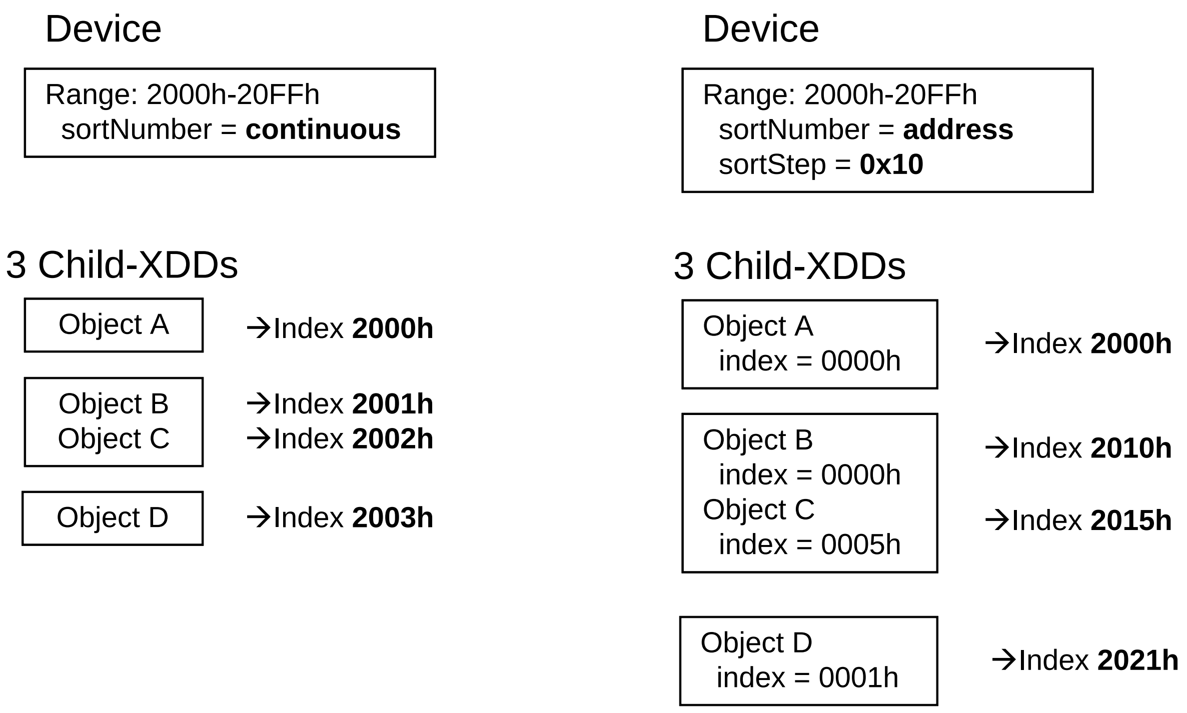

Fig. 28. Examples for index generation with sortNumber and sortStep

The resulting object index of the module configured for a given device consists of the baseIndex + sortStep (result of index generation as described in Fig. 28) + the index of the modules object.

7.5.5 TransportLayers

The TransportLayer element is empty, it is present for compatibility with the CANopen device description.

7.5.6 NetworkManagement

The following figure shows the structure of the NetworkManagement element. This element is optional for modular child devices but mandatory for any other POWERLINK device.

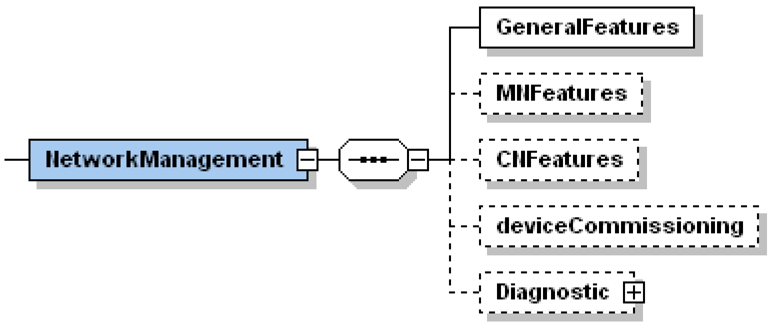

Fig. 29. NetworkManagement element

Device information not covered by mandatory objects are stored in subelements of NetworkManagement, these are:

- Features
  Are called device description entries in the POWERLINK Specification. See the POWERLINK Specification for a comprehensive list. The features are further grouped in general, MN and CN features.

