IODD

IO Device Description

Guideline

related to

IO-Link Interface and System Specification V1.1.2

and

IO Device Description Specification V1.1

Version 1.1
July 2013

Order No: 10.022

File name: IO-Device-Desc-Guideline_10022_V11_130715.doc

Prepared, approved and released by the IO-Link Community

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 Community 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. Easy access to the file and potential updates shall be possible. It is the responsibility of the IO-Link device manufacturer to test the IODD file with the help of the IODD-Checker tool available per download from www.io-link.com.

NOTE 3 Any IO-Link device shall provide an associated manufacturer declaration on the conformity of the device with the IO-Link Communication Specification and its related IODD, and test documents, available per download from www.io-link.com.

Disclaimer:

The attention of adopters is directed to the possibility that compliance with or adoption of IO-Link Community specifications may require use of an invention covered by patent rights. The IO-Link Community shall not be responsible for identifying patents for which a license may be required by any IO-Link Community specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. IO-Link Community specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

The information contained in this document is subject to change without notice. The material in this document details an IO-Link Community specification in accordance with the license and notices set forth on this page. This document does not represent a commitment to implement any portion of this specification in any company's products.

WHILE THE INFORMATION IN THIS PUBLICATION IS BELIEVED TO BE ACCURATE, THE IO-LINK COMMUNITY MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR PARTICULAR PURPOSE OR USE.

In no event shall the IO-Link Community be liable for errors contained herein or for indirect, incidental, special, consequential, reliance or cover damages, including loss of profits, revenue, data or use, incurred by any user or any third party. Compliance with this specification does not absolve manufacturers of IO-Link equipment, from the requirements of safety and regulatory agencies (TÜV, BIA, UL, CSA, etc.).

IO-Link ® is registered trade mark. The use is restricted for members of the IO-Link Community. More detailed terms for the use can be found in the IO-Link Community Rules on www.io-link.com.

Conventions:

In this guideline 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 the IO-Link specifications.

Publisher:
IO-Link Community
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	WG Technology	05-Mar-2013	Draft for IO-Link Review
1.1	WG Technology	07-May-2013	Incorporated review comments

1 Introduction

This IODD Guideline is intended for the IO-Link developer. It helps to

- design the device so that it can be effectively described by the IODD
- write a standards conformant IODD for the device
- release the IODD in a way that is convenient for the user
- design and test IO-Link Tools

It contains explanations of concepts, best practice examples and advices. The knowledge of the IO-Link Interface and System Specification and the IO Device Description Specification is indispensable – this guideline does neither replace nor amend these specifications.

2 Related documents

IO-Link Interface and System Specification Draft Version 1.1.2, November 2012, Order No: 10.002

IO Device Description Specification Version 1.1, August 2011, Order No: 10.012

IO-Link Smart Sensor Profile Specification Version 1.0, October 2011, Order No: 10.042

3 Things to consider when designing an IO-Link Device

3.1 Assigning Device ID / Product ID

When creating a device that is similar to an existing device it must be decided whether the new device is just a new DeviceVariant of the existing device (same Device ID, different Product ID) or a new device (different Device ID).

The prerequisite of treating the new device as a DeviceVariant of an existing device is that the device only differs in things which are not "seen" by the IO-Link Tool (see chapter 7.4.1 of the IO Device Description Specification).

But even if this is the case, it is not always wise to do so. Imagine a device that comes in either stainless steel or plastics body. For e.g. food and drug processing, the two variants might not be interchangeable because the material is mandated. In this case it may be better to choose different Device IDs for the two materials so that an automation solution is able to reject an IO-Link Device with the wrong body material.

3.2 Floating point values

The IODD supports the data type Float32T. When you write a floating point value into the IODD, e.g. as @defaultValue or SingleValue/@value or ValueRange/@lowerValue or @upperValue, don’t expect a specific bit pattern in your device!

You should only expect a value that is very close to the value in the IODD. There are several reasons for this:

- There are separate bit patterns for 0 and -0. Tools usually accept both but write only the pattern for 0.
- Tools usually process the values internally in double (64 bit) precision and convert to float (32 bit) just before writing, leading to rounding.
- This effect is increased when @gradient and/or @offset is used.

[Interesting reading on this topic:
David Monniaux. The pitfalls of verifying floating-point computations, *TOPLAS*, 30(3):12, May 2008, http://hal.archives-ouvertes.fr/docs/00/28/14/29/PDF/floating-point-article.pdf.]

3.3 Process Data

It is possible to design IO-Link devices whose process data has several different layouts. The only constraint being that all layouts must have the same size.

