IO-Link

IODD

IO Device Description

Specification

related to

IO-Link Communication Specification V1.1

and

IODD Schemas V1.1

Version 1.1
August 2011

Order No: 10.012

IO-Link

File name: IO-Device-Desc-Spec_10012_V11_20110809.doc

Prepared, approved and released by the IO-Link Consortium

Any comments, proposals, requests on this document are appreciated. Please use www.io-link-projects.com for your entries and provide name and email address.
Login: *IO-Link-DD*
Password: *Report*

Important notes:

NOTE 1 The IO-Link Consortium Rules shall be considered prior to the development and marketing of IO-Link products. The document can be downloaded from the www.io-link.com portal.

NOTE 2 Any IO-Link device shall provide an associated IODD file.

NOTE 3 Any device which uses the IODD as device description shall provide easy access to the file and potential updates. It is the responsibility of the device manufacturer to test the IODD file with the help of the IODD-Checker tool available per download from www.io-link.com.

Conventions:

In this specification the following key words (in **bold** text) will be used:
may: indicates flexibility of choice with no implied preference.
should: indicates flexibility of choice with a strongly preferred implementation.
shall: indicates a mandatory requirement. Designers **shall** implement such mandatory requirements to ensure interoperability and to claim conformity with this specification.

Publisher:
IO-Link Consortium
Haid-und-Neu-Str. 7
76131 Karlsruhe
Germany
Phone: +49 721 / 96 58 590
Fax: +49 721 / 96 58 589
E-mail: info@io-link.com
Web site: www.io-link.com

Content

Figures

Tables

Revision Log

Version	Originator	Date	Change Note / History / Reason
1.1	Team Technology	09-August-2011	Final version.

1 Introduction

An IODD (IO Device Description) is a set of files that formally describes a device e.g. IO-Link Device.

The IODD is created by the device vendor and shall be sufficient for IO-Link Tools to identify, communicate, parameterize and diagnose the device.

The set of files consists of the main IODD file, optional language files and optional picture files.

An IODD is mandatory for each IO-Link Device. This specification defines the IODD for IO-Link Devices that conform to the IO-Link Communication Specification Version 1.1.

2 Related documents and references

2.1 References

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

IO-Link Communication Specification Version 1.1, November 2010, Order No: 10.002

IO-Link Communication Specification Version 1.1.1, to be released soon.

IO-Link Profile Smart Sensor Specification Version 1.0, to be released soon

IO-Link Test Specification Version 1.1, May 2011, Order No: 10.032

ANSI/IEEE Std 754-1985, *IEEE Standard for Binary Floating-Point Arithmetic*

IETF RFC 2083, *PNG (Portable Network Graphics) Specification Version 1.0*, available at http://tools.ietf.org/html/rfc2083

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

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

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

ISO 15745-1 Amd 1:2007, *Industrial automation systems and integration – Open systems application integration framework – Part 1: Generic reference description, Amendment 1*

IEC 60757:1983, *Code for designation of colours*

The Unicode Standard, V6.0.0, available at http://www.unicode.org/

ITU-T recommendation V.42 (03/2002), *Error-correcting procedures for DCEs using asynchronous-to-synchronous conversion*, available at http://www.itu.int/rec/T-REC-V.42-200203-I/en

REC-xml-20081126, *Extensible Markup Language (XML) 1.0 (Fifth Edition) – W3C Recommendation 26 November 2008,* available at http://www.w3.org/TR/xml/

REC-xmlschema-1-20041028, *XML Schema Part 1: Structures Second Edition – W3C Recommendation 28 October 2004,* available at http://www.w3.org/TR/xmlschema-1/

REC-xmlschema-2-20041028, *XML Schema Part 2: Datatypes Second Edition – W3C Recommendation 28 October 2004,* available at http://www.w3.org/TR/xmlschema-2/

2.2 Related documents

ANSI INCITS 4-1986 (R2007), *Information Systems – Coded Character Sets – 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII)* (predecessor of ISO/IEC 646)

IEC 60559:1989, *Binary floating-point arithmetic for microprocessor systems*

IETF RFC 3629, *UTF-8, a transformation format of ISO 10646*, available at http://tools.ietf.org/html/rfc3629

IETF RFC 5905, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, available at http://tools.ietf.org/html/rfc5905

ISO/IEC 3309:1993, *Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures – Frame structure* (withdrawn)

ISO 8601:2004, *Data elements and interchange formats – Information interchange – Representation of dates and times*

ISO/IEC 10646:2003/Amd 7:2010, *Information technology – Universal Multiple-Octet Coded Character Set (UCS)*

ISO/IEC 15948:2004, *Information technology – Computer graphics and image processing – Portable Network Graphics (PNG): Functional specification*

REC-xslt-19991116, *XSL Transformations (XSLT), Version 1.0, W3C Recommendation 16 November 1999*, available at http://www.w3.org/TR/xslt

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this document, the terms and definitions given in ISO 15745-1:2003 apply.

IO-Link Tool
Device engineering tool for the IO-Link Master and the connected IO-Link Devices. Used for parameterization and diagnosis of IO-Link Devices on the basis of the IODD.

3.2 Abbreviated terms

ANSI	American National Standards Institute (http://www.ansi.org/)
ASCII	American Standard Code for Information Interchange (see ANSI INCITS 4-1986 (R2007) and the US variant of ISO/IEC 646:1991)
BIPM	Bureau International des Poids et Mesures (http://www.bipm.org/)
C/Q	Connection for communication (C) or switching (Q) signal (SIO)
CRC	Cyclic Redundancy Check
DI	Digital Input

DO	Digital Output
I/Q	NC, DI or DO
IEC	International Electrotechnical Commision (http://www.iec.ch/)
IEEE	Institute of Electrical and Electronics Engineers (http://www.ieee.org/)
IETF	Internet Engineering Task Force (http://www.ietf.org/)
IO or I/O	Input / Output
IODD	IO Device Description
ISDU	Indexed Service Data Unit
ISO	International Standardization Organisation (http://www.iso.org/)
ITU	International Telecommunication Union (http://www.itu.int/)
LF	Line Feed
MSXML	Microsoft XML Core Services (see http://msdn.microsoft.com/en-us/library/ms763742%28VS.85%29.aspx)
NC	Not Connected
PLC	Programmable Logic Controller
PNG	Portable Network Graphics (see RFC 2083 and ISO/IEC 15948:2004)
RFC	Request for Comments
SIO	Standard Input Output (digital switching mode)
UCS	Universal Multiple-Octet Coded Character Set (see *The Unicode Standard* or ISO/IEC 10646:2003/Amd 7:2010)
UL	Underwriters Laboratories (http://www.ul.com/)
UTC	Coordinated Universal Time (Temps Universel Coordonné) (coordinated by the BIPM) (corresponds to GMT = Greenwich Mean Time)
UTF	UCS Transformation Format (see *The Unicode Standard* or ISO/IEC 10646:2003/Amd 7:2010)
W3C	World Wide Web Consortium (http://www.w3.org/)
XML	Extensible Markup Language (see REC-xml-20081126)
XSD	XML Schema Definition (see REC-xmlschema-1-20041028 and REC-xmlschema-2-20041028)
XSL	Extensible Stylesheet Language
XSLT	XSL Transform (see REC-xslt-19991116)

4 Basic structure

The following figure shows the basic structure of the main IODD file. It follows the ISO 15745-1:2003 standard regarding the device profile and communication network profile. It consists of the elements DocumentInfo, ProfileHeader, ProfileBody, CommNetworkProfile, ExternalTextCollection and the Stamp.

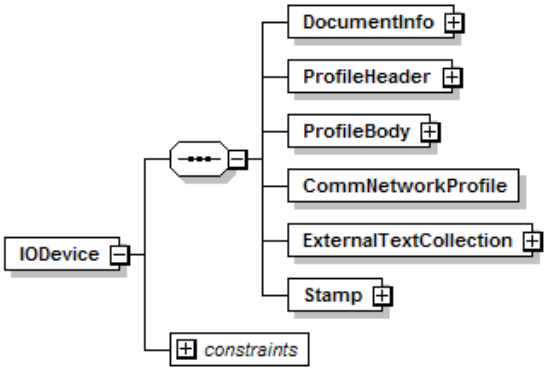


Figure 1 – Structure of main IODD file following ISO 15745-1

5 Files

Conceptionally, the IO Device Description consists of the set of files created by the device vendor, and the set of standard definition files which are part of this specification. IO-Link Tools combine information from both sets of files to get the complete device description.

All IODD XML files shall use “UTF-8” for the encoding. They shall use the name space http://www.w3.org/2001/XMLSchema-instance with the prefix “xsi” and the name space http://www.io-link.com/IODD/2010/10 with the prefix “iodd”. A schemaLocation for the name space http://www.io-link.com/IODD/2010/10 to the required schema shall be given. For the main IODD file, this is IODD1.1.xsd, and for the language files this is IODD-Primitives1.1.xsd. The schema file name shall be given without any path prefix. No other namespaces shall be described.

All XMLs generated by the vendor shall be checked by the IODD Checker software before delivery. This Checker is a tool available from the IO-Link web site (http://www.io-link.com/). It checks the content of the device description and if no errors were found writes a checksum over the file contents into the element Stamp at the end of the XML-file.

IO-Link Tools shall compare the checksum in the Stamp with the checksum calculated from the file contents. It is recommended to reject the IODD if there is a mismatch. Tools may then omit schema validation and additional checks.

IO-Link Tools shall use the file name of the IODD only to discover the language files that are associated with the main IODD file. Apart from that, tools shall not evaluate the file name; they always evaluate the file’s content. The device-specific file name is only intended for better legibility.

Adherence to the rules for file names makes it possible that all IODDs can be stored side-by-side in a single directory.

File names must not only be different in upper and lower case. Case sensitivity of default parts of file names shall be adhered to.

The following special characters are permitted in vendor name and device name: _, #, -

All files of the set of files belonging to a specific IODD shall have the same <vendor name> part in their file names. The <vendor name> should be the same for all IODDs of the same vendor. It is not required that the <vendor name> in the file name matches exactly the content of the DeviceIdentity/@vendorName attribute or the standard variable V_VendorName in the device. Usually, the latter two also contain the legal form of the company, e.g. “Inc.”, “AG”, “S.A.” but this is not included in the vendor name part of the file name.

5.1 Main IODD file

The file name shall follow the following rule:

<vendor name>**-**<device name>**-**<date of file creation>**-IODD**<schema version>**.xml**

e.g. VendorX-DeviceY-20110603-IODD1.1.xml

Contains information (in XML) about the identification of the device, communication characteristics, parameters, process data, and diagnosis data.

The IODD must always entirely contain texts in the PrimaryLanguage (English). The IODD may contain texts in further languages.

A style sheet for the vendor-specific description of Devices for a certain browser (optional):

e.g. VendorX-IODD1.1.xsl

The IODD shall not reference such style sheets with a processing instruction (<?xml-stylesheet … ?>).

5.2 Language files (optional)

To add support for additional languages after an IODD has been released, separate language files (in XML) may be created. Their file name must exactly match the name of the main IODD file, except that there is an additional language designation before the file name extension:

<vendor name>**-**<device name>**-**<date of file creation>**-IODD**<schema version>**-**<language>**.xml**

The “language” part follows ISO 639-1:2002. The “language” part shall correspond to the value of the ‘xml:lang’ attribute inside the language file. There shall be no additional language file for languages already covered in the main IODD file. The “language” part consists of two letters.

The ‘Text’ and ‘TextRedefine’ elements contained in the additional language file shall follow the same rules as specified for the respective elements in additional languages inside the main IODD.

e.g. VendorX-DeviceY-20110603-IODD1.1-ru.xml

Additional language file containing texts in Russian.

An IO-Link Tool shall select the appropriate language from the main IODD file or the accompanying language files according to its user interface language settings. A tool shall ignore files whose filename does not match to the naming convention of additional IODD files.

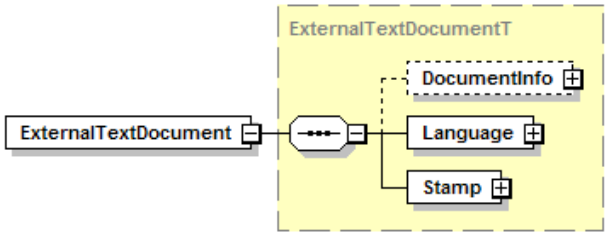


Figure 2 – Structure of language file

5.3 Image files (optional)

The file format shall be PNG (file extension .png, see RFC 2083 or ISO/IEC 15948:2004). The same rules for permitted characters apply as in section ‘Files’ (see above).

<vendor name>**-logo.png**
Vendor logo. 160 x 90 pixel, landscape format. The background of the logo should be transparent.

<vendor name>**-**<picture name>**-icon.png**
Device variant icon. 48 x 48 pixel.

<vendor name>**-**<picture name>**-pic.png**
Device variant picture. Min. 160 x 160 pixel, max. 320 x 320, square.

<vendor name>**-**<picture name>**-con-pic.png**
Device variant connection picture. Min. 160 x 160 pixel, max. 320 x 320, square.

The device variant icons and device variant pictures are referenced from the DeviceIdentity/ DeviceVariantCollection/DeviceVariant elements. The device variant connection pictures are referenced from the CommNetworkProfile/TransportLayers/PhysicalLayer/Connection elements. The referenced image files shall accompany the main IODD file for stamping and delivery.

5.4 Standard definitions files

IODD-StandardDefinitions1.1.xml
This file contains the definition of standardized variables, error types and events (see IO-Link Communication Specification Version 1.1) plus English language texts.

IODD-StandardDefinitions1.1-de.xml
Additional language file containing texts in German.

IODD-StandardUnitDefinitions1.1.xml
This file contains the definitions of all available unit codes plus English language texts.

IODD-StandardUnitDefinitions1.1-de.xml
Additional language file containing texts in German.

Those files are part of the standard and shall not be changed. Vendors of IO-Link Tools should use those files instead of hard-coding standardized things.

Additional language files for standard definitions files will be provided by the IODD subteam when needed on the IO-Link website.

5.5 Schema files

Schema files are needed to validate the structure of XML-files and to aid in editing.

IODD1.1.xsd
IODD-schema; includes the following sub-schemas:

IODD-Primitives1.1.xsd
includes basic schema elements

IODD-Datatypes1.1.xsd
includes schema elements for the definition of data types

IODD-Events1.1.xsd
includes schema elements for the definition of error types and events

IODD-Variables1.1.xsd
includes schema elements for the definition of variables

IODD-UserInterface1.1.xsd
includes schema elements for the definition of the user interface

IODD-Communication1.1.xsd
includes schema elements for the definition of the communication network profile

IODD-StandardDefinitions1.1.xsd
schema for the definition of system-specific elements used to validate the file IODD-StandardDefinitions1.1.xml and IODD-StandardUnitDefinitions1.1.xml.

6 Description mechanisms

6.1 Names of elements and attributes

Following one common pattern, the names of the elements begin with an uppercase letter while the names of the attributes begin with a lowercase letter. When names consist of several words, each word (except for the first in case of an attribute) starts with an uppercase letter. No separator character (like _) is used.

6.2 Ids

The values of the attribute ‘id’ shall follow the regular expression pattern:

“[A-Za-z][A-Za-z0-9 _-]*[A-Za-z0-9]”.

Ids shall be unique within the elements of the same type. The prefix “STD_” is reserved for ids in the standard definition files and shall not be used for elements in the main IODD.

6.3 Referencing

Each element that can be referenced within the IODD contains an explicit attribute ‘id’. The referencing element contains a type-dependent attribute with the following composition: <type>Id

Examples: textId, datatypeId, menuId, variableId

6.4 Text localization

All text components of the different languages which are referenced in the IODD are allocated in the ExternalTextCollection (for further information see “Language-Dependent Description Texts”).

The text components of the different languages are referenced in the relevant location according to a key (textId).

Further languages can be added in an appropriate file (see chapter 5.2).

The PrimaryLanguage in the IODD must be completely available. If there is a further language added in the IODD or in a separate language file, not all entries must be given. In this case, the interpreter has to go back to the entry of the PrimaryLanguage.

7 Device Description

For IO-Link Tools there is no such thing as a conformance class. Therefore IO-Link Tools shall support the IO-Link Communication Specification Version 1.1 and this IO Device Description Specification V1.1 completely. Furthermore IO-Link Tools shall interpret and use the content of IODDs completely.

IO-Link Tools distinguish devices by their VendorID and DeviceID, and the IODDs of a device by the IODD version they are based on and their DocumentInfo/@releaseDate. Tools shall prefer V1.0.1 IODDs over V1.0 IODDs, and within the IODD version newer IODDs over older IODDs. For V1.1 IO-Link devices there shall be only a single current (most recent) IODD based on V1.1, and if the device is compatible to IO-Link V1.0, additionally a single current IODD based on V1.0.1 or V1.0.

7.1 Notation of XML structure

The XML structure is hierarchical. As the whole structure is too complex to show in one figure, the description is split into a series of figures, starting with the root element and descending into the details.

Following each figure showing the structure of a particular section of the IODD, all the elements and their attributes are listed in the order in which they appear inside the figure.

The description of elements and attributes follows this pattern:

> **Element_or_attribute_name (Use[, XML_type])**
> Semantics of the element or attribute. If the element or attribute has a value, a possible default or fixed value is also described here.

Element_or_attribute_name is the name of the element or attribute. Remember that element names start with an uppercase letter while attribute names start with a lowercase letter.

Use is one of the following letters:

m	Mandatory
o	Optional
c	Conditional (depends on … see description)

XML_type is the XML schema data type of the element or attribute value (if applicable). Do not confuse this with the data types that the device's variables and process data may use. XML_type may be

- one of the basic XML types defined in REC-xmlschema-2-20041028. The namespace "xsd:" is omitted for brevity.
- the XML type xml:lang as defined in xml.xsd.
- one of the IODD XML types defined in IODD-Primitives1.1.xsd (see Table 1).

Table 1 – IODD XML types

XML_type	Defined as	Use
IdT	xsd:string with pattern: "[A-Za-z][A-Za-z0-9 _-]*[A-Za-z0-9]"	Used for an 'id' attribute at an element so it can be referenced.
RefT	xsd:string with pattern: "[A-Za-z][A-Za-z0-9 _-]*[A-Za-z0-9]" (same as IdT)	Used as a reference to some element that has an 'id' attribute.
SubindexT	xsd:unsignedByte restricted to 1..255 (0x01..0xFF)	For sub-adressing within an index.
BitCountT	xsd:unsignedShort	For bit offsets and bit lengths.
IsduLengthT	xsd:unsignedByte restricted to 1..232	For lengths (in octets) which must fit into an ISDU
DeviceIdT	xsd:unsignedInt restricted to 1..16777215 (0x000001..0xFFFFFF)	For a device ID.
CharacterEncodingT	xsd:string, either "UTF-8" or "US-ASCII"	The character encoding of a string.
VersionT	xsd:string with pattern: " V\d+(\.\d+){1,7}"	To express a version of e.g. the IO-Link specification, the IODD Checker,

		the IODD instance.
AccessRightsT	xsd:string, either "ro", "rw" or "wo"	Access rights read only, read-write or write-only.

Further restrictions to these XML types are mentioned directly at the XML type or in the element / attribute description.

7.2 Basic structure of the main IODD file

Figure 3 shows the basic structure of a device in a device description.

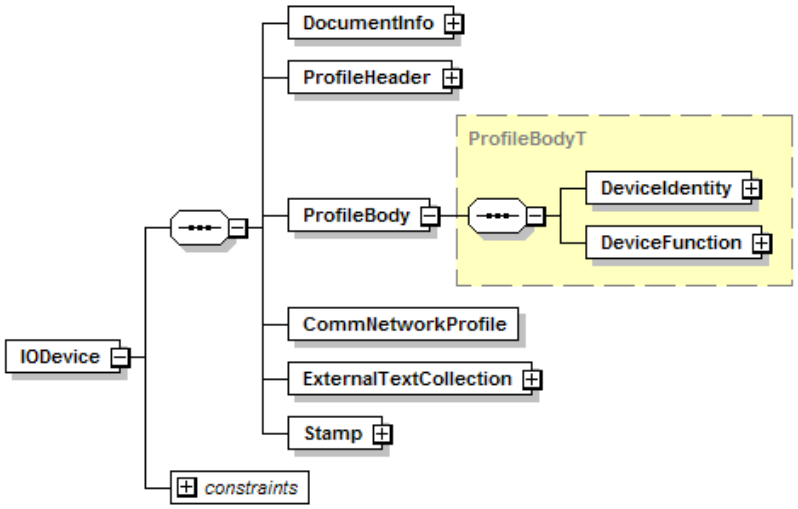


Figure 3 – Basic structure of main IODD file

7.3 Metainformation

7.3.1 DocumentInfo (m; o for language file)

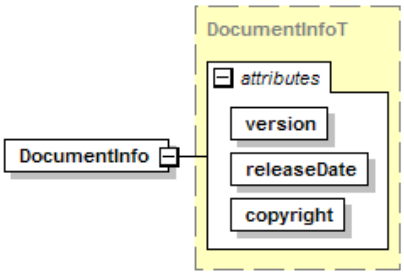


Figure 4 – DocumentInfo element

Here the vendor inserts the information for the IODD.

version (m, VersionT)
The ‘version’ attribute contains the version of the concrete instance and not the version of the IODD specification. The vendor shall increase this version for each official release of the IODD for a particular device.

releaseDate (m, date with pattern "\d{4}-\d{2}-\d{2}")
The date information in the IODD file name shall correspond to the ‘releaseDate’ attribute in the DocumentInfo element. There shall be no more than one official release of the IODD for a particular device per day. IO-Link Tools shall rely on this date for determining the newest version of the IODD for a device.

copyright (m, string)
Vendor-specific copyright text.

e.g.
File name: IO-Link-SampleDevice-20110603-IODD1.1.xml

DocumentInfo:
<DocumentInfo version="V5.17" releaseDate="2011-06-03" copyright="IO-Link Consortium"/>

7.3.2 ProfileHeader (m)

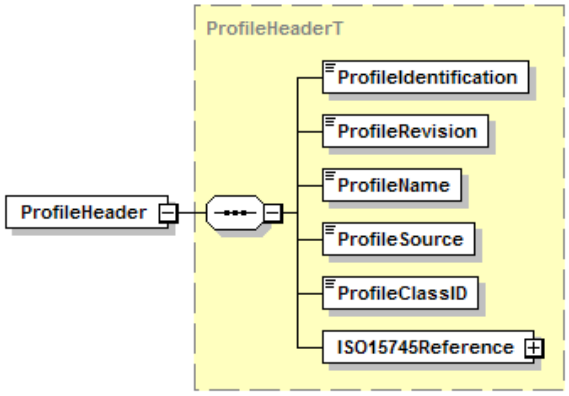


Figure 5 – ProfileHeader element

Within this element, the vendor shall give the following constant information in plain text.