- device commissioning
  Used by tools to store additional commissioning information (in device configuration files - .xdc only)
- Diagnostic
  Description of diagnostic data (errors) reported from the device

7.5.6.1 Features

Every feature is assigned to GeneralFeatures, MNFeatures or CNFeatures according to its availability in a MN, CN or both. The GeneralFeatures sub element is mandatory whereas MNFeatures and CNFeatures are optional. In a device description for a CN the element CNFeatures has to be present, vice versa for MNFeatures. It is also possible that MNFeatures and CNFeatures are present (for devices which support MN and CN mode of operation).

The elements GeneralFeatures, MNFeatures or CNFeatures have an empty content.

Every feature is described as an attribute of GeneralFeatures, MNFeatures or CNFeatures. The data type of a feature is derived from the data type defined in the POWERLINK Specification.

For a list and description of all POWERLINK Features please refer to the device description entries in the POWERLINK Specification.

7.5.6.1.1 Feature Names

Every feature defined in the POWERLINK specification (there called device description entry) has a name starting with "D_" and ending with a data type designator (e.g. "_U32"). The device description feature name strips these information and additionally all underlines ("_") are deleted.

Example:

"D_SDO_MaxParallelConnections_U32" gives the feature name "SDOMaxParallelConnections".

7.5.6.1.2 Feature Data Types

Every device description entry in the POWERLINK Specification has an assigned data type. These are converted to XML data types as follows:

POWERLINK	**XML**
BOOLEAN	xsd:boolean
INTEGER8	xsd:byte
INTEGER16	xsd:short
INTEGER32	xsd:int
UNSIGNED8	xsd:unsignedByte
UNSIGNED16	xsd:unsignedShort
UNSIGNED32	xsd:unsignedInt
REAL32	xsd:float
VISIBLE_STRING	xsd:string
OCTET_STRING	xsd:string
UNICODE_STRING	xsd:string
REAL64	xsd:double

Tab. 63 POWERLINK/XML data types

7.5.6.2 deviceCommissioning

The deviceCommissioning element is optional and has an empty content.

A .XDD contains no deviceCommissioning element, it is used by a tool to store configuration specific data in a .XDC file.

The deviceCommissioning element contains the attributes given in Tab. 64.

Attribute	Data type	Use	Description
networkName	xsd:string	required	The name of the network segment to which the device is connected.
nodeID	xsd:unsignedByte	required	The unique ID (node number) of the device.
nodeName	xsd:string	required	The name of the device.
nodeType	xsd:NMTOKENS	required	The allowed values are "CN", "MN", and possible for future extensions.
usedNetworkInterface	xsd:unsignedByte	default	For multi interface devices, index of used interface (see Object 1030h) default = 0

Tab. 64 Attributes of element deviceCommissioning

7.5.6.3 Diagnostic

The optional element Diagnostic contains the optional elements ErrorList and StaticErrorBitField. No attributes are defined.

The element Diagnostic is used to provide tools with presentational detail for the information transported in the StatusResponse frame (see the POWERLINK specification for details).

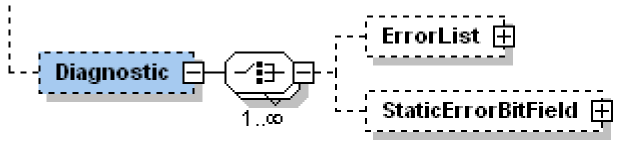

Fig. 30. Element Diagnostic

7.5.6.3.1 ErrorList

The element ErrorList is optional. It is further decomposed as shown in following figure.