These different layouts are usually the consequence of different modes that the Device is in. Also, the menus will usually be adapted according to that mode.

For each direction of the process data (in / out) there shall be only a single mode parameter (Variable or RecordItem) controlling the layout of the process data. The menus appropriate for the modes shall be switched by exactly the same mode parameter. There may be additional parameters switching menus, but these are not allowed to influence the process data layout.

It is possible to specify 'subindexAccessSupported' on the data type of the process data. This seems superfluous, as the process data is always transferred completely over the process data channel. But it is not! If the Device supports Service PDU 40 (V_ProcessDataInput) or 41 (V_ProcessDataOutput), and the process data has a complex type, you have to support subindex access according to this attribute specified at the data type of the process data.

3.4 ISDU versus DirectParameterPage

Vendor specific parameters may be placed into the DirectParameterPage 2 (Index 1) and into ISDUs (Index >= 64). It is highly recommended not to use both in the same Device. Please use the DirectParameterPage 2 only if your Device does not support ISDUs at all.

The IODD allows using DirectParameterOverlay to define a record that represents an overlay on top of the DirectParameterPage 2. This is convenient, but please keep in mind that the communication of this record is byte-oriented by nature. That means that RecordItems which can't be mapped onto exactly one DirectParameter byte cannot be transferred consistently. A concurrent read / write access may trash the read value, and a concurrent write / write access may trash the written value. The Device must be robust enough to survive the resulting trashed values.

3.5 Conditions

Conditions are used for switching process data and menus. They should be used sparingly. Instead of one Device with a lot of modes, consider making more than one Device with different characteristics. Instead of a lot of parameters controlling detailed aspects of menus consider using less parameters using more states, even if some menu entries must be duplicated.

3.6 Records

The layout of records should not be higgledy-piggledy.

- Let your integers start on a byte boundary.
- After an integer that does not completely fill the last byte, don't start another parameter in this byte that does not fit into the remaining space.
- If you have few booleans, put them into remaining space after integers.
- If you have more booleans, group them together into one or few bytes.

3.7 Default values

The default values of parameters (Attribute @defaultValue) shall match those values that the device returns in the condition as delivered (or after the factory reset command was performed, if the device supports it).

With the system variable V_ProductID a default value is a bit tricky, because there is only one default value but there may be a multitude of DeviceVariants each with its own different ProductID. If there is only one DeviceVariant, then the value of DeviceVariant/@productId may be used as default value of V_ProductID. If there is more than one DeviceVariant, the best thing is not to use a default value with V_ProductID. If you really need to specify a default value, use a pattern that will be understood by users (e.g. with *, x or ? to signal ambiguity).

3.8 Parameters without a value until written

Sometimes there are parameters which do not have a real value until initialized (written for the first time). When the parameter is read without prior initialization, the device should return a vendor specific ErrorType. Alternatively, if there exists a value that does not occur during normal operation, it could return this value. This value should be described in the IODD as a SingleValue.

3.9 Parameters influencing other parameters

When writing to a parameter leads to a change in other parameters, this parameter must be marked with the attribute @modifiesOtherVariables set to “true” in the IODD. Because it can’t be specified in the IODD which other parameters are influenced, IO-Link Tools have to upload all parameters after each write to such a parameter.

In order to avoid long delays use such parameters with caution and as little as possible.

3.10 Replicating a device

There are two methods for replicating the parameters of a device to another device of the same or compatible type:

1. With the IO-Link Tool:
   Read all read-write parameters from one device, then write to the other device.

2. With the parameter server:
   In the field, just replace the device. On startup of the device, the parameter server will write the parameters formerly read from the previous device.

The list of indices/subindices which are to be replicated is taken from the IODD in case of the IO-Link Tool and is taken from ISDU index 3 in case of the parameter server. Note that these two lists are not necessarily identical. The device may offer a multitude of small parameters in its IODD, each on its own ISDU index for convenient access, and a single parameter record (compact storage object), whose ISDU index is not published in the IODD, for efficient access by the parameter server.

In any case, it is very important that replicating the device with the IO-Link Tool must lead to the same result as replicating the device with the parameter server.

4 Things to consider when writing an IODD

4.1 XML Hints

Although any text editor may be used for writing an IODD, it is recommended to use an XML editor that is schema-aware. While editing, such editors use the IODD schemas to suggest required / allowed elements and attributes. They also do schema validation with one mouse click.

A small selection of XML Editors: (This does not constitute an endorsement by the working group.)