ProfileIdentification (m, string)
Fixed to "IO Device Profile"

ProfileRevision (m, string)
Fixed to "1.1".

ProfileName (m, string)
Fixed to "Device Profile for IO Devices".

ProfileSource (m, string)
Fixed to "IO-Link Consortium".

ProfileClassID (m, NMTOKEN)
Fixed to "Device".

ISO15745Reference (m)
Information about the underlying ISO standard

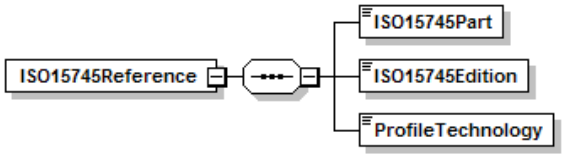


Figure 6 – ISO15745Reference element

ISO15745Part (m, positiveInteger)
Fixed to "1".

ISO15745Edition (m, positiveInteger)
Fixed to "1".

ProfileTechnology (m, string)
Fixed to "IODD".

7.3.3 ProfileBody (m)

The ProfileBody contains the description of identity and functionality of the device.

7.3.4 File validation

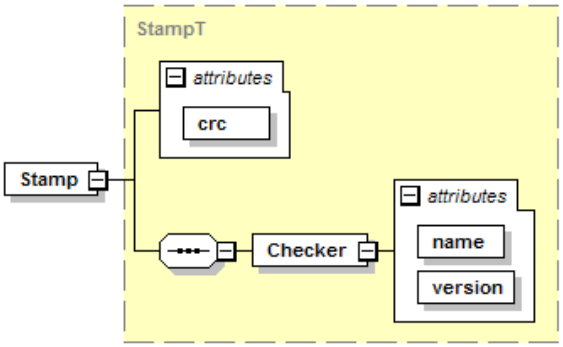


Figure 7 – Stamp element

There is a tool called “IODD Checker” that intensely tests the IODD if it is conformant to this specification. The Checker features a *check* and a *stamp* mode. In *check* mode, errors detected during the checking process are reported, but the file remains unchanged. In *stamp* mode, the Stamp element is always rewritten.

crc (m, unsignedInt)
If no errors are detected during the checking process, the ‘crc’ attribute is set to the CRC value calculated from the file contents. Otherwise, the ‘crc’ attribute is set to an invalid value. By checking the CRC, an IO-Link Tool can find out whether the IODD has been altered since the last successful check. In this case, the IODD should be rejected by the tool.

For the CRC, the CRC-32 algorithm is used (see section 8.1.1.6.2 of ITU-T recommendation V.42 (03/2002) or ISO/IEC 3309:1993). Before the actual calculation, the ‘crc’ attribute is set to an empty string and the checker inserts its name and version into the appropriate attributes. The generated CRC is then inserted into the ‘crc’ attribute.

Language files are stamped in the same way, but the CRC of the main IODD file is also included in the calculation. Before the actual calculation, the ‘crc’ attribute is set to an empty string and the checker inserts its name and version into the appropriate attributes. Only for CRC calculation the checker adds the CRC of the main IODD to the end of the language file. The generated CRC is then inserted into the ‘crc’ attribute.

Checker (m)
Identification of the IODD Checker version used to check and stamp this file. If there is a severe bug in a specific Checker version, or the method of calculating the CRC must be modified in the future, IO-Link Tools are able to adapt to this based on the Checker name and version.

name (m, string)
The name of the IODD Checker.

version (m, VersionT)
The version of the IODD Checker.

When writing a new IODD, before applying the IODD checker on it for the first time, it is recommended to set the attributes to the following values:

Stamp/@crc = "0"
Stamp/Checker/@name = "" (empty string)
Stamp/Checker/@version = "V0.0.0.0"

It is highly recommended, not to insert comments in or after the Stamp element.

7.4 Device identity

On import of a new IODD, IO-Link Tools shall use the pair vendorId and deviceId to decide wether this IODD describes a new device (catalog entries must be added) or this IODD is a new description of an already known device (catalog entries must be updated). This decision shall not be based on the filename of the IODD.

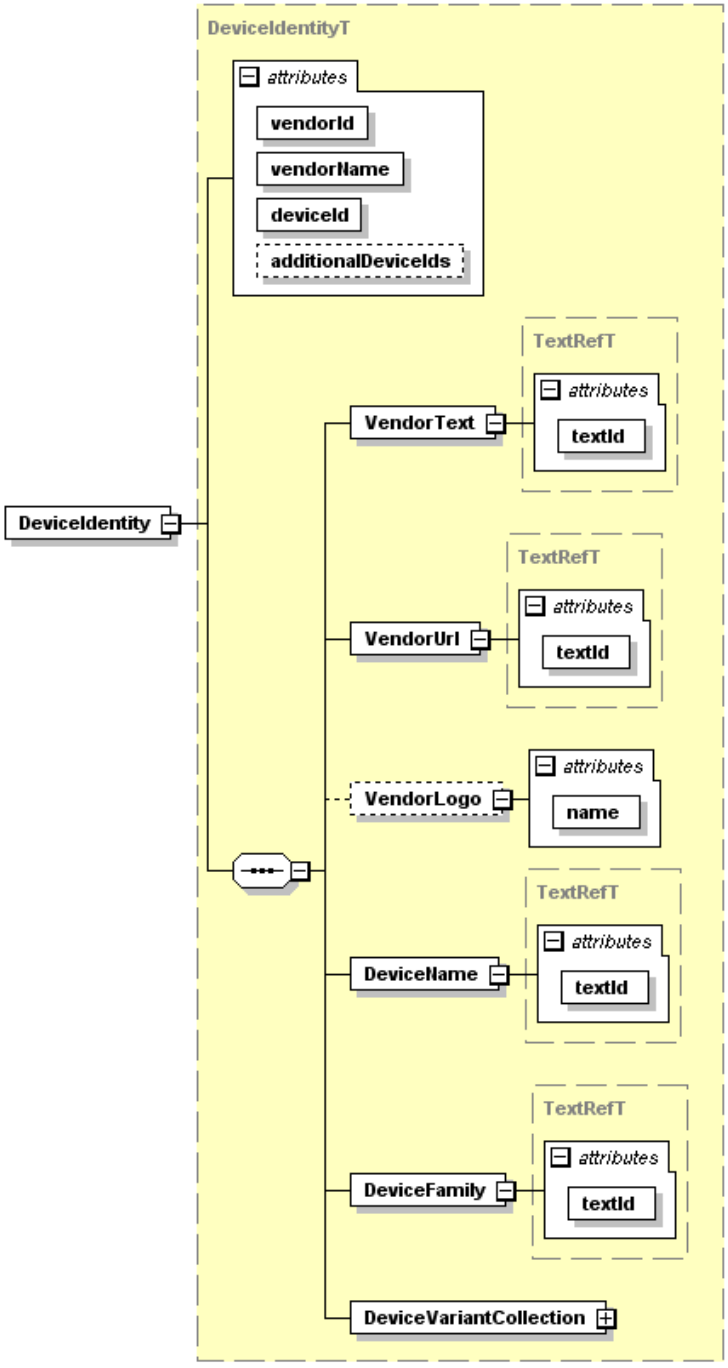


Figure 8 – DeviceIdentity element

vendorId (m, unsignedShort)
Unique identification of the vendor, assigned by the IO-Link Consortium. An IO-Link Tool shall display this id in decimal notation. The vendorId shall match the content of V_DirectParameters_1, subindex 8-9. The allowed value range is 1..65535 (0x0001..0xFFFF).

vendorName (m, string)
Name of the vendor of the device.

deviceId (m, DeviceIdT)
Vendor-internal unique identification of the device; an IO-Link Tool shall display this id in decimal notation. The deviceId shall match the content of V_DirectParameters_1, subindex 10-12.

additionalDeviceIds (o, list of DeviceIdT with min. one and max. 255 entries)
A list of device IDs which are supported by this device. In other words, this device may replace devices of the same vendor whose device IDs are listed in this attribute.

VendorText (m)
textId (m, RefT)
A text describing the vendor (a slogan).

VendorUrl (m)
textId (m, RefT)
The vendor's URL.

VendorLogo (o)
name (m, string with pattern "([\p{L}\d_#]+-)+logo\.png")
File name of the vendor's logo; in PNG format, 160 x 90 pixels. If the element 'VendorLogo' is used, the image file referenced by the attribute 'name' shall be present.

DeviceName (m)
textId (m, RefT)
Common name for all variants. IO-Link Tools may use this to group the device variants of a device in the device catalog.

DeviceFamily (m)
textId (m, RefT)
Vendor-specific classification of the devices. IO-Link Tools may use this for grouping devices in the device catalog.

7.4.1 Device variant collection

Using the Vendor ID and Device ID read out from an unknown IO-Link Device, it must be possible to uniquely find the latest version of the appropriate IODD. Thus is it not allowed that IO-Link Devices that differ in details described in ProfileBody/DeviceFunction or CommNetworkProfile share the same combination of Vendor ID and Device ID.

The things in which the devices may differ are those that are not "seen" by the IO-Link Tool, like:

- type of plug and length of cable
- materials: plastics, stainless steel
- shape: round, ..
- fastening: through-hole, bracket
- allowed environmental conditions: temperature range, humidity, shock resistance
- certificates: CE, UL

Devices that only differ in these things may use the same Vendor ID and Device ID and shall be described as different Device Variants in a single IODD. If the device supports ISDU Index

19 (V_ProductID), the value read from this ISDU must match exactly to the ‘productId’ attribute of exactly one DeviceVariant.

Examples for things in which devices may **not** differ:

- measurement ranges (with sensors)
- power range (with actuators)

There shall be at least one device variant.

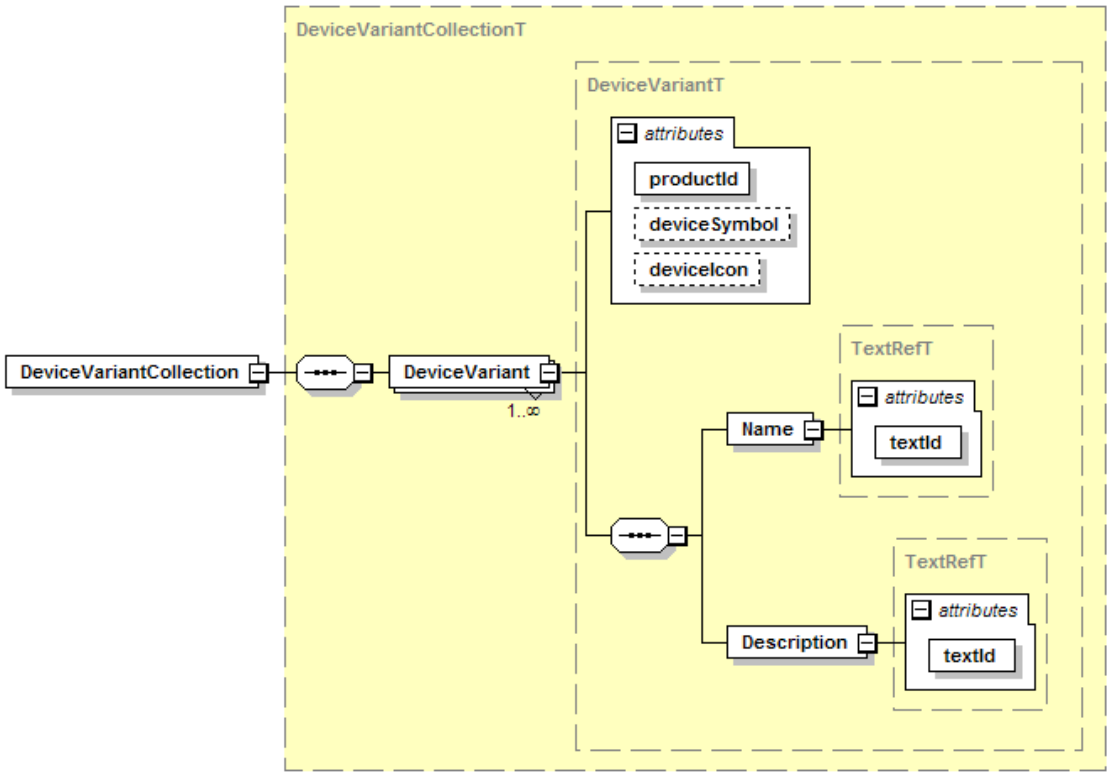


Figure 9 – DeviceVariantCollection element

productId (m, string)
Text; max. 64 octets UTF-8 coded. Uniquely identifies the product within the DeviceVariants. ‘productId’ in IODD corresponds to the ISDU standard parameter V_ProductID. If V_ProductID is not implemented in the device only one single device variant shall be referenced in the IODD.

deviceSymbol (o, string with pattern "([\p{L}\d_#]+-)+pic\.png")
File name of the device symbol. If this attribute is used, the referenced image file shall be present.

deviceIcon (o, string with pattern "([\p{L}\d_#]+-)+icon\.png")
File name of the device icon. If this attribute is used, the referenced image file shall be present.

Name (m)
textId (m, RefT)
Used to build the catalog entries for the device variants in the IO-Link Tool. Shall be unique for each DeviceVariant in all supported languages.

It shall correspond to the product name in the vendor’s catalogue or to the name which is labelled on the product.

Description (m)
textId (m, RefT)
Descriptive text of the device.

7.5 Device function

The entire functionality of the device is collected here. Parameters, process data, data types, error codes and events are defined. Their significances, addresses, and data fields are identified as well as a grouping of the views in menus is defined.

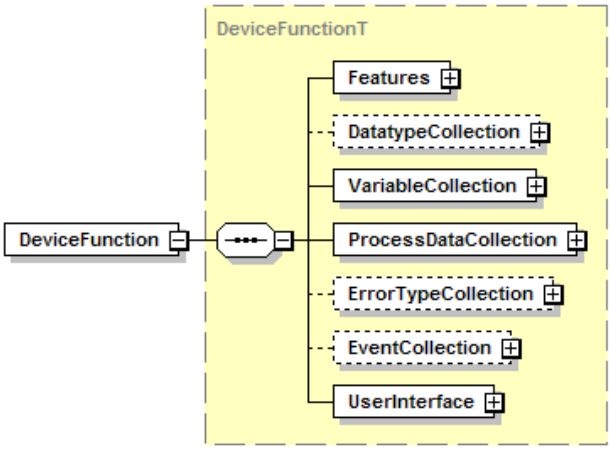


Figure 10 – DeviceFunction element

7.5.1 Features

Supported standardized features of the device are described.

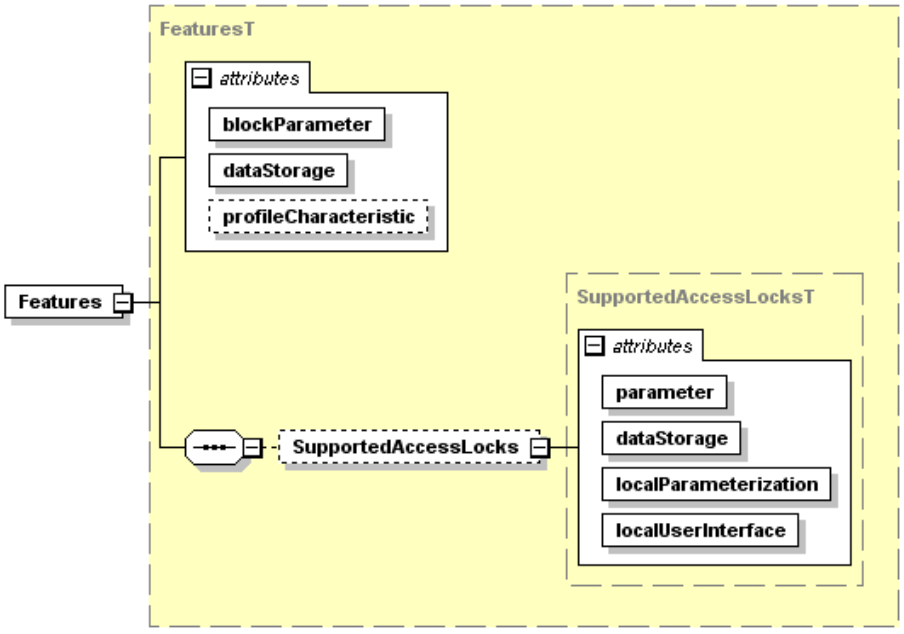


Figure 11 – Features element

blockParameter (m, boolean)
This attribute defines if a device supports the functionality of Block Parameter transmission. See IO-Link Communication Specification Version 1.1, chapter 10.3.5.

IO-Link Tools shall use Block Parameter transmission if the device supports it and if more than one variable is to be transferred due to a single user action.

Typical Block Parameter sequences:

Upload

- Send SystemCommand ParamUploadStart
- Upload all variables of the current user role
- Send SystemCommand ParamUploadEnd

Download

- Send SystemCommand ParamDownloadStart
- Download the changed (or all) variables of the current user role
- Send SystemCommand ParamDownloadEnd or ParamDownloadStore

dataStorage (m, boolean)
This attribute defines if a device supports data storage functionality.

profileCharacteristic (o, list of unsignedShort with min. one and max. 32 entries)
A list of Profile Identifiers (PID) which are supported by this device. This list decribes the supported profiles and function classes. See IO-Link Profile Smart Sensor Specification Version 1.0.

SupportedAccessLocks (c)
Shall be present if the standard variable V_DeviceAccessLocks is supported.

parameter (m, boolean)
Whether parameter access lock is supported.

dataStorage (m, boolean)
Whether data storage access lock is supported.

localParameterization (m, boolean)
Whether local parameterization access lock is supported.

localUserInterface (m, boolean)
Whether local user interface access lock is supported.

7.5.2 Data type collection

The DatatypeCollection incorporates all declarations for the reuse of data types (especially useful for records). There shall be no unreferenced Datatype elements. Standardized data types are described in the schema IODD-Datatypes1.1.xsd.

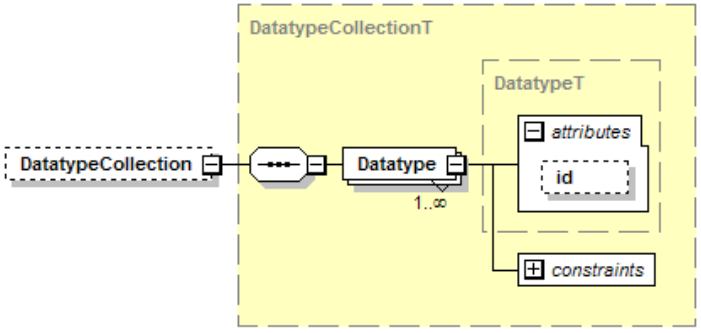


Figure 12 – DatatypeCollection element

For the Datatype element, this figure only shows the elements and attributes common to all data types. The actual selected data type needs additional elements and attributes. See chapter 7.5.3 for details.

id (c, IdT)
Datatype elements within the DatatypeCollection shall have an attribute ‘id’. Datatype elements outside of the DatatypeCollection shall not have an attribute ‘id’.

7.5.3 Data types

The IODD-Datatypes1.1.xsd schema provides derived types for all possible data types. The presence and type of required elements and attributes is checked by this schema.

Actually, the data types form the following hierarchy:

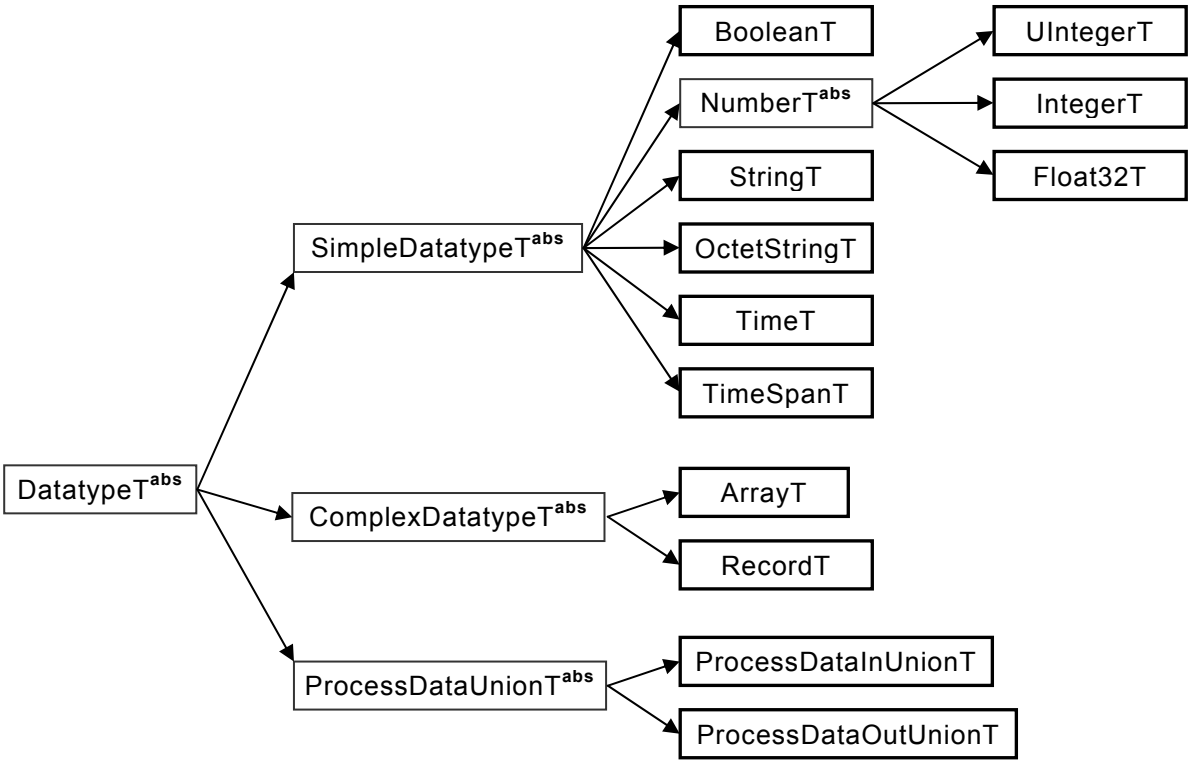


Figure 13 – Data type hierarchy

Each derivation adds elements and/or attributes appropriately.