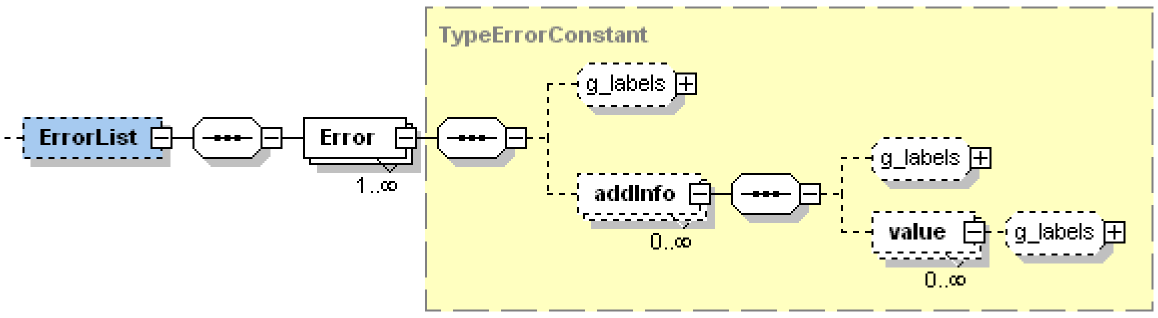

Fig. 31. Element ErrorList

The element ErrorList contains no attributes. One to many Error elements may be defined inside the ErrorList element.

7.5.6.3.1.1 Error

An Error element defines an error which may be signalled by the device. A device description file specifies only the Errors which are device or manufacturer specific. Errors which are already defined in the POWERLINK Specification are not presented in the device description file.

The element Error has the following attributes:

Attribute	Data type	Use	Description
name	xsd:string	required	A short name for the error.
value	xsd:string	required	The error code as signalled from the device.

Tab. 65 Attributes of element Error

The element Error may contain an optional g_labels element for multilingual descriptions of the error. It may also contain any number of optional addInfo elements.

The element addInfo is used to describe the content of the 64 Bits of additional information which may be sent from device in addition to the error code. If there is no addInfo element no additional information is provided for this error, therefore a tool shall not display the addition information (which is sent but has unspecified content).

The additional information may be organized in bit groups, a group containing 1 to 32 consecutive bits. Each of these groups is described by an addInfo element.

The element addInfo has the following attributes:

Attribute	**Data type**	**Use**	**Description**
bitOffset	xsd:unsignedByte	required	This is the bit starting an additional information bit group. The 64 Bits of additional information are numbered from 0 (LSB) to 31 (MSB). Refer to the POWERLINK specification for further information.
len	xsd:unsignedByte	required	The number of consecutive bits in the additional information bit group.
name	xsd:string	required	The short name for the additional information bit group.

Tab. 66 Attributes of element addInfo

The element addInfo may contain an optional g_labels element for multilingual descriptions of the additional information bit group. It may also contain any number of optional value elements.

The element value is provided to enumerate certain values of an additional information bit group. Tools may use this information to show a text instead of plain numbers. Values which have no text assigned to, are treated as unsigned numbers.

The element value has the following attributes:

Attribute	**Data type**	**Use**	**Description**
value	xsd:string	required	The value (unsigned number).
name	xsd:string	required	The short name assigned to value.

Tab. 67 Attributes of element value

The element addInfo may contain an optional g_labels element for assignment of multilingual texts to numbers.

7.5.6.3.2 StaticErrorBitField

This element is used to describe the 64 static error bits inside the StatusResponse frame (see the POWERLINK V2 specification for details). Bits which are already defined in the POWERLINK Specification are not presented in the device description file.

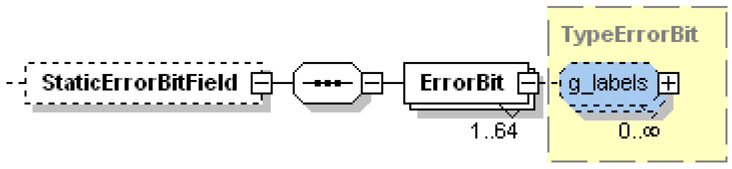

Fig. 32. Element StaticErrorBitField

The element StaticErrorBitField contains 1 to 64 ErrorBit elements.

The ErrorBit element describes one bit.

The element ErrorBit has the following attributes:

Attribute	**Data type**	**Use**	**Description**
name	xsd:string	required	The short name.
offset	xsd:nonNegativeInteger <=63	required	Bit number (starting at 0) in the static error bit field.

Tab. 68 Attributes of element ErrorBit

The element ErrorBit may contain an optional g_labels element for assignment of multilingual texts to error bits.

8 File naming convention

The file naming convention defined in this chapter shall apply to the following 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 POWERLINK framework, POWERLINK device profile or POWERLINK application profile. The extension shall be “XPD”.

The device description file is an XML file describing the device type of a POWERLINK device. The name of the file shall contain the vendor-ID of the POWERLINK device in the form of 8 hexadecimal digits. The extension shall be “XDD”.
The information in the XDD file is used as a blueprint for instantiation of devices in an actual network configuration. An XDD file contains the device default value but no device commissioning values and usually no actual values.