- XML Notepad 2007 (free, see http://www.microsoft.com/downloads/details.aspx?familyid=72d6aa49-787d-4118-ba5f-4f30fe913628&displaylang=en)

- Altova XMLSpy (commercial, see http://www.altova.com/xmlspy.html)
- SDL Xopus (commercial, see http://xopus.com/)
- XMLFox (free, see http://xmlfox.com/)
- XML Copy Editor (free, see http://xml-copy-editor.sourceforge.net/)
- <oXygen/> XML Editor (commercial, see http://www.oxygenxml.com/)
- Easy XML Editor (commercial, see http://www.edit-xml.com/, http://easy-xml-editor.de/)
- Notepad++ with the XML Tools plugin (free, see http://notepad-plus-plus.org/ and http://sourceforge.net/projects/npp-plugins/files/XML%20Tools/)

4.2 Encoding

UTF-8 encoding is mandatory for the IODD. This ensures that characters needed for non-english text can be represented.

But even an editor that supports UTF-8 can only display characters when there is a font installed on that computer that contains these characters. The fonts that come with an English / European Microsoft Windows contain characters for European languages only. To display characters from far-eastern languages, it may be necessary to download and install a unicode font that includes CJK (Chinese, Japanese and Korean) and tell the editor to use this font.

4.3 Different types of IDs

An IODD contains several IDs, whose meaning and using is explained here.

The attribute id is always used as a definition for texts, variables, data types, menus… on which those IODD-parts can reference.

- **Text IDs**

Definition

Text IDs are defined in the language specific part of the IODD.

```
<ExternalTextCollection>
        <PrimaryLanguage xml:lang="en">
                <Text id="TN_SP1" value="Switch point 1"/>
        </PrimaryLanguage>
        <Language xml:lang="de">
                <Text id="TN_SP1" value="Schaltpunkt 1"/>
        </Language>
</ExternalTextCollection>
```

Reference

Text IDs can be referenced from any other element which contains the attribute ‘textId’.

```
<Name textId="TN_SP1"/>
<Description textId="TD_SP1"/>
```

- **Variable IDs**

Definition

Variable IDs are defined in the VariableCollection element.

```
<VariableCollection>
```

```
        <Variable id="V_SP1" index="65" accessRights="rw" defaultValue="2500">
                <Datatype xsi:type="IntegerT" bitLength="16">
                        <ValueRange lowerValue="-900" upperValue="10000"/>
                </Datatype>
                <Name textId="TN_SP1"/>
                <Description textId="TD_SP1"/>
        </Variable>
</VariableCollection>
```

Reference

Variable IDs can be referenced from menus by elements VariableRef or RecordItemRef, attribute ‘variableId’.

```
<MenuCollection>
        <Menu id="M_Sample">
                <VariableRef variableId="V_SwitchingPoint1"/>
        </Menu>
</MenuCollection>
```

- **Menu IDs**

Definition

Menu IDs are defined in the MenuCollection element.

```
<MenuCollection>
        <Menu id="M_MR_SR_SwitchingOutputs">
                <Name textId="TN_SwitchingOutputs"/>
                <VariableRef variableId="V_SP1" gradient="0.001" offset="0" unitCode="1137"/>
        </Menu>
</MenuCollection>
```

Reference

Menu IDs can be referenced from other menu from elements MenuRef**,** attribute ‘menuId’.

```
<MenuCollection>
        <Menu id="M_SwitchingOutputs">
                <Name textId="TN_SwitchingOutputs"/>
                <MenuRef menuId="M_MR_SR_SwitchingOutputs"/>
        </Menu>
</MenuCollection>
```

- **ProcessData IDs**

Definition

ProcessData IDs are defined in the ProcessDataCollection element.

```
<ProcessDataCollection>
        <ProcessData id="P_Analog">
                <ProcessDataIn id="PI_Analog" bitLength="16">
                        <Datatype xsi:type="RecordT" bitLength="16">
                                <Name textId="TN_D_Analog"/>
                                <RecordItem subindex="1" bitOffset="0">
```

```
                                    <SimpleDatatype xsi:type="BooleanT"/>
                                    <Name textId="TN_D_Switchstate1"/>
                                </RecordItem>
                                <RecordItem subindex="2" bitOffset="1">
                                    <SimpleDatatype xsi:type="BooleanT"/>
                                    <Name textId="TN_D_Switchstate2"/>
                                </RecordItem>
                                <RecordItem subindex="3" bitOffset="2">
                                    <SimpleDatatype xsi:type="IntegerT" bitLength="14"/>
                                    <Name textId="TN_D_Value"/>
                                </RecordItem>
                            </Datatype>
                            <Name textId="TN_PI_Analog"/>
                        </ProcessDataIn>
                    </ProcessData>
</ProcessDataCollection>
```

Reference

The ProcessDataIn and ProcessDataOut IDs can be referenced from elements ProcessDataRef, attribute ‘processDataId’.