Excursion on XML schema *abstract types*:
An abstract type can’t be used itself. Only non-abstract types which are derived from an abstract type can be used. The instance selects the desired derived type with xsi:type=”*name of the derived type*”.

This technique is used here with the ‘Datatype’ element to adapt the XML structure to the requirements of the specific data type.

For the serialization of the data types see IO-Link Communication Specification Version 1.1, chaper E.

7.5.3.1 Simple data types

The coding of simple data types is shown only for singular use which is characterized by

- Process data consisting of one simple data type
- Parameter consisting of one simple data type
- Subindex (>0) access on individual data items of parameters of complex data types (arrays, records)

7.5.3.1.1 General

When the Datatype element appears inside the DatatypeCollection, the attribute ‘id’ shall be present. Otherwise, the attribute ‘id’ shall not be present.

SingleValue and ValueRange elements are strongly typed.

Where SingleValue and / or ValueRange elements are permitted, the following rules shall be considered:

- When neither SingleValue nor ValueRange elements are given, the complete value range of the data type is allowed. When SingleValue(s) and / or ValueRange(s) are given, only these values are allowed.
- In ValueRanges, both the lowerValue and the upperValue are included in the range of allowed values.
- In ValueRanges, the lowerValue shall be less than the upperValue (not equal).
- SingleValues and ValueRanges shall not overlap.

7.5.3.1.2 BooleanT

Figure 14 shows the IODD representation of the data type BooleanT.

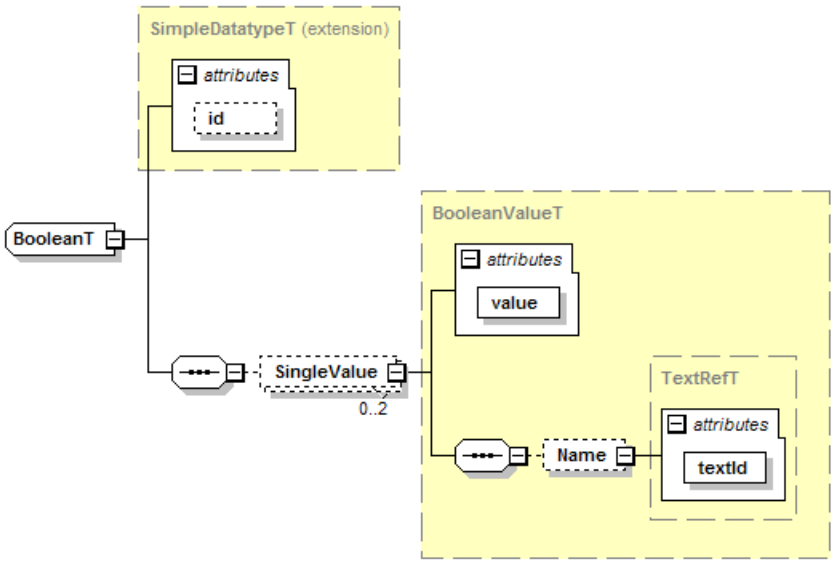


Figure 14 – BooleanT

Lexical representation:

Conforms to the representation of “boolean” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#boolean

Regular expression pattern: “true|false|1|0”

Example:

```
<Datatype xsi:type="BooleanT">
        <SingleValue xsi:type="BooleanValueT" value="false">
```

```
            <Name textId="TI_Inversion_Off"/>
        </SingleValue>
        <SingleValue xsi:type="BooleanValueT" value="true">
            <Name textId="TI_Inversion_On"/>
        </SingleValue>
</Datatype>
```

7.5.3.1.3 UIntegerT

Figure 15 shows the IODD representation of the data type UIntegerT.

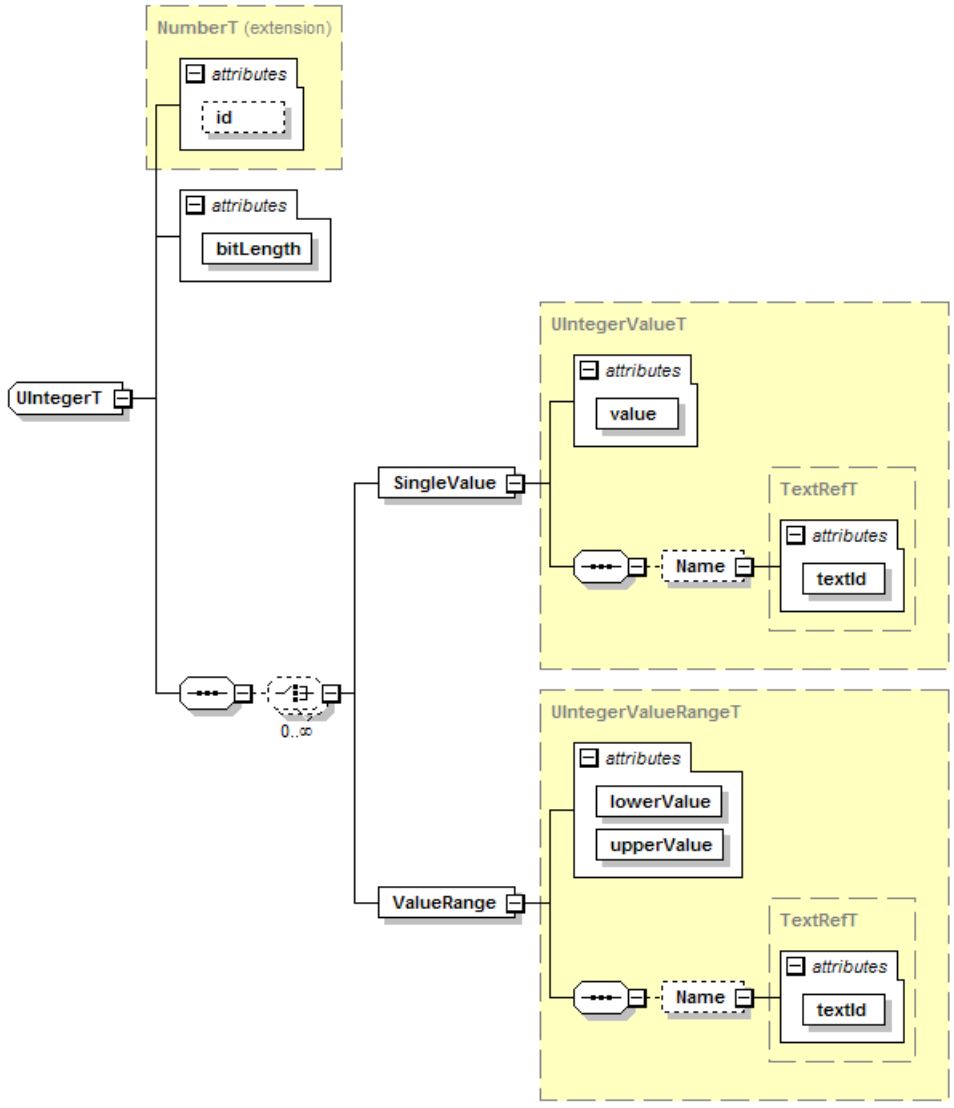


Figure 15 – UIntegerT

bitLength (m, BitCountT)
Specifies the size of the unsigned integer in bits. The allowed value range is 2..64.

Lexical representation:

Conforms to the representation of “unsignedLong” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#unsignedLong

Regular expression pattern: “+?\d+”

Example:

```
<Datatype xsi:type="UIntegerT" bitLength="8">
        <SingleValue xsi:type="UIntegerValueT" value="96">
                <Name textId="TI_System"/>
        </SingleValue>
</Datatype>
```

7.5.3.1.4 IntegerT

For the representation of the data type IntegerT in the IODD and an example see 7.5.3.1.3.

Lexical representation:

Conforms to the representation of “long” in XML Schema, see
http://www.w3.org/TR/xmlschema-2/#long

Regular expression pattern: “[+-]?\d+”

7.5.3.1.5 Float32T

Figure 16 shows the IODD representation of the data type Float32T.

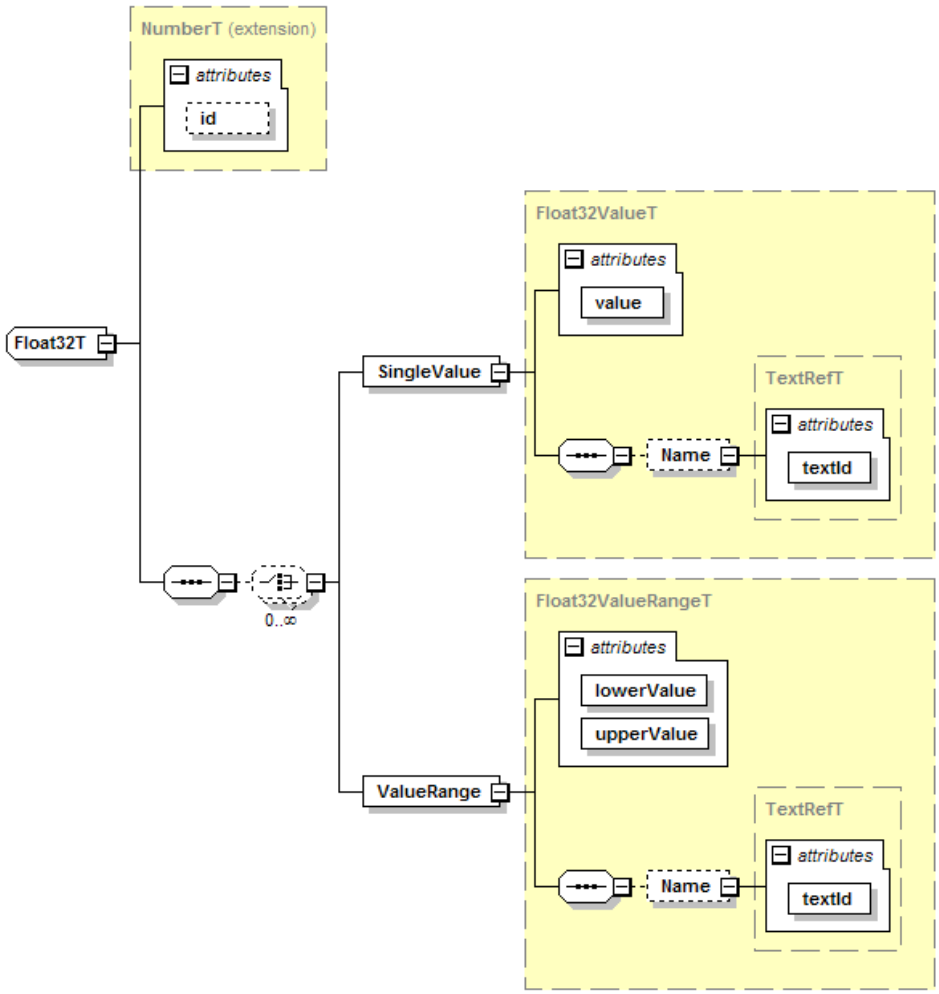


Figure 16 – Float32T

Lexical representation:

Conforms to the representation of “float” in XML Schema, see
http://www.w3.org/TR/xmlschema-2/#float

Regular expression pattern: “[+-]?\d+(\.\d+)?([eE][+-]?\d+)?|-?INF|NaN”

Example:

```
<Datatype xsi:type="Float32T">
        <SingleValue xsi:type="Float32ValueT" value="0.0">
                <Name textId="TI_Zero"/>
        </SingleValue>
        <ValueRange xsi:type="Float32ValueRangeT" lowerValue="1.0" upperValue="1000.0">
                <Name textId="TI_Valid"/>
        </SingleValue>
</Datatype>
```

7.5.3.1.6 StringT

Figure 17 shows the IODD representation of the data type StringT.

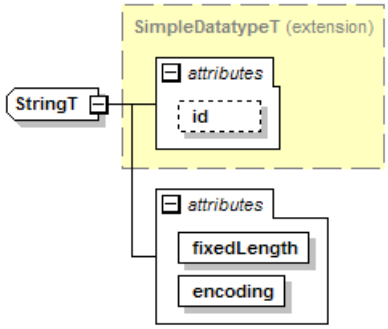


Figure 17 – StringT

fixedLength (m, IsduLengthT)
Specifies the length of the string in octets.

encoding (m, CharacterEncodingT)
The character encoding of the string. Note that US-ASCII consists of 7-bit characters only. Note that string constants in UTF-8 may need more than one octet per character.

Lexical representation:

Conforms to the representation of “string” in XML Schema, see
http://www.w3.org/TR/xmlschema-2/#string

Regular expression pattern: “.*” (No restriction, just the string.)

Special characters shall be coded according to the XML syntax. See REC-xml-20081126, chapter 2.4 Character Data and Markup.

& → &
' → ' (only required if inside a string enclosed in ' characters)
> → >
< → <
" → " (only required if inside a string enclosed in " characters)

Example:

```
<Datatype xsi:type="StringT" fixedLength="64" encoding="UTF-8"/>
```

7.5.3.1.7 OctetStringT

Figure 18 shows the IODD representation of the data type OctetStringT.

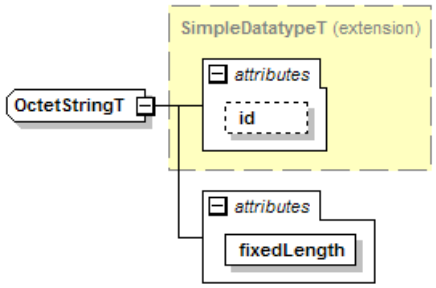


Figure 18 – OctetStringT

fixedLength (m, IsduLengthT)
Specifies the length of the octet string in octets.

Lexical representation:

Regular expression pattern: “(0x[0-9A-Fa-f][0-9A-Fa-f],)*0x[0-9A-Fa-f][0-9A-Fa-f]”

Example:

```
<Datatype xsi:type="OctetStringT" fixedLength="10"/>
```

7.5.3.1.8 TimeT

Figure 19 shows the IODD representation of the data type TimeT.

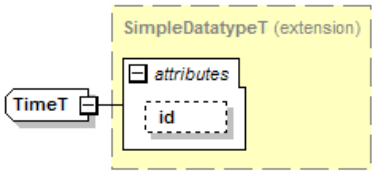


Figure 19 – TimeT

Lexical representation:

Follows the representation of “dateTime” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#dateTime, but is stricter:

Regular expression pattern: “\d{4}\-\d{2}\-\d{2}(T\d{2}:\d{2}:\d{2}(\.\d{1,3})?)?”

(yyyy-mm-dd[Thh:mm:ss[.fff]] where fff = fraction of a second, up to millisecond)

Example:

```
<Datatype xsi:type="TimeT"/>
```

7.5.3.1.9 TimeSpanT

Figure 20 shows the IODD representation of the data type TimeSpanT.

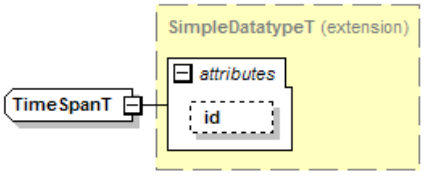


Figure 20 – TimeSpanT

The representation of TimeSpanT is shown in Table 2 and Table 3.

Table 2 – TimeSpanT

Data type name	Value range	Resolution	Length
TimeSpanT	Octet 1 to 8 (see Table 3): $-2^{63} \le i \le (2^{63}-1)$	$(1/2^{32})$ s	8 octets (64 bit integer)
NOTE 64 bit integer is the normal computer science data type			

Table 3 – Coding of TimeSpanT

Bit	7	6	5	4	3	2	1	0	Definitions
Octet 1	2^{63}	2^{62}	2^{61}	2^{60}	2^{59}	2^{58}	2^{57}	2^{56}	Time span in fractions of seconds as 64 bit integer. One unit is $1/(2^{32})$ s.
Octet 2	2^{55}	2^{54}	2^{53}	2^{52}	2^{51}	2^{50}	2^{49}	2^{48}	
Octet 3	2^{47}	2^{46}	2^{45}	2^{44}	2^{43}	2^{42}	2^{41}	2^{40}	
Octet 4	2^{39}	2^{38}	2^{37}	2^{36}	2^{35}	2^{34}	2^{33}	2^{32}	
Octet 5	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	
Octet 6	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 7	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 8	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
	MSB							LSB	MSB = Most significant bit LSB = Least significant bit

Lexical representation:

Follows the representation of “duration” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#duration, but is much stricter:

Regular expression pattern: “[+-]?PT\d+(\.\d{1,3})?S”

Example:

<Datatype xsi:type="TimeSpanT"/>

7.5.3.2 Complex data types

7.5.3.2.1 General

Complex data types are combinations of simple data types. Complex data types consist of several simple data types in a packed manner within a sequence of octets. Unused bit space shall be padded with "0".

The coding of simple data types within complex data types shall be the same as for singular use specified in chapter 7.5.3.1, except for:

BooleanT

The coding of BooleanT is only 1 bit wide. A value of 0 indicates false and a value of 1 indicates true. There is no padding to an octet.

UIntegerT and IntegerT

The coding of UIntegerT and IntegerT is as wide as indicated by the attribute ‘bitLength’. There is no padding to 1 / 2 / 4 / 8 octets.

7.5.3.2.2 Arrays

Figure 21 shows the IODD representation of the data type ArrayT.

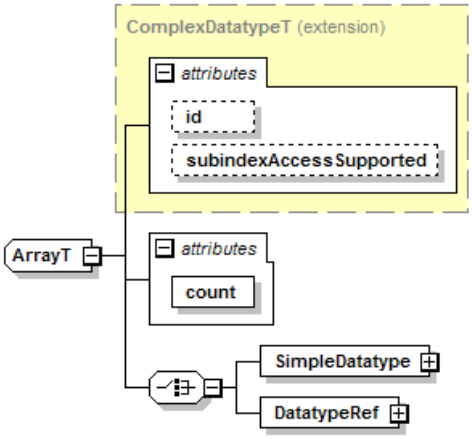


Figure 21 – ArrayT

subindexAccessSupported (o, boolean)
If this attribute is present and set to false, individual data items of the array cannot be accessed via their subindex. It is only possible to access the complete array via subindex 0.

count (m, SubindexT)
Specifies the fixed number of data items in the array.

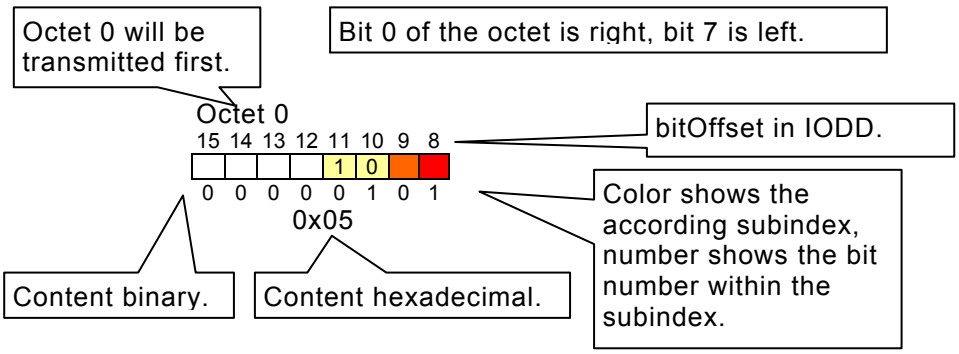
The SimpleDatatype element allows any of the types derived from SimpleDatatypeT. Instead of defining the simple data type inside the array definition, it is also possible to reference the definition of a simple data type from the DatatypeCollection with DatatypeRef/@datatypeId.

Lexical representation:

There is no lexical representation for a value of type ArrayT.

Examples

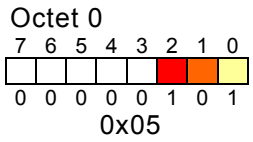
Notation:



Bit array

```
<Datatype xsi:type="ArrayT" count="3">
        <SimpleDatatype xsi:type="BooleanT"/>
</Datatype>
```

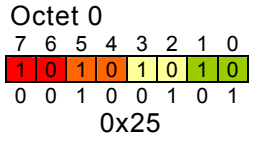
Subindex	Value
1	true
2	false
3	true



Integer array

```
<Datatype xsi:type="ArrayT" count="4">
        <SimpleDatatype xsi:type="IntegerT" bitLength="2"/>
</Datatype>
```

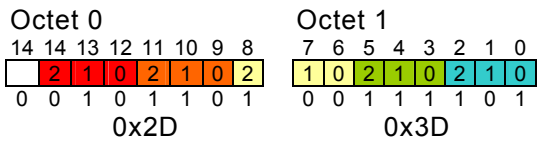
Subindex	Value
1	0
2	-2
3	1
4	1



Integer array

```
<Datatype xsi:type="ArrayT" count="5">
        <SimpleDatatype xsi:type="IntegerT" bitLength="3"/>
</Datatype>
```

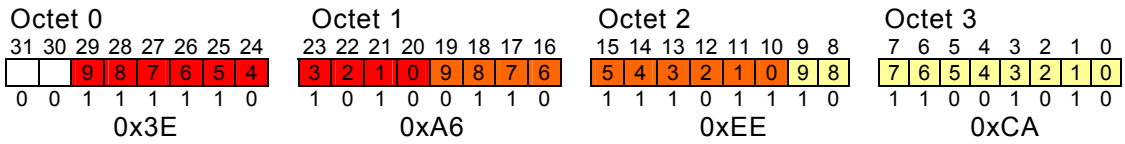
Subindex	Value
1	2
2	6
3	4
4	7
5	5



Integer array

```
<Datatype xsi:type="ArrayT" count="3">
        <SimpleDatatype xsi:type="IntegerT" bitLength="10"/>
</Datatype>
```

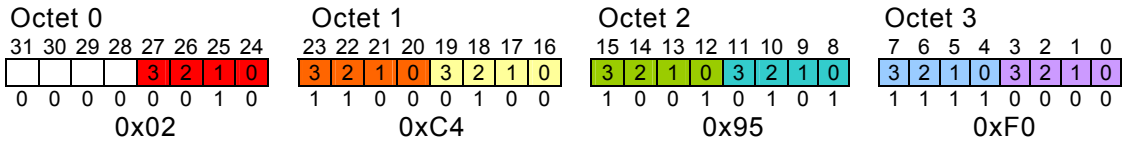
Subindex	Value
1	0x3EA
2	0x1BB
3	0x2CA



Integer array

```
<Datatype xsi:type="ArrayT" count="7">
        <SimpleDatatype xsi:type="IntegerT" bitLength="4"/>
</Datatype>
```

Subindex	Value
1	2
2	12
3	4
4	9
5	5
6	15
7	0



7.5.3.2.3 Records

Figure 22 shows the IODD representation of the data type RecordT.