The device configuration file is an XML file describing a configured POWERLINK device. The extension shall be “XDC”.
All information from the XDD file plus actual values and/or device commissioning values can be stored in an XDC file. i.e. An XDC file stores the information for a specific instantiation of a device in a specific network environment.

See App. 3 for details on file naming syntax.

9 Tool behaviour

All the information needed to configure a POWERLINK network is contained in the device description files. Especially 2 areas inside the file are important.

1. The ObjectList contains all objects the device supports. This includes the mandatory objects as defined in the POWERLINK specification. The objects default value (as stored in the attribute defaultValue) has to be considered for network and device configuration purposes. It is not necessary to have an online connection to the device for configuration, the whole configuration process can be done offline using the default values.
2. Device features (see elements GeneralFeatures, MNFeatures, CNFeatures) represent information which is partly not contained in the device's object dictionary.

App. 1 Value syntax (normative)

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 IEC 61131-3.
NOTE The definition of hexadecimal values is different to the definition of IEC 61131-3. The definition is according to ANSI C.

boolean_value	::= 'true' | 'false'
unsigned	:= digit { digit }
integer_value	::= ['+' | '–'] unsigned
hexadecimal_value	::= '0' 'x' { hex_digit }
exponent	::= { 'E' | 'e' } integer_value
real_value	::= integer_value '.' unsigned [exponent]
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'

App. 2 XML file content convention (normative)

The XML file content shall be encoded in UTF8.

App. 3 XML file naming syntax (normative)

The syntax defined in this annex applies to the following XML files

- the profile definition file (XPD),
- the device description file (XDD), and
- the device configuration file (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 POWERLINK 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'
```

App. 4 Text resource

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.

Following figure shows the structure of the textResource element.

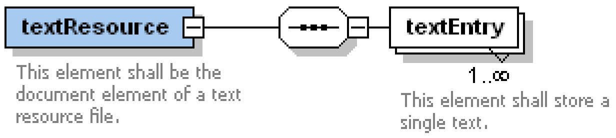
textResource
This element shall be the document element of a text resource file.
textEntry
1..∞
This element shall store a single text.

Fig. 33. Element textResource

The element textResource has the following attributes:

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

Tab. 69 Attributes of element textResource

The element textEntry contains the text data as content.
The element textEntry has the following attributes:

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

Tab. 70 Attributes of element textEntry

App. 5 Child identification (normative)

Object *identList*[3]: NMT_ChildIdentData_ADOM

NMT_ChildIdentData_ADOM holds identification data for child devices. It allows the access to identification data of up to 254 child devices. If the interface supports more than 254 child devices the list will continue with sub-index 1 of the index following *identList*.

Index	*identList*	Object Type	ARRAY
Name	NMT_ChildIdentData_ADOM		
Data Type	DOMAIN	Category	O

- **Sub-Index 00$_h$: NumberOfEntries**

Sub-Index	00$_h$		
Name	NumberOfEntries		
Value Range	01$_h$ .. FE$_h$	Access	ro
Default Value	01$_h$	PDO Mapping	No

- **Sub-Index 01$_h$ .. FE$_h$: ChildIdent**

Sub-Index	01$_h$ .. FE$_h$		
Name	ChildIdent		
--	--	Category	O
Value Range	DOMAIN	Access	ro
Default Value	-	PDO Mapping	No

ChildIdent sub-indices provide access to the identification data of child devices.

Each sub-index in the array corresponds to the child device with the module number equal to the sub-index.

- **Value Interpretation**

VendorId _U32	ProductCode _U32	RevisionNo _U32	SerialNo _U32	ApplSwDate _U32	ApplSwTime _U32	…

Tab. 71 ChildIdent value interpretation

ChildIdent is composed of at least 24 octets. The vendor may define more (vendor specific) data after this. However the size of the domain shall not exceed 128 bytes.

NOTE: This is to avoid problems with clients which cannot handle very large domains.

VendorId_U32, ProductCode_U32_RevisionNo_U32 and SerialNo_U32 are equivalent to the objects 1018h (NMT_IdentityObject_REC). ApplSwDate_32 and ApplSwTime_U32 are equivalent to 1F52h (PDL_LocVerApplSw_REC). For definitions and details see EPSG DS 301 [7].

[3] *identList* represents the value given by the attribute identList.

App. 6 Child firmware update (normative)

Object *firmwareList*[4]: PDL_DownloadChildProgData_ADOM

PDL_DownloadChildProgData_ADOM holds downloaded programs for child devices. It allows the access to up to 254 programs. If the interface supports more than 254 child devices the list will continue with sub-index 1 of the index following *firmwareList*.

Index	*firmwareList*	Object Type	ARRAY
Name	PDL_DownloadChildProgData_ADOM		
Data Type	DOMAIN	Category	O

- **Sub-Index 00_h: NumberOfEntries**

Sub-Index	00_h		
Name	NumberOfEntries		
Value Range	01_h .. FE_h	Access	ro
Default Value	01_h	PDO Mapping	No

- **Sub-Index 01_h .. FE_h: ChildProgram**

Sub-Index	01_h .. FE_h		
Name	ChildProgram		
--	--	Category	O
Value Range	DOMAIN	Access	cond
Default Value	-	PDO Mapping	No

ChildProgram sub-indices provide access to the firmware of child devices.

If the device supports reading of the program the access shall be rw otherwise wo.

Each sub-index in the array corresponds to the child device with the module number equal to the sub-index.

[4] *firmwareList* represents the value given by the attribute firmwareList.

App. 7 ParameterGroup and parameterRef bitOffset attribute usage

The following XDD example code implements the bitOffset attribute of the parameterGroup element and parameterRef element to configure the actual value of an Object element. It shows a hierarchy of parameterGroup and parameterRef elements used to set a number of specific bits in a 8 byte sized object entry.

The dataType and accessType of the referenced parameterGroup with uniqueID “UID_PG1” is also valid for all nested parameterRef and parameterGroup elements.