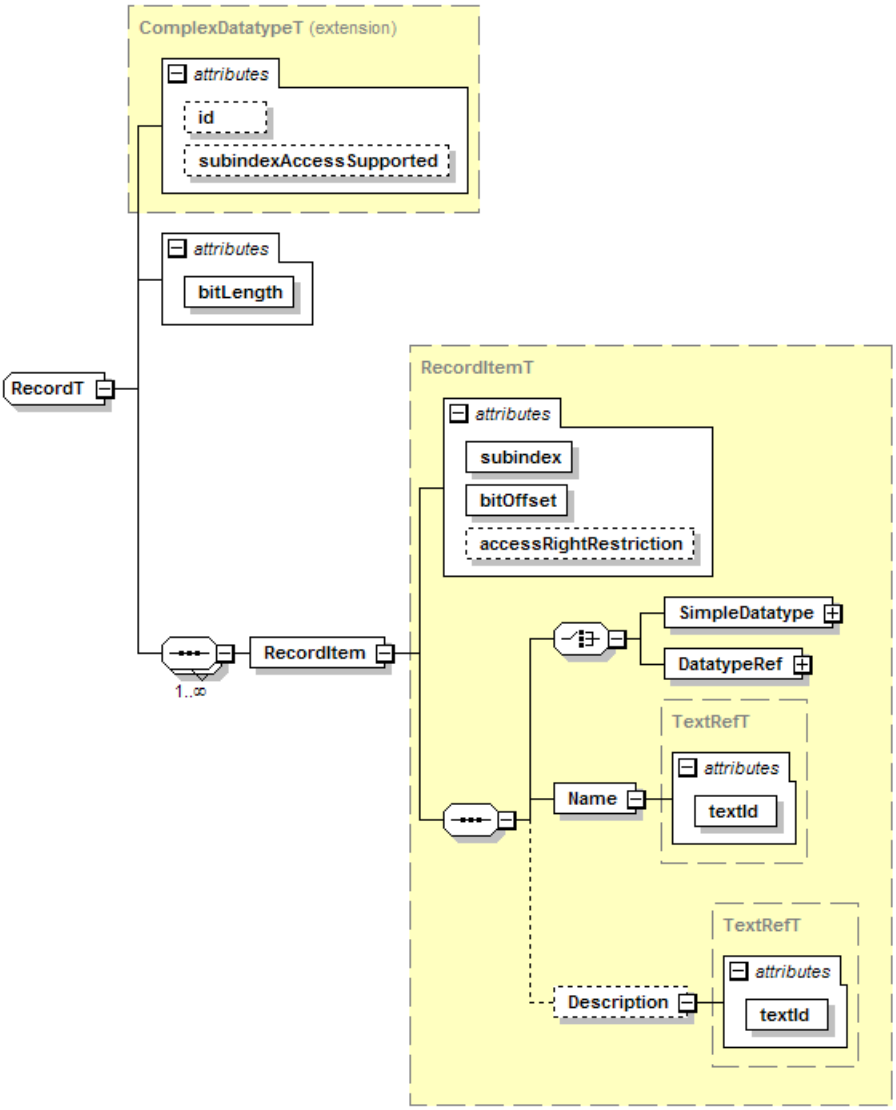


Figure 22 – RecordT

subindexAccessSupported (o, boolean)
If this attribute is present and set to false, individual record items cannot be accessed via their subindex. It is only possible to access the complete record via Subindex 0.

bitLength (m, BitCountT)
Specifies the total length of the record in bits. The allowed value range is 1..1856.

RecordItem (m)
An individual item of a record, addressed by the subindex.

subindex (m, SubindexT)
Specifies the Subindex assigned to this record item. The record items shall be ordered by Subindex within the record. The subindex shall be unique within the RecordItems of a Record.

It is recommended that the Subindices occur in increasing order within the octet sequence. If Subindices are placed in previously unused areas of the octet sequence, one might deviate from this recommendation. If compatible extensions are foreseen, it is better to reserve enough Subindices for the unused areas.

bitOffset (m, BitCountT)
bit position of the record item within the octet sequence. The record items within a record shall not overlap. The allowed value range is 0..1855.

accessRightRestriction (o, AccessRightsT)
This attribute is only applicable for service parameter, not for process data.

Individual record items may have less access rights than the record in general. This is indicated by the attribute 'accessRightRestriction'. For the access to the complete record, this means:

- If the Record is 'rw' and the record item is restricted to 'ro' the device shall tolerate (ignore) the data written to this Subindex.
- If the Record is 'rw' and the record item is restricted to 'wo', the IO-Link Tool shall ignore data read from this Subindex.

Within the record item, the SimpleDatatype element allows any of the types derived from SimpleDatatypeT. Instead of defining simple data types inside the record definition, it is also possible to reference the definition of simple data types from the DatatypeCollection with DatatypeRef/@datatypeId.

Name (m)
textId (m, RefT)
Assigns a human readable name to the record item. This name shall be displayed additionally by the IO-Link Tool.

Description (o)
textId (m, RefT)
Contains a description of the RecordItem (e.g. information text, help, etc.)

Lexical representation:

There is no lexical representation for a value of type RecordT.

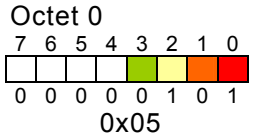
Examples:

Regarding the notation see 7.5.3.2.2.

Several Booleans in an Octet

```
<Datatype xsi:type="RecordT" bitLength="4">
        <Name textId="TI_Switches"/>
        <RecordItem subindex="1" bitOffset="0">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="1">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch2"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="2">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch3"/>
        </RecordItem>
        <RecordItem subindex="4" bitOffset="3">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch4"/>
        </RecordItem>
</Datatype>
```

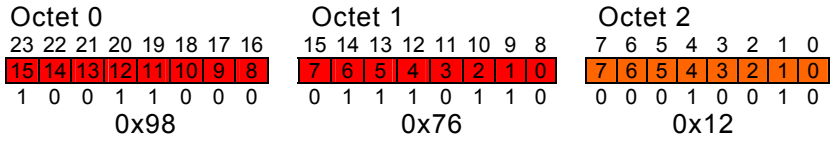
RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	BooleanT	–	0	true
2	2	BooleanT	–	1	false
3	3	BooleanT	–	2	true
4	4	BooleanT	–	3	false



A word and an octet

```
<Datatype xsi:type="RecordT" bitLength="24">
        <Name textId="TI_Values"/>
        <RecordItem subindex="1" bitOffset="8">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TI_Value1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="8"/>
                <Name textId="TI_Value2"/>
        </RecordItem>
</Datatype>
```

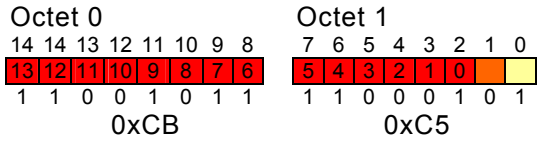
RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	16	8	0x9876
2	2	UIntegerT	8	0	0x12



Analog value and two signal bits

```
<Datatype xsi:type="RecordT" bitLength="16">
        <Name textId="TI_ProcessData"/>
        <RecordItem subindex="1" bitOffset="2">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="14"/>
                <Name textId="TI_AnalogValue"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="1">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Signal2"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="0">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Signal1"/>
        </RecordItem>
</Datatype>
```

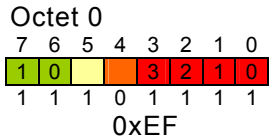
RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	14	2	0x32F1
2	2	BooleanT	–	1	false
3	3	BooleanT	–	0	true



Boolean and enumerations in an octet

```
<Datatype xsi:type="RecordT" bitLength="8">
        <Name textId="TI_ComplexSettings"/>
        <RecordItem subindex="1" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="4"/>
                <Name textId="TI_Enum1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="4">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch1"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="5">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch2"/>
        </RecordItem>
        <RecordItem subindex="4" bitOffset="6">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="2"/>
                <Name textId="TI_Enum2"/>
        </RecordItem>
</Datatype>
```

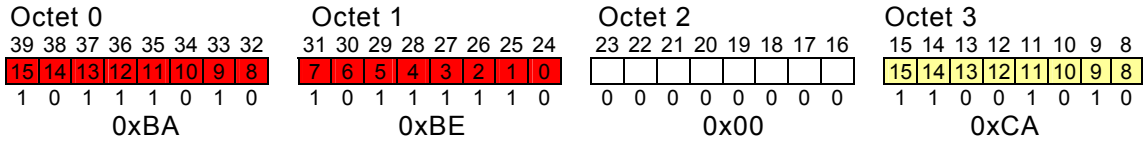
RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	4	0	0xF
2	2	BooleanT	–	4	false
3	3	BooleanT	–	5	true
4	4	UIntegerT	2	6	0x3

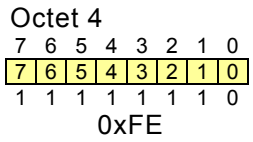


With a gap (reserved area for future extension)

```
<Datatype xsi:type="RecordT" bitLength="40">
        <Name textId="TI_Gap"/>
        <RecordItem subindex="1" bitOffset="24">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TI_Value1"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TI_Value2"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	16	24	0xBABE
2	3	UIntegerT	16	0	0xCAFE

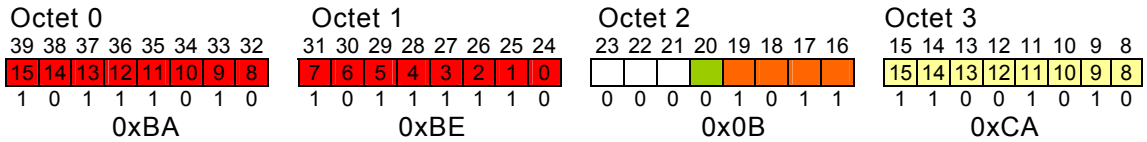


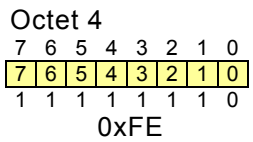


Previous example, extended with two record items

```
<Datatype xsi:type="RecordT" bitLength="40">
        <Name textId="TI_GapFilled"/>
        <RecordItem subindex="1" bitOffset="24">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TI_Value1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="16">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="4"/>
                <Name textId="TI_Enum"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TI_Value2"/>
        </RecordItem>
        <RecordItem subindex="4" bitOffset="20">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Switch"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	16	24	0xBABE
2	2	UIntegerT	4	16	0xB
3	3	UIntegerT	16	0	0xCAFE
4	4	BooleanT	–	20	false

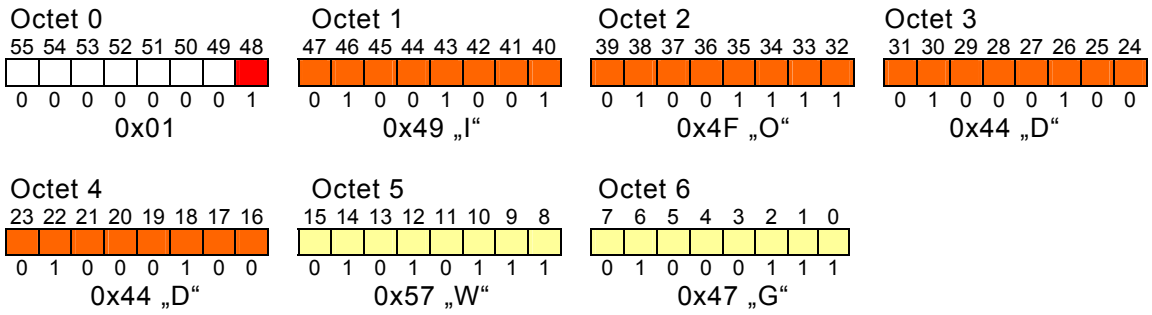




Strings

```
<Datatype xsi:type="RecordT" bitLength="49">
        <Name textId="TI_String"/>
        <RecordItem subindex="1" bitOffset="48">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Valid"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="16">
                <SimpleDatatype xsi:type="StringT" fixedLength="4"/>
                <Name textId="TI_Text1"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="0">
                <SimpleDatatype xsi:type="StringT" fixedLength="2"/>
                <Name textId="TI_Text2"/>
        </RecordItem>
</Datatype>
```

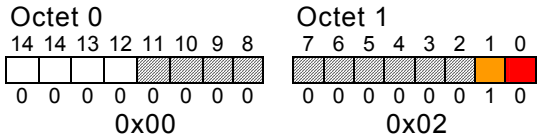
RecordItem	Subindex	Datentyp	fixedLength	bitLength	bitOffset	Value
1	1	BooleanT	–	–	48	true
2	2	StringT	4	–	16	„IODD“
3	3	StringT	2	–	0	„WG“