```
…
<parameter uniqueID="UID_Index21xx_Sub64_RegNumber" access="readWrite">
        <label lang="en">register Number</label>
        <label lang="de">Registernummer</label>
        <UINT/>
        <defaultValue value="1"/>
</parameter>
…
<parameter uniqueID="UID_Index21xx_MOD_CfgEntry_Size1"
templateIDRef="UID_PT_Index21xx_MOD_CfgEntry_Size">
        <label lang="en">size of MOD_CfgEntry_U64 data</label>
        <label lang="de">Größe der MOD_CfgEntry_U64-Daten</label>
        <USINT/>
        <defaultValue value="1"/>
</parameter>
….
<parameter uniqueID="UID_Index21xx_MOD_CfgEntry_Type2"
templateIDRef="UID_PT_Index21xx_MOD_CfgEntry_Type">
        <label lang="en">type of MOD_CfgEntry_U64</label>
        <label lang="de">Typ des MOD_CfgEntry_U64</label>
        <dataTypeIDRef uniqueIDRef="UID_DT_Nibble"/>
        <defaultValue value="2"/>
</parameter>
…
<dataTypeList>
        <struct name="Nibble" uniqueID="UID_DT_Nibble">
                <varDeclaration name="VarNibble" uniqueID="UID_VarNibble" size="4">
                        <label lang="de">4 Bit</label>
                        <label lang="en">4 bit</label>
                        <BITSTRING/>
                </varDeclaration>
        </struct>
</dataTypeList>
…
<parameterGroup uniqueID="UID_PG1">
        <label lang="en">Group desc</label>
        <parameterRef uniqueIDRef="UID_Index21xx_Sub64_RegNumber" bitOffset="0"/>
        <parameterGroup uniqueID="UID_internal_group" bitOffset="16">
                <label lang="en">Group desc</label>
                <parameterRef uniqueIDRef="UID_Index21xx_MOD_CfgEntry_Size1"/>
        </parameterGroup>
        <parameterRef uniqueIDRef="UID_Index21xx_MOD_CfgEntry_Type2" bitOffset="24"/>
</parameterGroup>…
…
<ObjectList>
        <Object index="2100" name=”Object1" objectType="9">
                <SubObject subIndex="01" name="SubObj1" objectType="7" dataType="0001B"
                accessType="rw" uniqueIDRef="UID_PG1"/>
        </Object>
</ObjectList>
…
```

Fig. 34. Example XDD fragment

According to this sample XDD fragment, the actual 8 byte value of the object with index 0x2100, subindex 0x01 is set to “0x0000000002010001”.