Two signal bits with reserved space

```
<Datatype xsi:type="RecordT" bitLength="12">
        <Name textId="TI_ProcessData"/>
        <RecordItem subindex="1" bitOffset="0">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Signal2"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="1">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TI_Signal1"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	BooleanT	–	0	false
2	2	BooleanT	–	1	true



7.5.3.3 Process data union data types

The types ProcessDataInUnionT and ProcessDataOutUnionT are restricted to the description of the process data standard variables (Index 40 and 41) in IODD-StandardDefinitions1.1.xml and thus are not allowed in a normal IODD. The IO-Link Tool shall take the data type of the appropriate ProcessDataIn / ProcessDataOut element. If more than one ProcessDataIn / ProcessDataOut element is given, it is necessary to select the currently valid element by evaluating the Condition elements.

7.5.3.3.1 ProcessDataInUnionT

A ProcessDataInUnionT corresponds to the data type used in ProcessDataCollection/ ProcessData/ProcessDataIn.

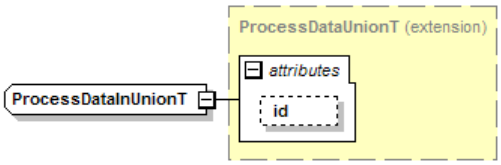


Figure 23 – ProcessDataInUnionT

7.5.3.3.2 ProcessDataOutUnionT

A ProcessDataOutUnionT corresponds to the data type used in ProcessDataCollection/ ProcessData/ProcessDataOut.

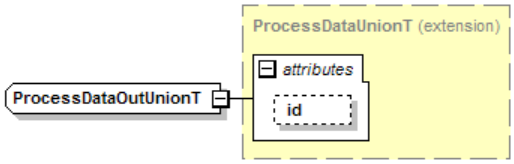


Figure 24 – ProcessDataOutUnionT

7.5.4 Variable collection

All parameters of the device are included here. Standard parameters are defined in IODD-StandardDefinitions1.1.xml and are referenced by StdVariableRef. DirectParameterOverlay allows defining a Record which is being layed over the DirectParameterPage 2 (DirectParameters 16 – 31). All other device-specific variables are named under ‘Variable’.

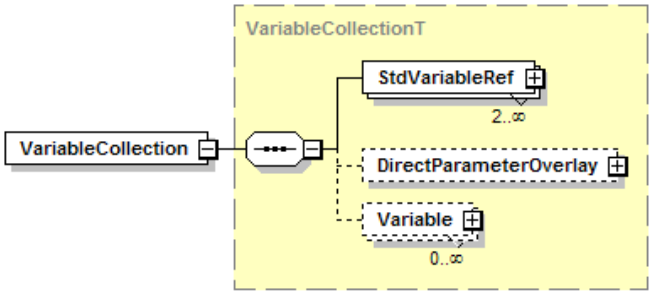


Figure 25 – VariableCollection element

All the variables that the device supports (i.e. the referenced standard variables, the direct parameter overlay and the vendor specific variables) shall have unique Names in all supported languages.

7.5.4.1 StdVariableRef

Here it is described, which of the standard variables are used. They are referenced here by an explicit key. Because direct parameters are mandatory, the variables V_DirectParameters_1 and V_DirectParameters_2 shall always be referenced. All standard ISDU variables marked with the attribute mandatory=”true” in the IODD-StandardDefinitions1.1.xml shall be referenced if the device supports ISDU access. The optional standard variables V_ProcessDataInput and V_ProcessDataOutput shall only be referenced if there is at least one ProcessDataIn / ProcessDataOut element in the ProcessDataCollection.

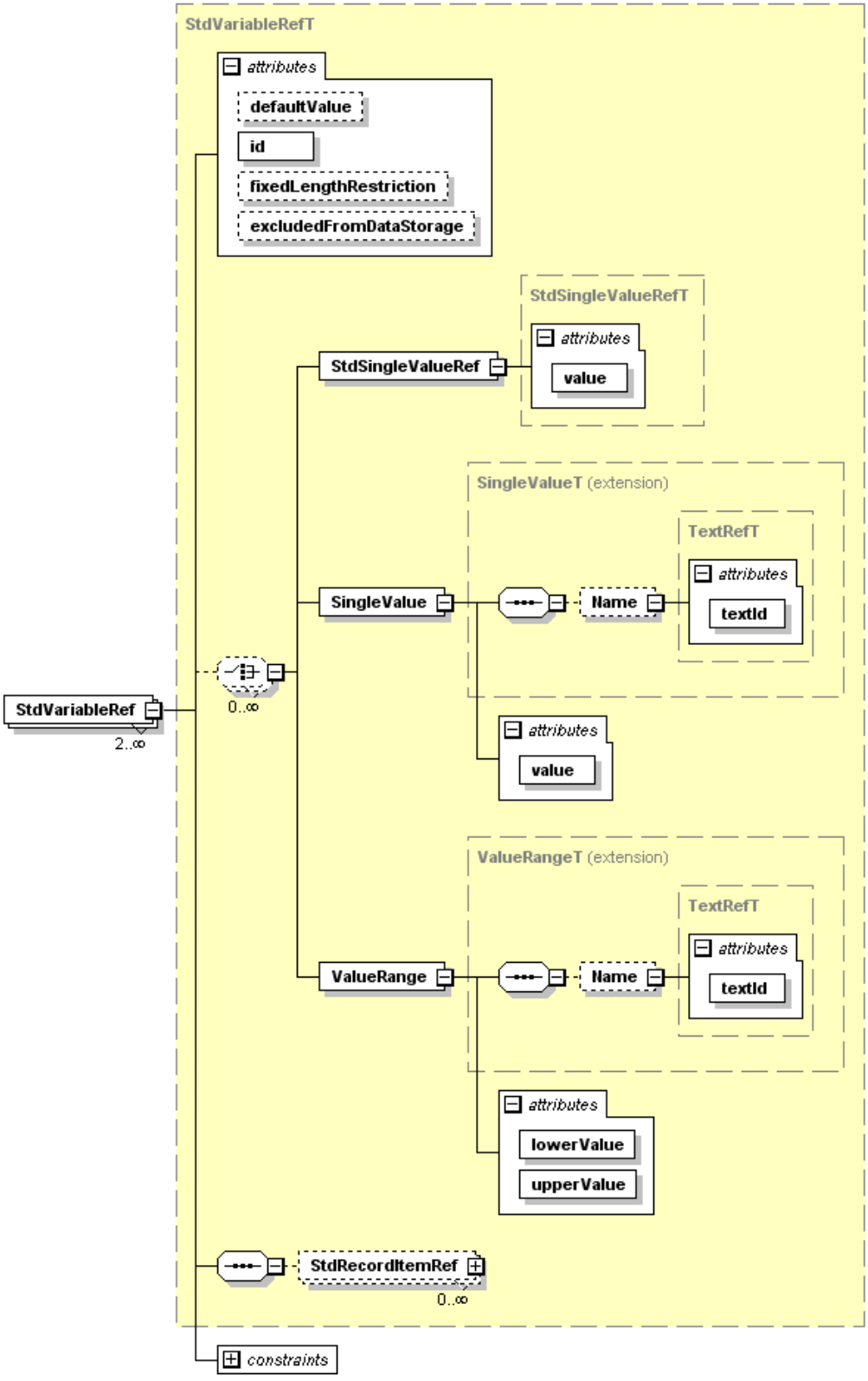


Figure 26 – StdVariableRef element

id (m, IdT)
This id is special since it can be both starting and end point of a referencing process. As end point of the referencing process, it contains the key of those variables within the IODD. As starting point, it references to a standard variable.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the standard variable. Offline default value; it always refers to the complete variable. If the variable is a record, use StdRecordItemRef element(s) to specify default values for individual record items. On a variable of type array, the specified defaultValue shall be applied to all array members. For references to V_ProductID, V_ProcessDataInput or V_ProcessDataOutput this attribute shall not be specified.

fixedLengthRestriction (o, SubindexT)
Only applicable to standard variables of type string, octet string or array. Standard variables of types string or octet string have a 'fixedLength' attribute describing the maximum length that the IO-Link standard allows. A Device may implement standard variables with (octet) string shorter than what the IO-Link standard allows.

Standard Variables of type array have a 'count' attribute describing the maximum size that the IO-Link standard allows. A Device may implement standard variables with arrays shorter than what the IO-Link standard allows.

'fixedLengthRestriction' shall be less or equal to the 'fixedLength' (on standard variables of type string or octetstring) or 'count' (on standard variables of type array). If 'fixedLengthRestriction' is used with V_ApplicationSpecificTag, it shall be greater or equal to 16.

excludedFromDataStorage (o, boolean)
If set to true, indicates that the contents of the standard variable are not stored with the data storage mechanism. This attribute shall not be set to true if the accessRights of the standard variable are not "rw", which limits its use to V_ApplicationSpecificTag, V_DeviceAccessLocks and V_OffsetTime. The default is "false".

Allowed values:
Only applicable to the standard variable V_SystemCommand and V_OffsetTime.

> **StdSingleValueRef (o)**
> Specifies a single supported standard value. The 'value' attribute must match the 'value' attribute of a SingleValue defined at the standard variable.
>
> **SingleValue (o)**
> Specifies a single supported vendor-specific value with an optional name.
>
> **ValueRange (o)**
> Specifies a supported vendor-specific value range with an optional name.

In addition to the above, the following rules shall apply for referencing for standard variables with StdVariableRef or StdRecordItemRef:

- When neither SingleValue nor ValueRange nor StdSingleValueRef elements are given, the standard variable's value range as defined in IODD-StandardDefinitions1.1.xml is taken.
- When SingleValue(s) or ValueRange(s) or StdSingleValueRef(s) are given, only these values are allowed.
- SingleValues and ValueRanges shall not overlap with SingleValues at the standard variable, no matter whether these are referenced by StdSingleValueRef or not (i.e. standard values can't be redefined in a vendor specific way).

StdRecordItemRef (o)
Used to specify additional information for RecordItems of standard variables. At least one of the optional attributes and elements shall be present.

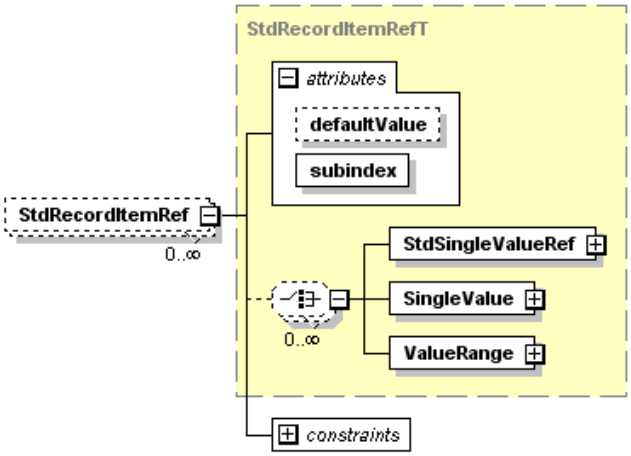


Figure 27 – StdRecordItemRef element

subindex (m, SubindexT)
Used for addressing the record item within the record. The subindex shall be unique within the StdRecordItemRefs of a StdVariableRef.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the RecordItem. Offline default value.

Specifying allowed values for a RecordItem using StdSingleValueRef, SingleValue and / or ValueRange is only applicable to the standard variable V_DirectParameters_1, subindex 16 (system command for devices without ISDU support). The meaning and the rules regarding these elements shall be the same as with the StdVariableRef element shown above.

7.5.4.2 DirectParameterOverlay

This element corresponds to the device-specific data within the DirectParameter page. If the DirectParameterOverlay is used, TextRedefines should also be added to provide names for each used DirectParameter octet (see chapter 7.7).

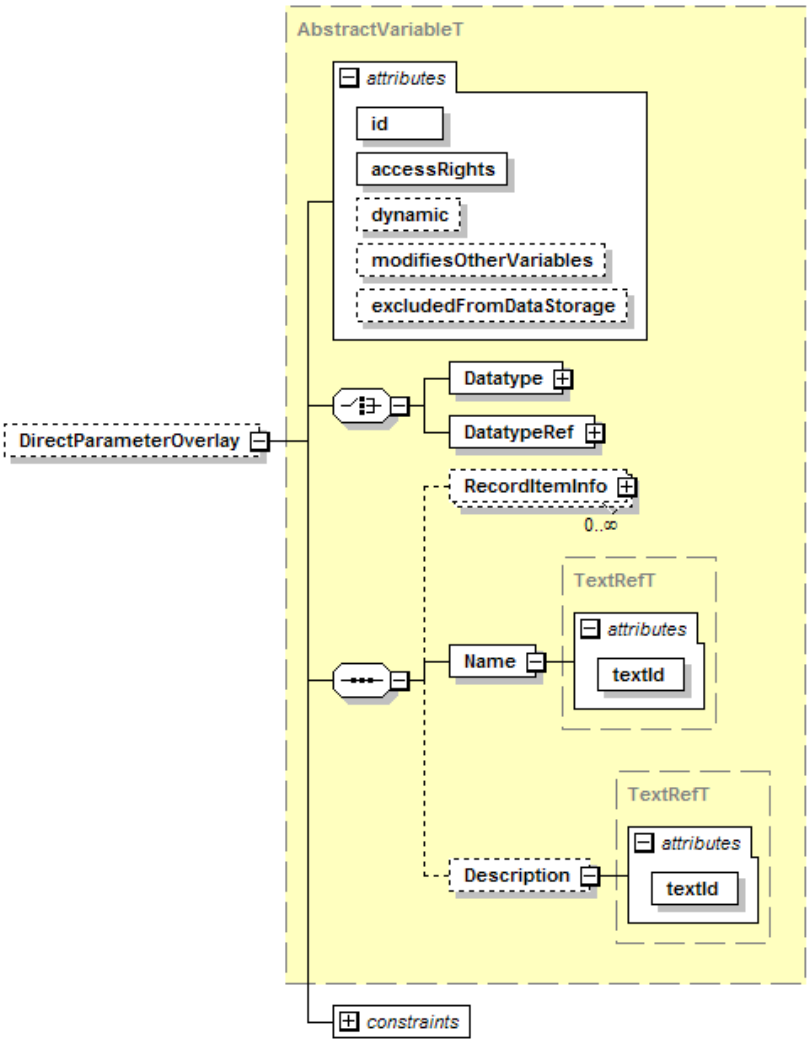


Figure 28 – DirectParameterOverlay element

id (m, IdT)
As the end point of a referencing process, it contains the key of the variable within the IODD. The id of any standard variable shall not be used as id for the DirectParameterOverlay, even if the standard variable is not referenced from this IODD.

accessRights (m, AccessRightsT)
“ro” read-only,
“wo”, write-only,
“rw”, read-write

dynamic (o, boolean)
Serves as information, whether the variable is autonomously changed by the device. This attribute shall not be set to true if the accessRights of the DirectParameterOverlay are not “rw” or “ro”. The default is “false”.

modifiesOtherVariables (o, boolean)
If set to true, indicates that a write access to the DirectParameterOverlay (or to any of its subindices) may change the value of other variables. IO-Link Tools should re-load the variables of the device after a write access to this variable. This attribute shall not be set to true if the accessRights of the DirectParameterOverlay are not “rw” or “wo”. The default is “false”.

excludedFromDataStorage (o, boolean)
If set to true, indicates that the contents of the DirectParameterOverlay are not stored with the data storage mechanism. This attribute shall not be set to true if the accessRights of the DirectParameterOverlay are not “rw”. The default is “false”.

Datatype (c)
Directly given data type (see Note below)

DatatypeRef (c)
Reference to a data type that was defined in the DatatypeCollection (see Note below)

RecordItemInfo (o)
Contains additional information for record items. See 7.5.4.4.

Name (m)
textId (m, RefT)
Contains the name of the variable

Description (o)
textId (m, RefT)
Contains a description of the variable (e.g. information text, help, etc.)

Note: The data type shall be a record with a minimum length of 1 bit and a maximum length of 128 bits. The last octet of this record is mapped to the last octet of the direct parameter page 2.

Example

RecordItem	Subindex	Datentyp	bitLength	bitOffset
1	1	UIntegerT	16	32
2	2	UIntegerT	32	0

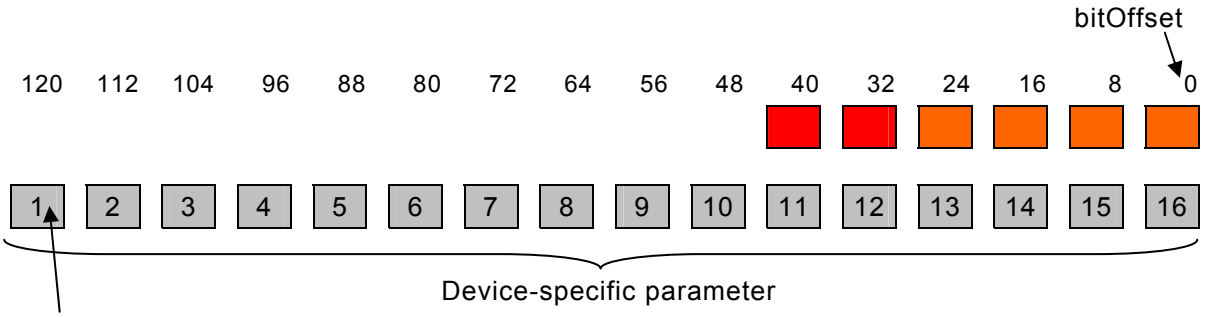


Figure 29 – Direct parameter overlay

Note: The communication of direct parameters is octet oriented.

- For record items, which cross an octet boundary the consistency cannot be guaranteed.
- If an octet contains more than one record item, the subindex access will influence all contained record items or parts.
- For record items, which cross an octet boundary, the device cannot rely on the order of the single accesses. This means, the device shall tolerate intermediate values that may exceed the allowed value range.

Recommendation: Use DirectParameterOverlay only for devices that do not support ISDU access.

7.5.4.3 Variable

Contains the description of a device parameter.

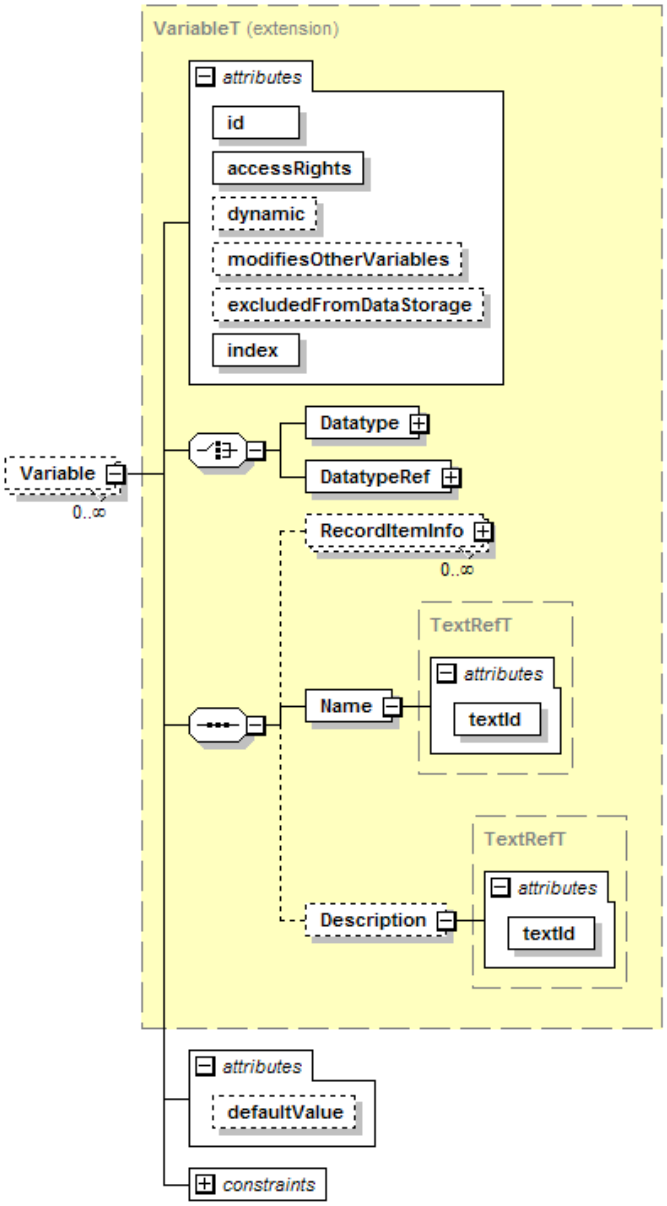


Figure 30 – Variable element

id (m, IdT)
As the end point of a referencing process, it contains the key of the variable within the IODD. The id of any standard variable shall not be used as id for the Variable, even if the standard variable is not referenced from this IODD.

accessRights (m, AccessRightsT)
“ro” read-only,
“wo”, write-only,
“rw”, read-write

dynamic (o, boolean)
Serves as information, whether the variable is autonomously changed by the device. This attribute shall not be set to true if the accessRights of the variable are not “rw” or “ro”. The default is “false”.

modifiesOtherVariables (o, boolean)
If set to true, indicates that a write access to this variable (or to any of its subindices) may change the value of other variables. IO-Link Tools should re-load the variables of the device after a write access to this variable. This attribute shall not be set to true if the accessRights of the variable are not “rw” or “wo”. The default is “false”.

excludedFromDataStorage (o, boolean)
If set to true, indicates that this variable’s contents are not stored with the data storage mechanism. This attribute shall not be set to true if the accessRights of the variable are not “rw”. The default is “false”.

index (m, unsignedShort)
Index for the addressing of a variable. Shall be in the range for vendor specific indices or profile specific indices that are not already described as standard variables in IODD-StandardDefinitions1.1.xml.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the variable. Offline default value; it always refers to the complete variable. If the variable is a record, use RecordItemInfo element(s) to specify default values for individual record items. On a variable of type array, the specified defaultValue shall be applied to all array members.

Datatype (c)
Directly given data type

DatatypeRef (c)
Reference to a data type that was defined in the DatatypeCollection

RecordItemInfo (o)
Only applicable if the variable is of type record. Contains additional information for record items. See 7.5.4.4.

Name (m)
textId (m, RefT)
Contains the name of the variable

Description (o)
textId (m, RefT)
Contains a description of the variable (e.g. information text, help, etc.)

7.5.4.4 RecordItemInfo

For variables of type RecordT, contains optional attributes for a RecordItem addressed by the subindex. At least one of the optional attributes shall be present.

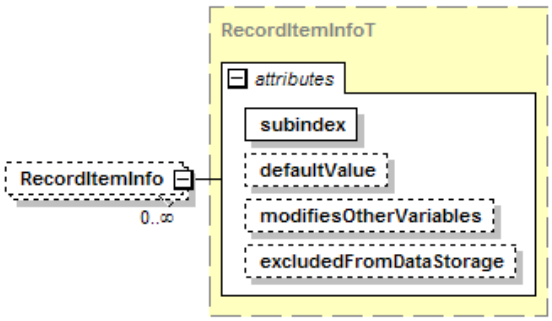


Figure 31 – RecordItemInfo element

subindex (m, SubindexT)
Used for addressing the record item within the record.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the record item. Contains the default value for the RecordItem.

modifiesOtherVariables (o, boolean)
If set to true, indicates that a write access to this subindex may change the value of other variables. IO-Link Tools should re-load the variables of the device after a write access to this subindex. This attribute shall not be set to true if the accessRights of the subindex are not “rw” or “wo”. This attribute shall not be specified both on the DirectParameterOverlay/Variable element and a subordinated RecordItemInfo element. The default is “false”.

excludedFromDataStorage (o, boolean)
If set to true, indicates that this subindex’s contents are not stored with the data storage mechanism. This attribute shall not be set to true if the accessRights of the subindex are not “rw”. This attribute shall not be specified both on the DirectParameterOverlay/Variable element and a subordinated RecordItemInfo element. The default is “false”.

7.5.5 Process data collection

Contains all process data of the device

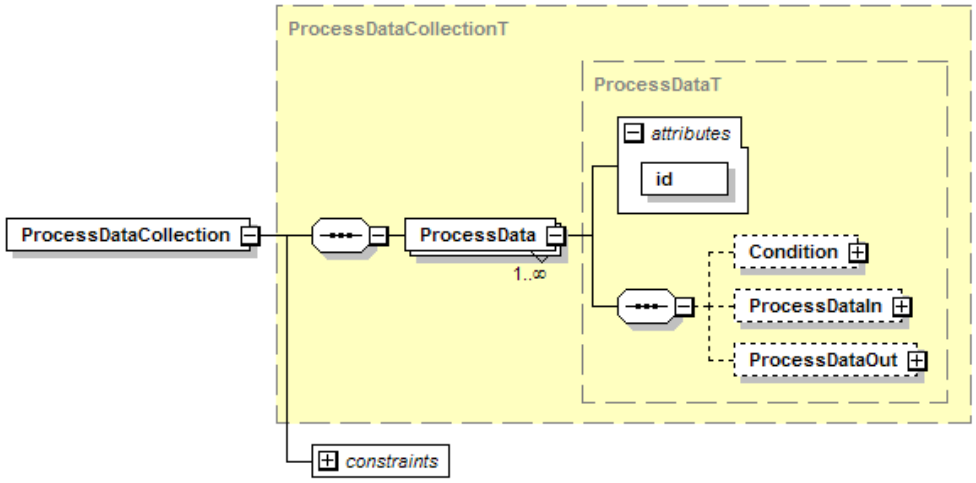


Figure 32 – ProcessDataCollection element

The element ProcessData may occur multiple times in the collection. If ProcessData occurs more than once,

- all the ProcessData elements shall contain a Condition element
- the attributes ‘variableId’ and ‘subindex’ in the Condition elements shall be the same (there shall only be exactly one variable / record item used for the switching of the process data)
- the attribute ‘value’ in the Condition elements shall be unique within the ProcessData elements
- the attribute ‘bitLength’ in the ProcessDataIn elements shall be the same for all ProcessData
- the attribute ‘bitLength’ in the ProcessDataOut elements shall be the same for all ProcessData

- the variable / record item referenced in the Condition elements selects the currently valid ProcessData element when its value matches the ‘value’ attribute of the Condition element

The attribute ‘id’ shall be unique within all the elements ProcessData, ProcessDataIn and ProcessDataOut.

ProcessData (m)

id (m, IdT)
Explicit id of the ProcessData

Condition (o)
Serves to switch between different ProcessData.

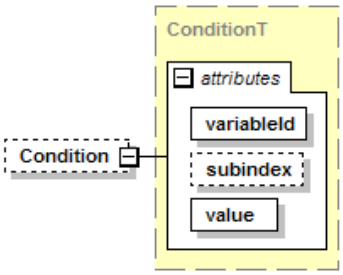


Figure 33 – Condition element

variableId (m, RefT)
References a variable. The variable shall be of data type BooleanT, IntegerT, UIntegerT or RecordT. The variable shall have a default value if it is not of type RecordT.

subindex (c, SubindexT)
This attribute shall be given if and only if the referenced variable is of type RecordT. Used for addressing the record item within the record. The record item shall be of data type BooleanT, IntegerT or UIntegerT and shall have a default value.

value (m, unsignedByte)
Must be a valid value for the variable / record item. This attribute can only hold values 0..255, thus limiting the possible IntegerT and UIntegerT values. Also, BooleanT condition values shall be entered as 0 for false and 1 for true.

ProcessDataIn (o)
Description of the input process data

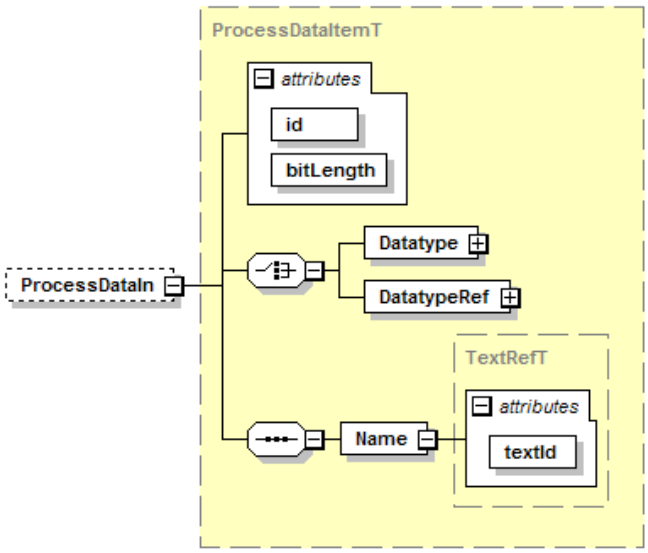


Figure 34 – ProcessDataIn element

id (m, IdT)
Explicit id of the ProcessDataIn description

bitLength (m, BitCountT)
Length of the input process data (in bits). The allowed value range is 1..256.

It shall represent the underlying ProcessDataIn data type in a bit granular manner. For record data types this bitLength shall equal the 'bitLength' attribute of the record.

The value of the DirectParameterPage 1, subindex 5 (Process Data In), shall be calculated from the 'bitLength' attribute value by the following formula:

```
If bitLength <= 16 then
        ProcessDataIn = bitLength
Else
        ProcessDataIn = bitLength rounded up to the next multiple of 8
End If
```

Name (m)
textId (m, RefT)
Name specification of the input process data

ProcessDataOut (o)
Description of the output process data

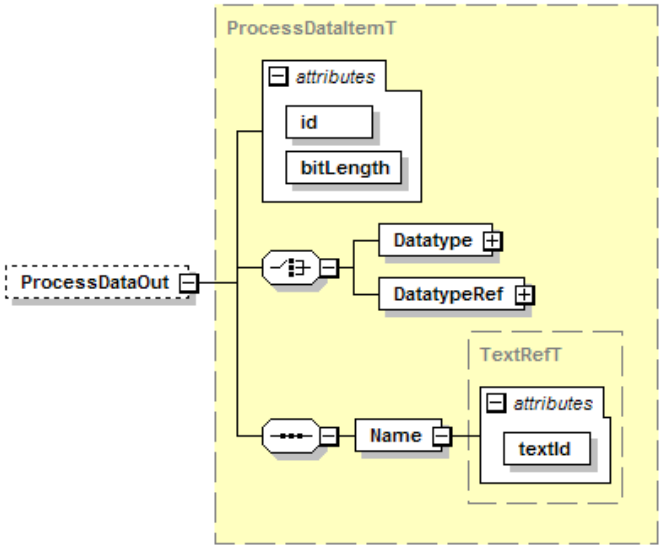


Figure 35 – ProcessDataOut element

id (m)
Explicit id of the ProcessDataOut description

bitLength (m, BitCountT)
Length of the output process data (in bits). The allowed value range is 1..256.

The description of the 'bitLength' attribute at the ProcessDataIn element above also applies here, but for Process Data Out (DirectParameterPage 1, subindex 6).

Name (m)
textId (m, RefT)
Name specification of the output process data

7.5.6 Error type collection

All error types that the device may return are collected here. There are system defined error types (code=128) and vendor specific error types (code=129), see IO-Link Communication Specification Version 1.1, annex C. The system defined error types are described in IODD-StandardDefinitions1.1.xml and referenced by 'StdErrorTypeRef', while the vendor specific error types are specified with 'ErrorType'.

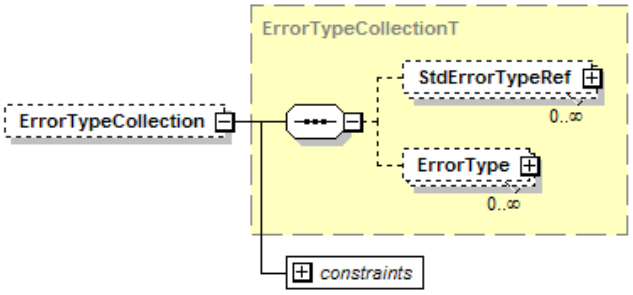


Figure 36 – ErrorTypeCollection element

StdErrorTypeRef (o)
Standard error types are referenced by their 'additionalCode'.

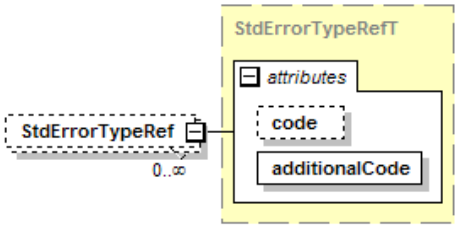


Figure 37 – StdErrorTypeRef element

code (o, unsignedByte)
Fixed to 128 by the IO-Link specification.

additionalCode (m, unsignedByte)
The additional code. Shall be unique within the 'StdErrorTypeRef' elements, and shall reference one of the error types defined in IODD-StandardDefinitions1.1.xml.

ErrorType (o)
Vendor specific error type, identified by its 'additionalCode'.

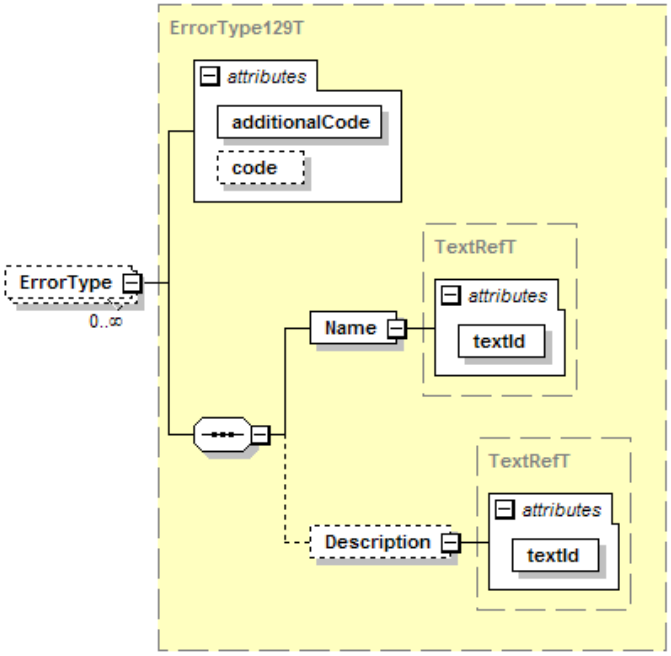


Figure 38 – ErrorType element

code (o, unsignedByte)
Fixed to 129 by the IO-Link specification.

additionalCode (m, unsignedByte)
The additional code. Shall be unique within the 'ErrorType' elements.

Name (m)
textId (m, RefT)
Use this text for the error message.

Description (o)
textId (m, RefT)
Use this text for the possible cause of the error and the remedy.

Error Types shall be handled by IO-Link Tools.

Since a device can respond to an ISDU with an ErrorType, IO-Link Tools shall list all incoming ErrorTypes during an up- or download. Up- or downloads shall not be interrupted. If a tool

supports device replication and the device responds with a single ErrorType, this replication shall be interrupted.

7.5.7 Event collection

All events that the device may return are collected here. There are system defined events and vendor specific events, see IO-Link Communication Specification Version 1.1, annex D. The system defined events are described in IODD-StandardDefinitions1.1.xml and referenced by 'StdEventRef', while the vendor specific events are specified with 'Event'.

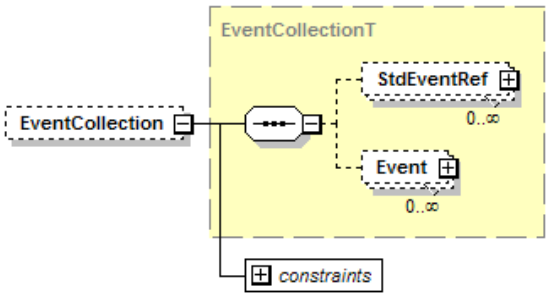


Figure 39 – EventCollection element

StdEventRef (o)
Indicates that the device may return the standard event identified by the 'code'.

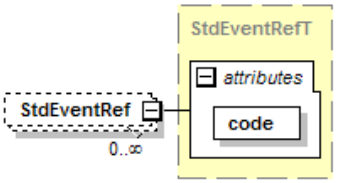


Figure 40 – StdEventRef element

code (m, unsignedShort)
The event code that identifies the standard event described in IODD-StandardDefinitions1.1.xml.

Event (o)
Describes a vendor specific event.

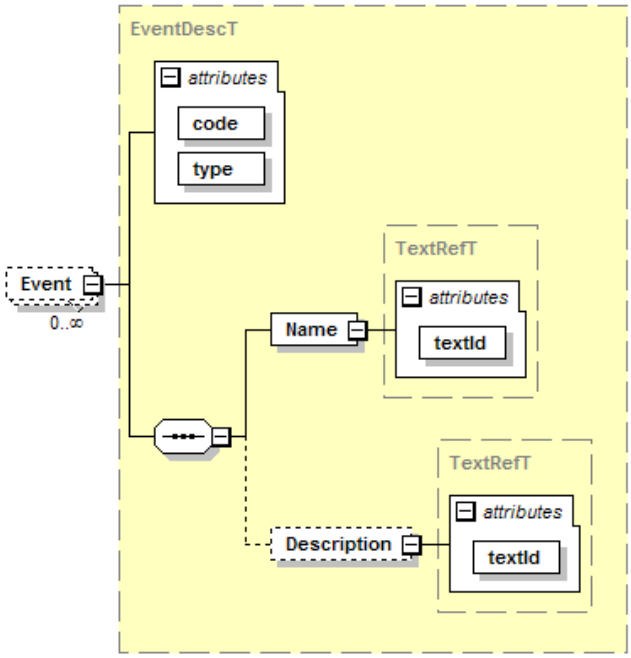


Figure 41 – Event element

code (m, unsignedShort)
The event code that identifies the vendor specific event. Shall be in the range for vendor specific or profile specific event codes.

type (m, string)
The TYPE part of the EventQualifier (see IO-Link Communication Specification Version 1.1, chapter A.6.4). One of “Notification”, “Warning” or “Error”.

Note that the MODE depends on the TYPE, SOURCE is always 0 (device application) and INSTANCE is always 4 (application).

Name (m)
textId (m, RefT)
Use this text for the event message.

Description (o)
textId (m, RefT)
Use this text for the possible cause of the event and the remedy.

7.5.8 User interface

Contains the menus of the device

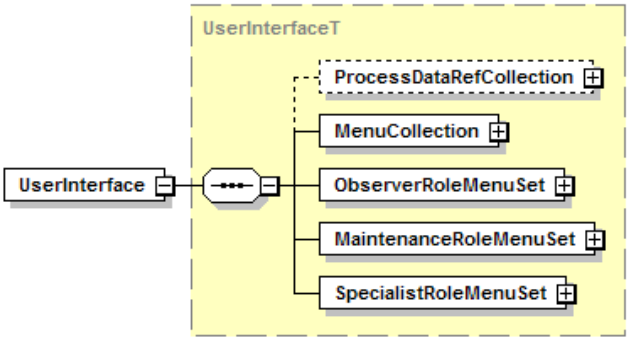


Figure 42 – UserInterface element

7.5.8.1 ProcessDataRef collection (o)

Some IO-Link masters support read access to the process data indipendently of the device. Process data read his way are shown in a separate menu by the IO-Link Tools for these masters. This collection allows defining how the process data read via the master are to be displayed.

On the other hand, a device may support the optional standard variables V_ProcessDataInput / V_ProcessDataOutput for access to the process data. Usually these variables are then referenced from the Observation or Diagnosis menu. The 'VariableRef' or 'RecordItemRef' controls how the process data read from the device are to be displayed.

Even if the device supports V_ProcessDataInput / V_ProcessDataOutput, when the 'VariableRef' or 'RecordItemRef' uses attributes to control how the process data is displayed, the ProcessDataRefCollection should be entered using the same attribute values.

If an IO-Link Tool wants to display process data and the IODD does not contain a corresponding ProcessDataRef for it, the tool shall display the process data just according to its data type. The tool shall not try to find the display format by walking the menus searching for V_ProcessDataInput / V_ProcessDataOutput references.

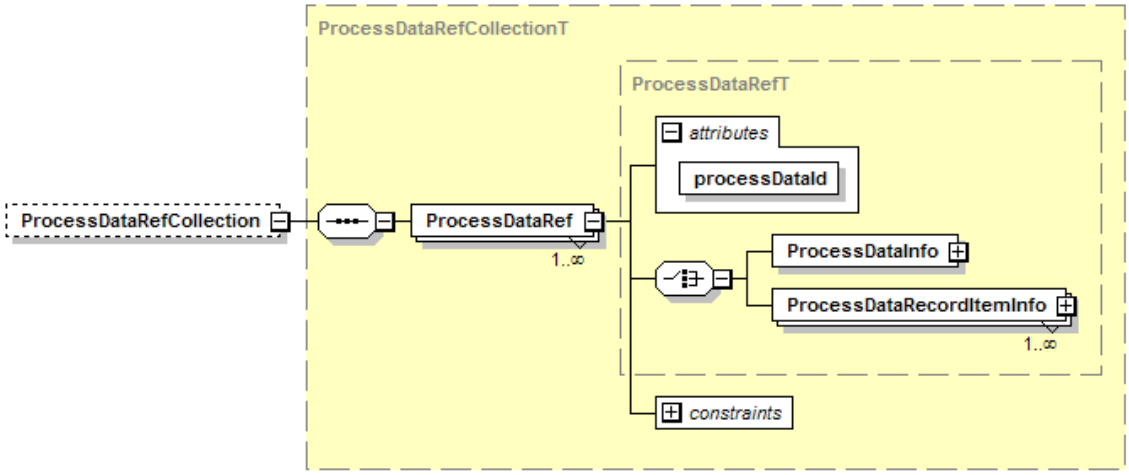


Figure 43 – ProcessDataRefCollection element

processDataId (m, RefT)
Refers to DeviceFunction/ProcessData/ProcessDataIn or ProcessDataOut.

ProcessDataInfo (c)
Corresponds to the 'VariableRef' element of a menu except for 'accessRightRestriction' and 'Button' which are not applicable (see 7.5.8.4).

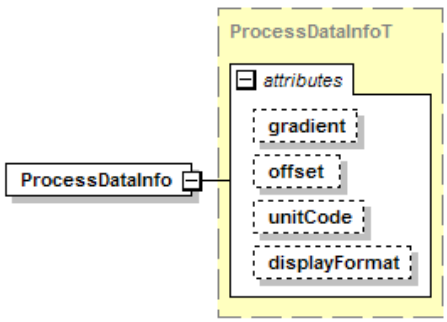


Figure 44 – ProcessDataInfo element

Use this for process data which is not of type record.

ProcessDataRecordItemInfo (c)
Corresponds to the ‘RecordItemRef’ element of a menu except for ‘accessRightRestriction’ and ‘Button’ which are not applicable (see 7.5.8.5).

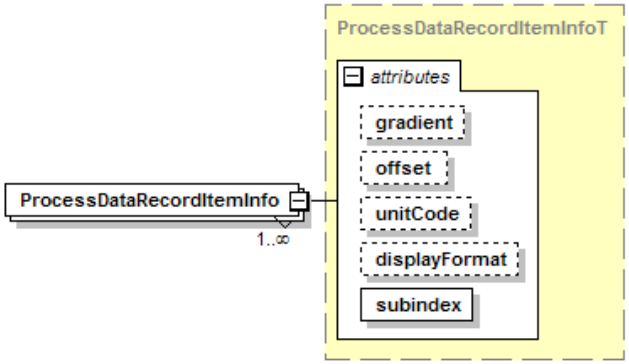


Figure 45 – ProcessDataRecordItemInfo element

Use this for process data which is of type record.

7.5.8.2 MenuSets (m)

User Roles
A UserInterface must always be divided into three user roles. It is up to the vendor how the roles are organized. The IO-Link Tool must assign the entered UserLevel to the respective menu. At most three menu levels below the role assignment are acceptable.

Example:

ObservationRoleMenuSet
->IdentificationMenu
-> Menu1
-> MenuRef1
-> Menu2
-> MenuRef1
MaintenanceRoleMenuSet
->ObservationMenu
-> MenuX
-> MenuRefY

IO-Link Tools shall upload or download only the variables of the current user role. If the tool supports a special function to replicate an IO-Link device, this function shall use the variables of the specialist role and can be available in all user roles.

ObserverRoleMenuSet (m)
This menu is designed for users who may not carry out any modifications on the device.

MaintenanceRoleMenuSet (m)
This menu is designed for observers and to undertake “uncritical” editing. It is up to the vendor to assess that.

SpecialistRoleMenuSet (m)
If the user is logged in as a specialist, he/she has total access to the device. Again, the vendor can decide which parameters may be edited.

For each role, there is a set of fixed top-level menus given.

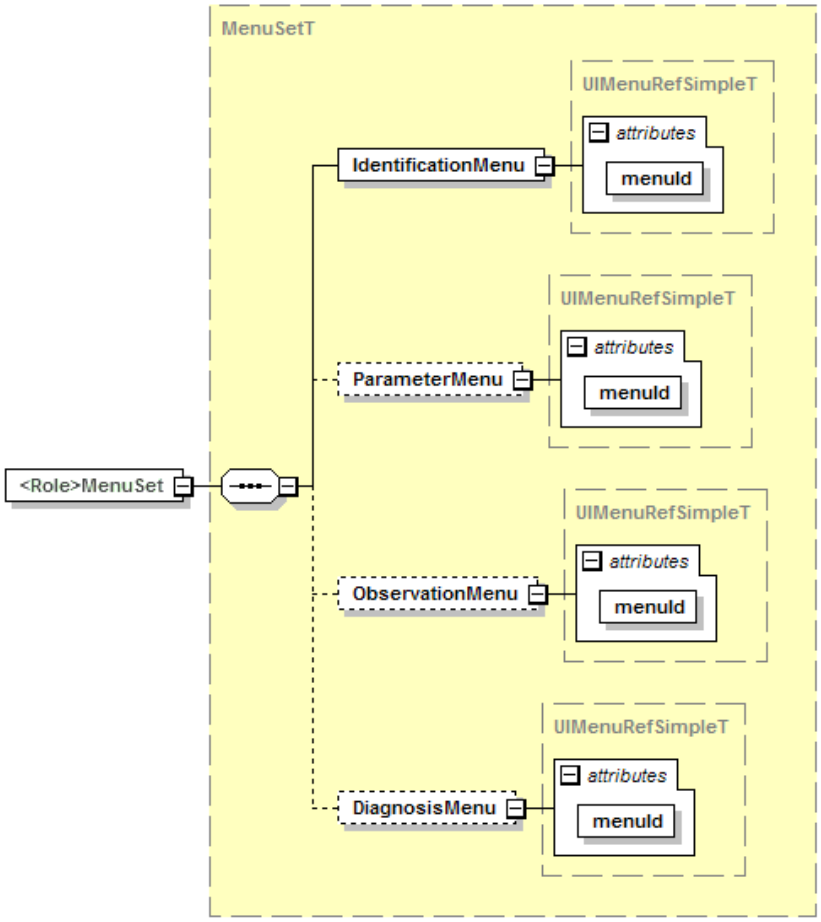


Figure 46 – <Role>MenuSet element

IdentificationMenu (m)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the identification of the device.

ParameterMenu (o)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the parameterization of the device.

ObservationMenu (o)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the observation of the device (process data, dynamic variables, etc.).

DiagnosisMenu (o)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the diagnosis of the device (events, etc.).

7.5.8.3 Menu collection

The names of top level menus, like IdentificationMenu, ParameterMenu, ObservationMenu or DiagnosisMenu are given from tooling. If a name is specified, it shall be ignored by tooling.

In underlying menus, a menu name shall be given by IODD.

MenuCollection (m)
All menu entries of the device are collected in the MenuCollection. These menu entries may be referenced by different roles (ObserverRole, MaintenanceRole, and SpecialistRole). There shall be no unreferenced Menu elements.

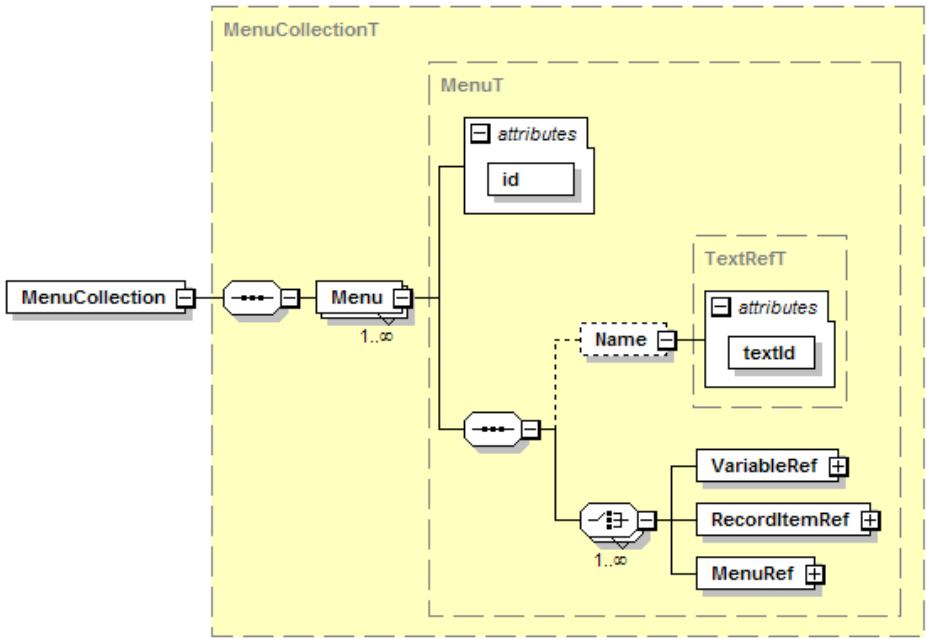


Figure 47 – MenuCollection element

Menu (m)
Variables, RecordItems and other menus may be referenced here.

id (m, IdT)
Explicit id of the menu.

Name (c)
textId (m, RefT)
Name of the menu. Top-level menus (i.e. those referenced from one of the MenuSets) may have a Name element, but it shall be ignored by IO-Link Tools. Instead, hard-coded names shall be used by the tools. Nested menus shall have a Name element which is shown by the tools.

7.5.8.4 VariableRef

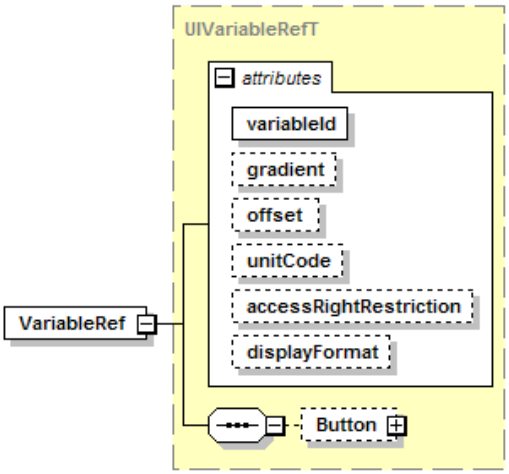


Figure 48 – VariableRef element

variableId (m, RefT)
Referenced variable

Regardless of the type of the referenced Variable or RecordItem, if gradient and / or offset are given, they shall be specified as floating point values.

Displayed value = (value read from the Device x gradient) + offset

When applying gradient and / or offset to convert the Variable or RecordItem value to the displayed value, the value will be implicitely converted to a floating point value. Consequently, the only allowed displayFormat on such values shall be “Dec”. (The displayFormat “Hex”, “Bin”, … does not force a conversion back to the original type of the Variable or RecordItem.)

When applying gradient and / or offset to convert an entered value back to the new value of a Variable or RecordItem, the resulting floating point value will be rounded to the nearest possible value of the type of the Variable or RecordItem.

Single array members can’t be referenced with RecordItemRef. If you need to access a single member, you have to define a record instead of an array.

A variable of type array can only be referenced as a whole, i.e. with VariableRef. All the elements and attributes in VariableRef (gradient, offset, unitCode, accessRightRestriction, displayFormat and Button) apply to each of the array members.

gradient (o, decimal)
Gradient of the indicated variables. The value shall not be zero. When offset is specified and gradient is not specified, a value of 1.0 shall be used.

offset (o, decimal)
Zero-offset of the indicated variables. When gradient is specified and offset is not specified, a value of 0.0 shall be used.

unitCode (o, positiveInteger)
Unit code to which the indicated variable refers. For valid unit codes see IODD-StandardUnitDefinitions1.1.xml.

unitCode shall only be used with datatypes UIntegerT, IntegerT and Float32T.

unitCode shall only be used without displayFormat or with displayFormat Dec and Dec.x.

accessRightRestriction (o, AccessRightsT)
For certain UserRoles, the access rights may be limited here.

displayFormat (o, string with pattern)
Specifies how an IO-Link Tool shall display the value in the menu. The values of the attribute ‘displayFormat’ shall follow the regular expression pattern:

“Bin|Hex|Dec(\.\d)?”

Meaning of the values:

Bin:	Binary notation with postfix “b”, e.g. 0101 1010 1010 0101b
Hex:	Hexadecimal notation with postfix “h”, e.g. 5AA5h
Dec:	Decimal notation without postfix, e.g. 23205
Dec.2:	Decimal notation with given precision (number of digits after the decimal point) e.g. 23.00

The following table shows the valid combinations of the data type of the referenced Variable / RecordItem and the displayFormat, gradient and offset. Combinations not listed here shall not be used.

Table 4 – Allowed combinations of datatype, displayFormat, gradient and offset

datatype	allowed displayFormat	gradient and/or offset allowed	IO-Link Tool behaviour
BooleanT	Dec	No	Display as “0” for false and “1” for true.
	default	No	Display as “false” or “true”.
UIntegerT	Bin	No	Display as e.g. “0101 1010 1010 0101b”. Show 8, 16, 32 or 64 binary digits.
	Hex	No	Display as e.g. “5AA5h”. Show 2, 4, 8 or 16 hexadecimal digits.
	Dec	Yes	Without gradient and/or offset: Display as e.g. “23205”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec
	Dec.x	Yes	Without gradient and/or offset: Display as e.g. “23205.00”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec.x
	default	Yes	Same as Dec.
IntegerT	Bin	No	Display as e.g. “1111 1011 0010 1110b”. Show 8, 16, 32 or 64 binary digits. Show negative values as two’s complement.
	Hex	No	Display as e.g. “FB2Eh”. Show 2, 4, 8 or 16 hexadecimal digits. Show negative values as two’s complement.
	Dec	Yes	Without gradient and/or offset: Display as e.g. “-1234”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec
	Dec.x	Yes	Without gradient and/or offset: Display as e.g. “-1234.00”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec.x
	default	Yes	Same as Dec.

Float32T	Dec	Yes	Display digits after the decimal point as needed (up to an implementation-defined maximum).
	Dec.x	Yes	Display exactly “x” digits after the decimal point (also in exponential representation). Rounding shall be done with midpoint rounding away-from-zero (e.g. with “Dec.3” 23.3455 gets rounded to 23.346, and −23.3455 gets rounded to −23.346).
	default	Yes	Same as Dec.
StringT	*default*	No	Display just the string.
OctetStringT	*default*	No	Display as e.g. 0x00,0x56,0x78.
TimeT	*default*	No	Display as yyyy-mm-dd hh:mm:ss.fff where yyyy is the year, mm is the month, dd is the day, hh is the hour, mm is the minute, ss is the second and fff is the milliseconds.
TimeSpanT	*default*	No	Display as [+-][d]hh:mm:ss.fff where d is the days (optional, one or more digits), hh is the hour, mm is the minute, ss is the second and fff is the milliseconds.
ArrayT	Display all array elements. Button is not allowed. The allowed displayFormat, gradient/offset and unitCode is determined by the data type of the array elements.		
RecordT	*default*	No	Display all RecordItems in the order in which they appear in the Record definition, i.e. with ascending subindices, with their default display format. Button and / or unitCode are not allowed.

Some standard variables have complex types which are not modelled as special types in IODD because:

- The types cannot be used with other variables in a meaningful way.
- The types are difficult to describe in XML, increasing the complexity of the IODD.
- The types are IO-Link specific, complicating the general use of the IODD.

These standard variables could be displayed as bare numbers the way they are described in IODD-StandardDefinitions1.1.xml, but it is recommended that IO-Link Tools identify them by their name or index and display them specially, as described in the following table:

Table 5 – Standard variables with special display

Standard Variable / RecordItem	Special display
V_DirectParameters_1 (index 1), MasterCycleTime (subindex 2)	The octet consists of a Time Base in bits 7 to 6 and a Multiplier in bits 5 to 0. The IO-Link Tool shall calculate the time according chapter B.1.4 of the IO-Link Communication Specification Version 1.1 and display it as a decimal number with the unit milliseconds (ms).
V_DirectParameters_1 (index 1), MinCycleTime (subindex 3)	The octet consists of a Time Base in bits 7 to 6 and a Multiplier in bits 5 to 0. The IO-Link Tool shall calculate the time according chapter B.1.4 of the IO-Link Communication Specification Version 1.1 and display it as a decimal number with the unit milliseconds (ms).
V_DirectParameters_1 (index 1), M-sequence Capability (subindex 4)	The octet consists of a PREOPERATE M-sequence type in bits 5 to 4, an OPERATE M-sequence type in bits 3 to 1, and ISDU in bit 0. The IO-Link Tool shall decode this into text according to chapter B.1.5 of the IO-Link Communication Specification Version 1.1.
V_DirectParameters_1 (index 1), RevisionID (subindex 5)	The octet consists of a MajorRev in bits 7 to 4 and a MinorRev in bits 3 to 0. It shall be displayed as Vx.y, where x is the MajorRev and y is the MinorRev.
V_DirectParameters_1 (index 1), ProcessDataIn (subindex 6)	The octet consists of BYTE in bit 7, SIO in bit 6 and Length in bits 4 to 0. The IO-Link Tool shall display whether SIO is supported and the length in bits / octets according chapter B.1.7 of the IO-Link Communication Specification Version 1.1.
V_DirectParameters_1 (index 1), ProcessDataOut (subindex 7)	The octet consists of BYTE in bit 7, SIO in bit 6 and Length in bits 4 to 0. The IO-Link Tool shall display whether SIO is supported and the length in bits / octets according chapter B.1.8 of the IO-Link Communication Specification Version 1.1.
V_DetailedDeviceStatus (index 37)	Each array element shall be treated as an event data structure (see IO-Link Communication Specification Version 1.1, chapter A.6, and Figure 49). It shall be decoded and displayed to text using the EventCollection in the IODD.
V_OffsetTime (index 48)	The octet consists of a Time Base in bits 7 to 6 and a Multiplier in bits 5 to 0. The IO-Link Tool shall calculate the time according chapter B.2.20 of the IO-Link Communication Specification Version 1.1 and display it as a decimal number with the unit milliseconds (ms).

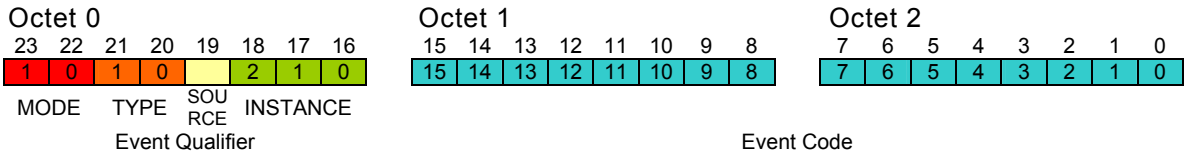


Figure 49 – Event data structure

Button (o)
Buttons are intended for implementing a command interface to the device. Several commands can be implemented on the same variable / record item using different values to be written.

If this element is given, the IO-Link Tool shall display a button instead of a value. The attributes ‘gradient’, ‘offset’, ‘unitCode’ and ‘displayFormat’ shall not be used when the element ‘Button’ is present.

The button shall be labelled with the Name that is given to the SingleValue at the data type of the referenced Variable or RecordItem whose ‘value’ corresponds to the ‘buttonValue’.

Use of this element is restricted to the data types BooleanT, UIntegerT and IntegerT. Note that this does not include arrays of these data types.

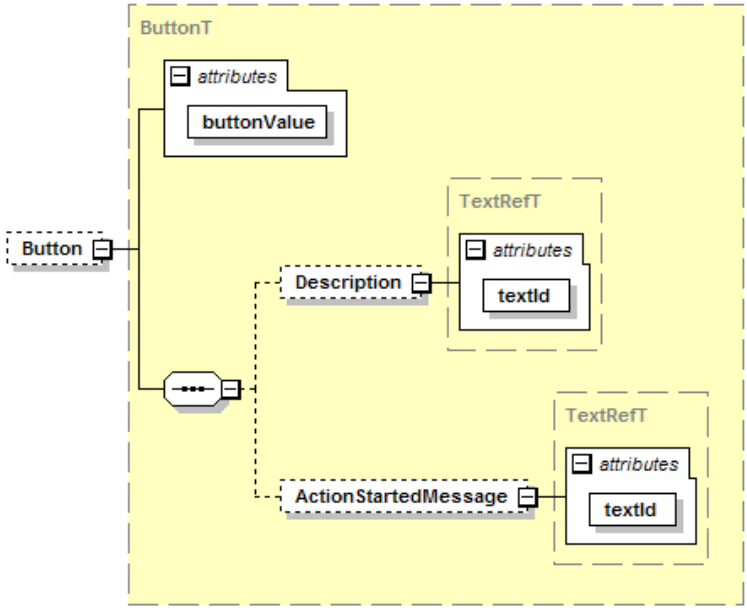


Figure 50 – Button element

A variable referenced as “Button”
shall have accessRights “wo”
shall only be displayed as a button
shall not be used as a condition variable, to switch menus or process data.

The buttonValue
will be sent to the device immediately by pushing the button.
shall not be part of the block-download sequence.

buttonValue (m, union of boolean, unsignedLong and long)
This value must correspond to a SingleValue/@value of the data type of the referenced Variable or RecordItem. It is sent to the device when the button is clicked.

Description (o)
textId (m, RefT)
A text that explains the action that will be started by pressing the button.

ActionStartedMessage (o)
textId (m, RefT)
A text that is shown after the button value was successfully sent to the device. Use this as a feedback to the user for actions that may take a while to complete or that require some user action to complete.

7.5.8.5 RecordItemRef

Corresponds to VariableRef with an additional subindex. The variable referenced by variableId shall be of type record. If 'Button' is specified, the referenced variable shall support subindex access.

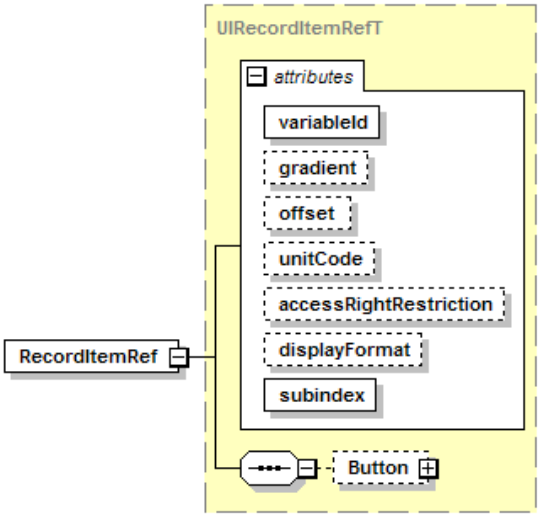


Figure 51 – RecordItemRef element

subindex (m, SubindexT)
Addresses the record item of a variable of type record.

For the other attributes and the element 'Button', see VariableRef above.

7.5.8.6 MenuRef

Reference to a (sub)menu nested inside this menu.

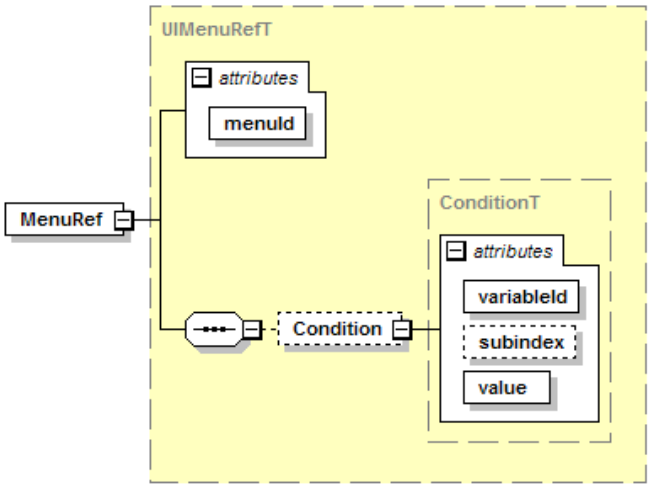


Figure 52 – MenuRef element

menuId (m, RefT)
References the (sub)menu from the MenuCollection.

Condition (o)
Condition for the display of this menu; an IO-Link Tool shall show the referenced menu only if the value of the referenced variable / record item equals the value of the attribute ‘value’.

variableId (m, RefT)
References a variable. The variable shall be of data type BooleanT, IntegerT, UIntegerT or RecordT. The variable shall have a default value if it is not of type RecordT.

subindex (c, SubindexT)
This attribute shall be given if and only if the referenced variable is of type RecordT. Used for addressing the record item within the record. The record item shall be of data type BooleanT, IntegerT or UIntegerT and shall have a default value.

value (m, unsignedByte)
Must be a valid value for the variable / record item. This attribute can only hold values 0..255, thus limiting the possible IntegerT and UIntegerT values. Also, BooleanT condition values shall be entered as 0 for false and 1 for true.

Conditions may be used in all menu levels.

Condition variables shall appear as VariableRef or RecordItemRef at least in a read-only way in a menu which is referenced in the same user role.

If there is more than one ProcessData element, selected by conditions, and the variable V_ProcessDataInput or V_ProcessDataOutput is referenced in a menu, one of the following shall hold:

- The type of reference (VariableRef / RecordItemRef) and the gradient, offset, unitCode and displayFormat fit to each of the ProcessData elements.
- The menu is conditioned in the same way as one of the ProcessData elements, and the type of reference (VariableRef / RecordItemRef) and the gradient, offset, unitCode and displayFormat fit to this particular ProcessData element.

“Conditioned in the same way” means that this or one of the parent menus has the same condition (same variable, same subindex, same value).

7.6 Communication characteristics

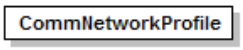


Figure 53 – CommNetworkProfile element

Excursion on XML schema *abstract types*:
An abstract type can't be used itself. Only non-abstract types which are derived from an abstract type can be used. The instance selects the desired derived type with xsi:type="*name of the derived type*".

This technique is used here with the 'CommNetworkProfile' element to adapt the XML structure to the requirements of the specific communication. This allows easy extension of the IODD to non-IO-Link devices with different communication characteristics as long as the applicative concept remains the same (i.e. addressing via index/subindex, standardized variables).

For IO-Link, the following derived type IOLinkCommNetworkProfileT describes the communication characteristics of an IO-Link interface.

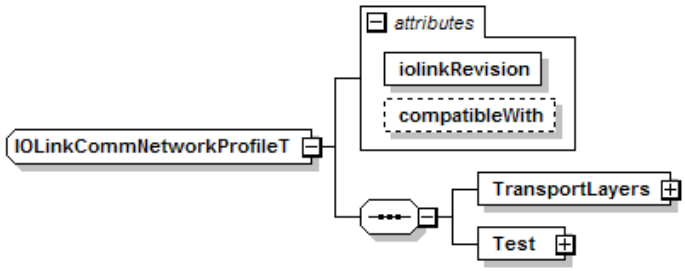


Figure 54 – CommNetworkProfile element – IO-Link variant

iolinkRevision (m, VersionT)
Implemented protocol version. Fixed to "V1.1".

compatibleWith (o, VersionT restricted to "V1.0")
Specify this attribute if the device is compatible with IO-Link revision 1.0, i.e. also runs on a V1.0 IO-Link master. This requires an IODD V1.0.1 (or V1.0).

TransportLayers (m)

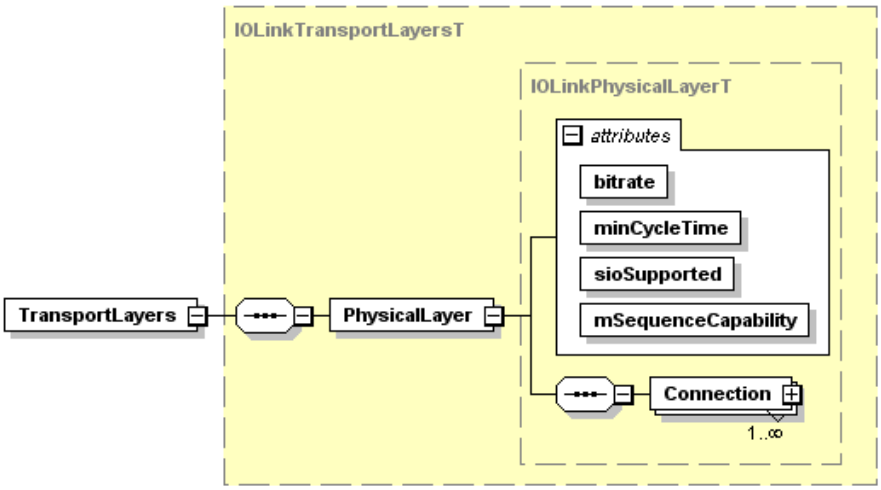


Figure 55 – TransportLayers element – IO-Link variant

PhysicalLayer (m)

bitrate (m, string)
Allowed values are “COM1”, “COM2” or “COM3”.

minCycleTime (m, unsignedInt)
The minimum cycle time of the slave; specified in 1 microsecond (µs) units. E.g. the value 2300 represents 2.3 milliseconds (ms). The allowed value range is 0..6300 in steps of 100, 6400..31600 in steps of 400 and 32000..132800 in steps of 1600 (see IO-Link Communication Specification Version 1.1, chapter B.1.4).

sioSupported (m, boolean)
Whether the fall-back to SIO mode is supported.

mSequenceCapability (m, unsignedByte)
Enter the content of V_DirectParameters_1, subindex 4 (M-sequence Capability) here as a decimal number.

Test (m)
Contains information to enable automatic testing of the device.

Enter appropriate data for the ISDU and event test configurations (see IO-Link Test Specification Version 1.1, chapter 6.5.1).

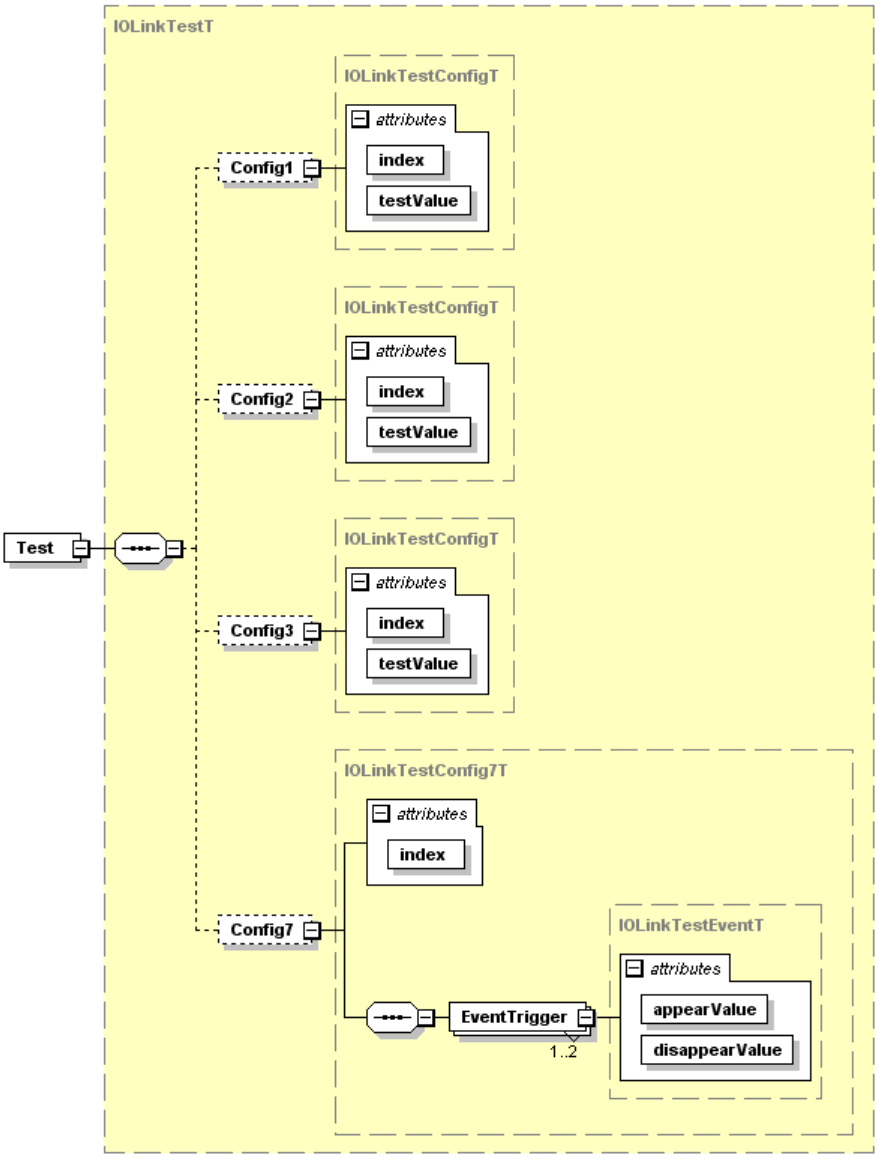


Figure 56 – Test element

Config1 (o)
Shall be present if the device supports ISDU access, and refers to a read-write 8 bit index. The testValue shall be small enough (<=12 octets) so that the ExtLength coding of the ISDU is not used.

Config2 (o)
Shall be present if the device supports ISDU access, and refers to a read-write 16 bit index. If the device supports a read-write variable on such an index, the testValue shall be accepted by the device. If the device does not support such an index a valid ErrorType shall be returned (Index not available).

Config3 (o)
Shall be present if the device supports ISDU access, and refers to a read-write 8 bit index. The testValue shall have a length big enough (>12 octets) to cause the ExtLength coding of the ISDU.

Config7 (o)
Shall refer to an index raising different types of events. For details, please refer to the IO-Link Test Specification.

index (m, RefT)
References an Index to be used for testing.

testValue (m, string with pattern: "(0x[0-9A-Fa-f][0-9A-Fa-f],)*0x[0-9A-Fa-f][0-9A-Fa-f]")
Must be an acceptable octet string value for the index.

EventTrigger (m)
Trigger values for up to two events.

appearValue (m, unsignedByte)
The value that triggers an event when written to Config7/@index.

disappearValue (m, unsignedByte)
The value that quenches the event triggered by @appearValue when written to Config7/@index.

Connection (m)
Describes, how the device can be connected. For each device variant a separate Connection shall be defined.

This element has the following XML abstract type:

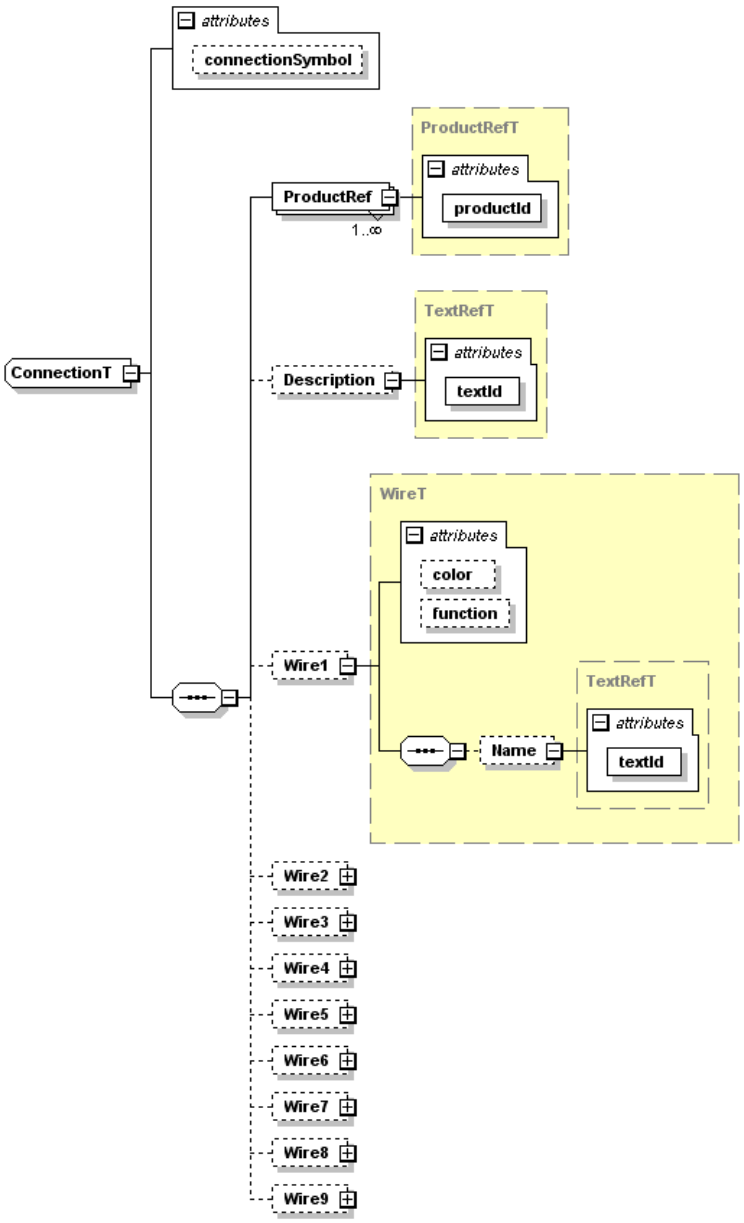


Figure 57 – ConnectionT abstract type

connectionSymbol (o, string with pattern "([\p{L}\d_#]+-)+con-pic\.png")
File name of the connection symbol. If this attribute is used, the referenced image file shall be present.

ProductRef (m)
productId (m, string)
Selects the device variants that use this Connection. Must correspond to one of the DeviceIdentity/DeviceVariantCollection/DeviceVariant/@productId values.

Description (o)
textId (m, RefT)
Description of the connection.

Wire<X>
Describes one of the wires. If the connection is some type of connector, the number <X> also designates the pin / hole number.

For OtherConnectionT and CableConnectionT, if the number of wires exceeds 9, the wires relevant for IO-Link shall be described. In this case, the <X> does not necessarily equal the pin / hole number. The real pin number should be described in the Wire<X>/Name element.

color (o, string)
A color code according to IEC 60757:1983.

Table 6 – Wire colors

Code	Color
“BK”	Black
“BN”	Brown
“RD”	Red
“OG”	Orange
“YE”	Yellow
“GN”	Green
“BU”	Blue (including light blue)
“VT”	Violet (purple)
“GY”	Grey (slate)
“WH”	White
“PK”	Pink
“GD”	Gold
“TQ”	Turquoise
“SR”	Silver

function (o, string)
The function of the wire.

Table 7 – Wire functions

Function	Description
“NC”	Not connected
“L+”	Power supply (+), pin 1, brown
“L-”	Power supply (-), pin 3, blue
“P24”	Extra power supply (+)
“N24”	Extra power supply (-)
“Other”	e.g. signal (DI, DO, analog) or power supply
“C/Q”	Communication signal, pin 4, black

Name (o)
textId (m, RefT)
An additional textual description of the wire. Do not repeat the wire color or wire function in textual form here.

The allowed XML derived types are: M5ConnectionT, M8ConnectionT, M12-4ConnectionT, M12-5ConnectionT, OtherConnectionT and CableConnectionT. They restrict the abstract type accordingly.

M5ConnectionT
The M5 connector as specified in IO-Link Communication Specification Version 1.1, chapter 5.5.

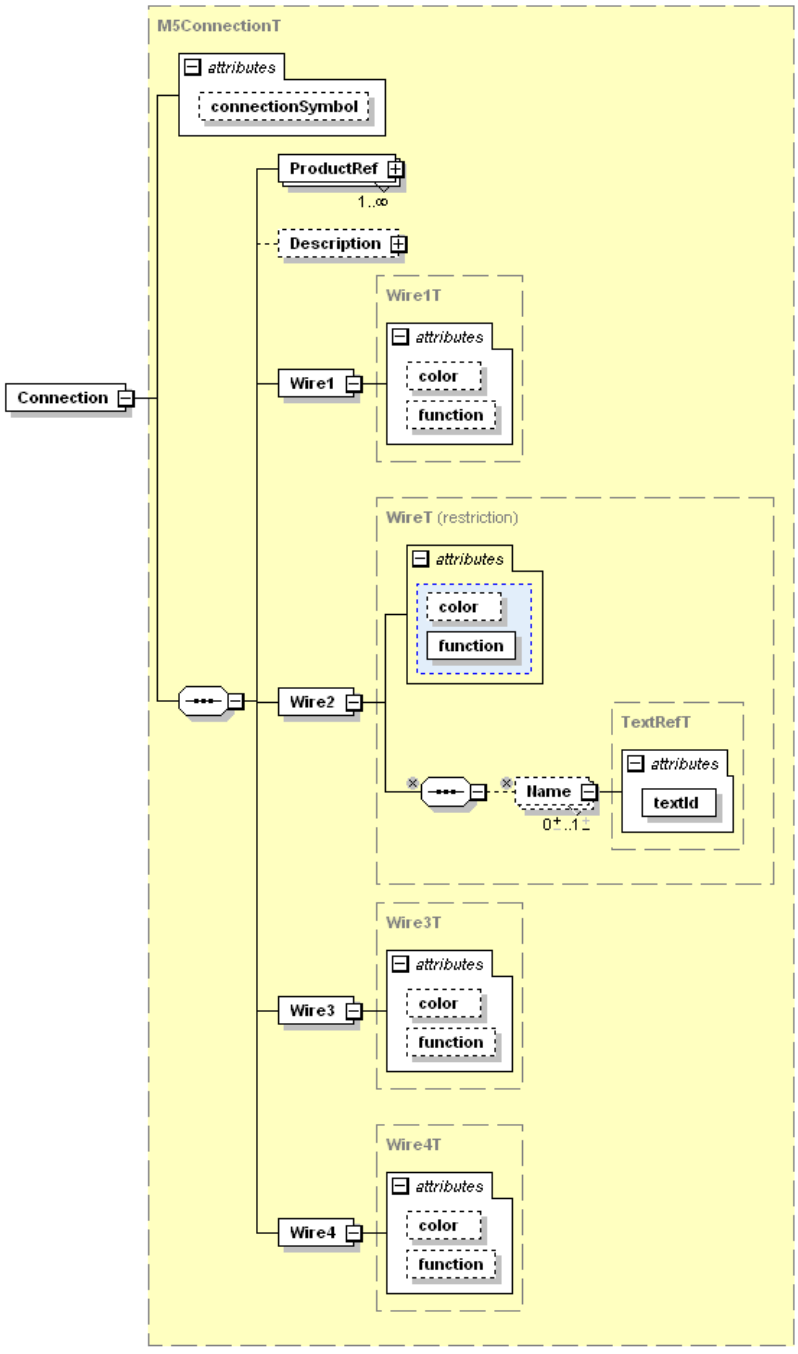


Figure 58 – Connection element – M5ConnectionT variant

Wire1, Wire3 and Wire4 have fixed color and function. No Name is allowed.

Wire 2 has a fixed color “WH” (white), and a function restricted to “NC” or “Other”. The function attribute is mandatory.

M8ConnectionT and M12-4ConnectionT
Same as M5ConnectionT.

M12-5ConnectionT
The M12-5 connector as specified in IO-Link Communication Specification Version 1.1, chapter 5.5.

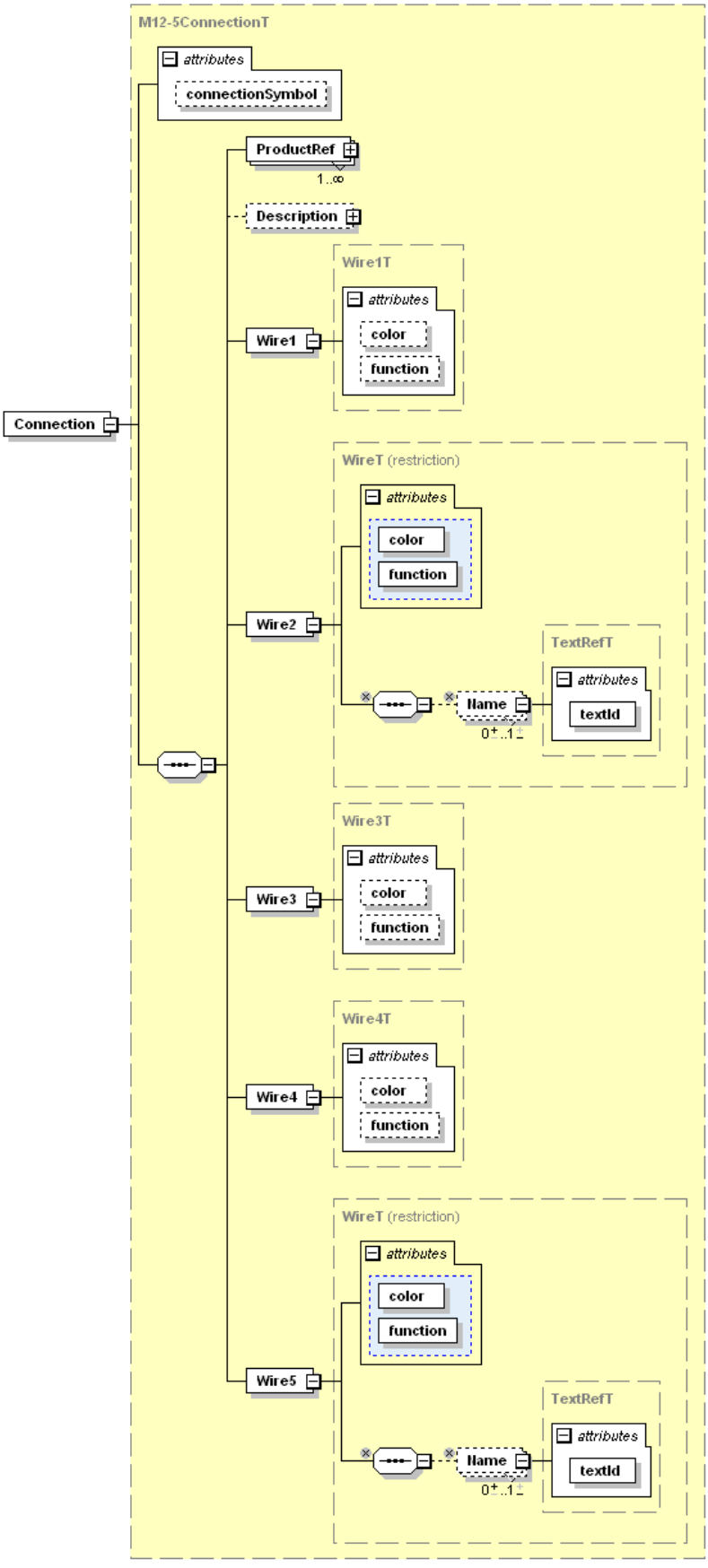


Figure 59 – Connection element – M12-5ConnectionT variant

Wire1, Wire3 and Wire4 have fixed color and function. No Name is allowed.

Wire 2 has a function restricted to “NC”, “P24” or “Other”. If its function is I/Q, its color shall be “WH” (white). The color and function attributes are mandatory.

Wire 5 has a function restricted to “NC” or “N24”. The color and function attributes are mandatory.

OtherConnectionT
Some non-standard connector.

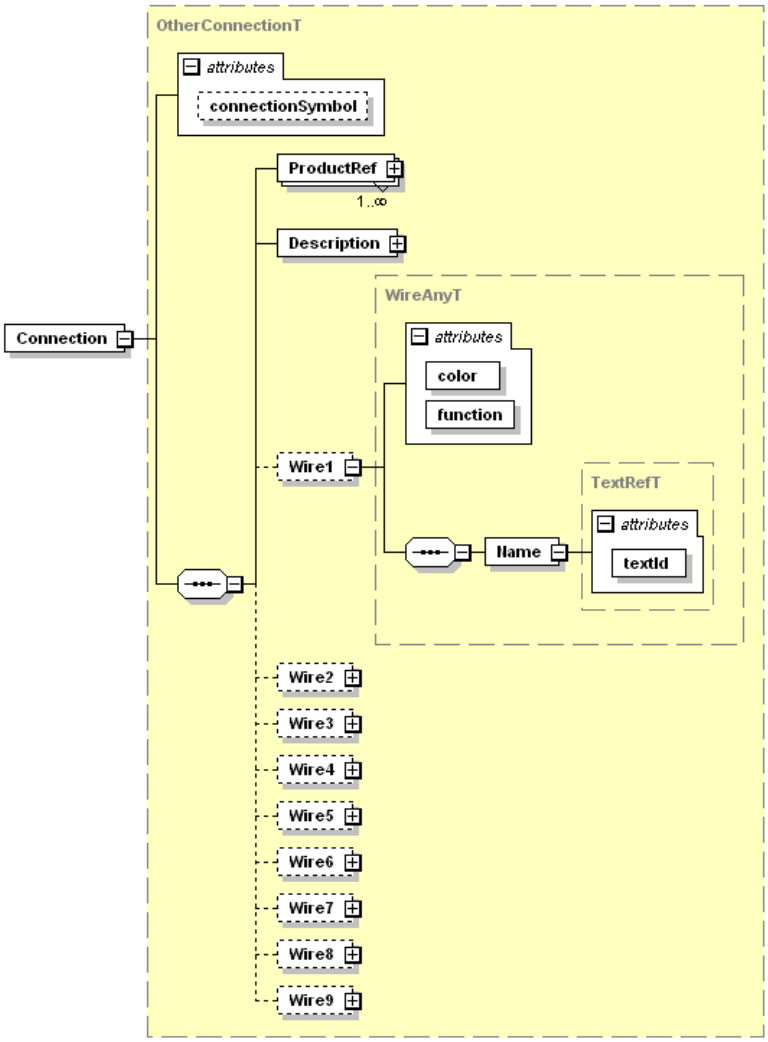


Figure 60 – Connection element – OtherConnectionT variant

The Description is mandatory. For all given Wires, the element ‘Name’ and the attributes ‘color’ and ‘function’ are mandatory.

Wires with functions “L+”, “L-“ and “C/Q” shall be present.

CableConnectionT
Same as OtherConnectionT, but Description is not mandatory. The wire number in the ‘Wire…’ element name does not designate a pin number here, but any arbitrary numbering of the wires.

7.7 Language dependent description texts

All text components of the different languages are given in the ExternalTextCollection. There may be one or more languages deposited. Additional languages may be stored in separate files.

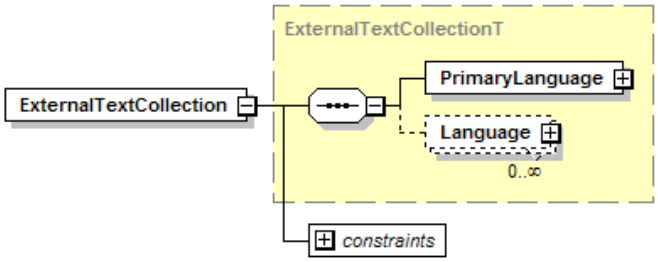


Figure 61 – ExternalTextCollection element

7.7.1 PrimaryLanguage (m)

Shall be in English.

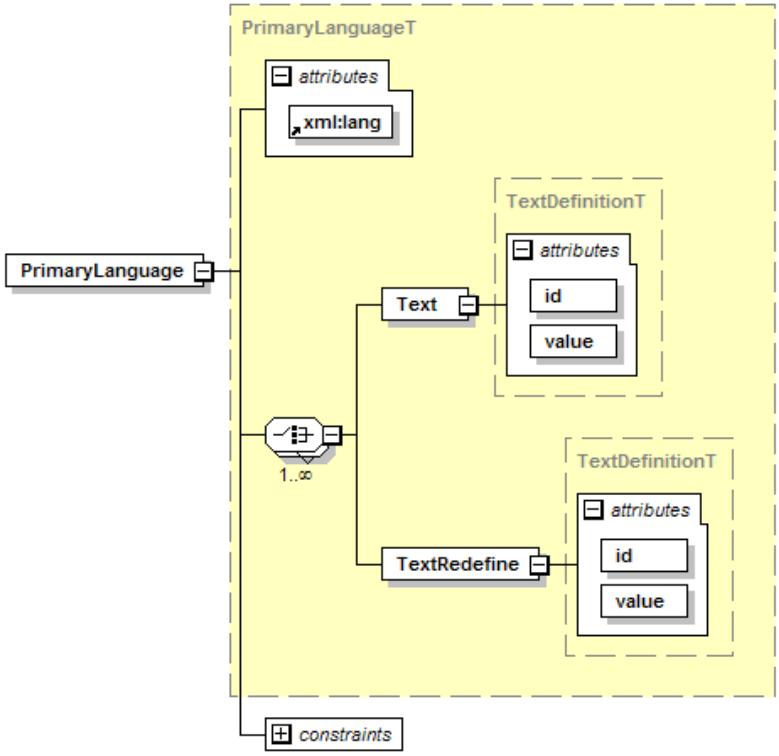


Figure 62 – PrimaryLanguage element

xml:lang (m, language)
The code for the language according to ISO 639-1:2002. Shall be “en” for English.

Text (m)
Language dependent text which is referenced by its id.

id (m, IdT)
Shall be referenced by other elements via their ‘textId’ attribute (there shall be no unreferenced Text elements)

value (m, string)
Text in the denoted language.

NOTE: Special characters shall be coded according to the XML syntax. See REC-xml-20081126, chapter 2.4 Character Data and Markup.

& → &

' → ' (only required if inside a string enclosed in ' characters)

> → >

< → <

" → " (only required if inside a string enclosed in " characters)

LF →

Only the line feed is allowed for formatting the text.

TextRedefine (o)
Language dependent text which overrides a standard text. Only applicable for texts describing the octets of DirectParameter page 2.

id (m, IdT)
Shall be one of STD_TN_DeviceSpecific_1 to STD_TN_DeviceSpecific_16.

value (m, string)
Text in the denoted language. The NOTE at Text/@value also applies.

7.7.2 Language (o)

Optional specification of texts in another language. The attribute ‘xml:lang’ specifies the language (see ISO 639-1:2002). The structure of this element corresponds to the structure of the element PrimaryLanguage.

8 Compatibility

IO-Link devices conforming to the IO-Link Communication Specification Version 1.1 shall be described with an IODD according to this IO Device Description Specification Version 1.1.

IO-Link devices conforming to the IO-Link Communication Specification Version 1.0 shall be described with an IODD according to the IO Device Description Specification Version 1.0.1 (or the legacy version 1.0).

It is not possible to describe a V1.0 device with an IODD V1.1 or a V1.1 device with an IODD V1.0.1 or V1.0.

Two IODDs having the same vendorId and deviceId, one being based on V1.0.1 (or V1.0) and the other being based on V1.1 are only allowed in the following use cases:

- A V1.0 device exists and has a V1.0.1 (or V1.0) IODD. A new, compatible version of the device is built using the same vendorId and deviceId based on V1.1 needing a V1.1 IODD.

- A V1.1 device is newly built with a new vendorId /deviceId pair. It is also V1.0 compatible and thus needs two IODDs.

In both cases, the DeviceFamily and all DeviceVariant/Name (previously ProductName) for all languages, as well as all DeviceVariant/@productId must be same.

An IO-Link Tool shows the Device / the DeviceVariants only once in the catalog. Depending on the master it will use the IODD V1.1 or V1.0.1 (or V1.0).

Annex A IODD schemas

The following schemas and standard definition files are part of this specification:

Schema files

- IODD1.1.xsd — main IODD schema
- IODD-Primitives1.1.xsd — basic definitions
- IODD-Datatypes1.1.xsd — data types
- IODD-Events1.1.xsd — events
- IODD-Variables1.1.xsd — variables
- IODD-UserInterface1.1.xsd — user interface
- IODD-Communication1.1.xsd — communication network profile
- IODD-StandardDefinitions1.1.xsd — main schema for the standard definition files

Standard definition files

- IODD-StandardDefinitions1.1.xml — list of standard variables, error types and events + english texts
- IODD-StandardDefinitions1.1-de.xml — german texts
- IODD-StandardUnitDefinitions1.1.xml — list of available unit codes + english texts
- IODD-StandardUnitDefinitions1.1-de.xml — german texts

Annex B Definitions of IODD quantity structure

The following table lists limits on the number and the length of elements of the IODD. IODDs shall not exceed these limits. IO-Link Tools shall accept all IODDs that do not exceed these limits.

Table 8 – IODD quantity structure

Element	Maximum Number/Length	Comments
Length of filename	255	
Number of DeviceVariants	255	
Number of ValueRanges	32	
Number of SingleValues	255	
Number of ProcessData Elements	8	
Number of Datatypes	255	
Number of Variables	1024	
Length of text which is referenced by Description/@textId in all languages	255	Characters, not octets
Length of URL which is referenced by VendorUrl/@textId in all languages	1024	Characters, not octets
Length of @vendorName and all other texts referenced by a @textId (e.g. VendorText, DeviceName, DeviceFamily, Name, …)	64	Characters, not octets
Length of DocumentInfo/@copyright text	255	Characters, not octets
Number of Menus in MenuCollection	255	
Number of elements per menu (VariableRef+RecordItemRef+MenuRef)	64	
Variable@id length	64	Characters, not octets
Datatype@id length	64	Characters, not octets
@textId length	64	Characters, not octets
Number of supported languages	see ISO 639-1	Currently 185
Number of Menu Levels	3	

IO-Link Consortium
Haid-und-Neu-Str. 7
76131 Karlsruhe
Germany
Phone: +49 (0) 721 / 96 58 590
Fax: +49 (0) 721 / 96 58 589
e-mail: info@io-link.com
http://www.io-link.com/