The ProcessData ID cannot be referenced from within the IODD.

- **Datatype IDs**

Definition

Datatype IDs are defined in the DatatypeCollection element.

```
<DatatypeCollection>
        <Datatype id="D_SwitchingPoint" xsi:type="UIntegerT" bitLength="14"/>
</DatatypeCollection>
```

Reference

Datatype IDs can be referenced from Variables or ProcessData by elements DatatypeRef, attribute ‘datatypeId’.

```
<Variable id="V_SwitchingPoint1" index="64" accessRights="rw">
        <DatatypeRef datatypeId="D_SwitchingPoint"/>
        <Name textId="TN_SwitchingPoint1"/>
</Variable>
```

- **Definition of StandardVariable IDs**

Definition

StdVariable IDs are defined in the VariableCollection element of the IODD-StandardDefinitions1.1.xml File.

```
        <Variable id="V_DirectParameters_1" index="0" accessRights="ro" mandatory="true"/>
```

Reference

StdVariable IDs will be referenced from StdVariableRef elements, attribute ‘id’.

```
<StdVariableRef id="V_DirectParameters_1"/>
```

- **ProductIDs**

Definition

Product IDs are defined in the DeviceVariant and in the Connection element.

```
<DeviceVariantCollection>
    <DeviceVariant productId="SampleDev1">
        <Name textId="TI_ SampleDev1_Name"/>
        <Description textId="TI_ SampleDev1_Descr"/>
    </DeviceVariant>
    <DeviceVariant productId="SampleDev2">
        <Name textId="TI_ SampleDev2_Name"/>
        <Description textId="TI_ SampleDev2_Descr"/>
    </DeviceVariant>
</DeviceVariantCollection>
```

Reference

DeviceVariant/@productId is a plain text which is referenced from the Connection/ProductRef element, attribute ‘productId’.

```
<PhysicalLayer>
    <Connection xsi:type="OtherConnectionT" connectionSymbol="SD1-con-pic.png">
        <ProductRef productId="SampleDev1"/>
        <Description textId="TI_OtherConnection"/>
    </Connection>
    <Connection xsi:type="OtherConnectionT" connectionSymbol="SD2-con-pic.png">
        <ProductRef productId="SampleDev2"/>
        <Description textId="TI_OtherConnection"/>
    </Connection>
</PhysicalLayer>
```

4.4 Recommended prefixes of IDs

Texts:	use prefix “T_” for text IDs
Names:	use prefix “TN_” for text IDs of names
Description:	use prefix “TD_” for text IDs of descriptions
Menus:	use prefix “M_” for menu IDs
Datatypes:	use prefix “D_” for data type IDs
Variables:	use prefix “V_” for variable IDs
Processdata:	use prefix “P_” for process data IDs
Processdata in:	use prefix “PI_” for process data in IDs
Processdata out:	use prefix “PO_” for process data out IDs

4.5 Device identification in the IODD compared to the Device

The DeviceIdentity element looks like this:

```
<DeviceIdentity deviceId="40" vendorId="65535" vendorName="IO-Link Community">
        <VendorText textId="T_VendorText"/>
        <VendorUrl textId="T_VendorUrl"/>
        <VendorLogo name="IO-Link-Logo.png"/>
        <DeviceName textId="T_DeviceName"/>
        <DeviceFamily textId="T_DeviceFamily"/>
        <DeviceVariantCollection>
                <DeviceVariant productId="SampleDev1" deviceSymbol="SampleDev1-pic.png"
deviceIcon="SampleDev1-icon.png">
                        <Name textId="TN_Product1"/>
                        <Text textId="TD_Product1"/>
                </DeviceVariant>
                <DeviceVariant productId="SampleDev2" deviceSymbol="SampleDev2-pic.png"
deviceIcon="SampleDev2-icon.png">
                        <Name textId="TN_Product2"/>
                        <Text textId="TD_Product2"/>
                </DeviceVariant>
        </DeviceVariantCollection>
</DeviceIdentity>
```

Element DeviceIdentity

@vendorId	Corresponds to the mandatory standard parameter DirectParameterPage 1, Subindex 8 (high byte), 9 (low byte), decimal value. The value is assigned by the IO-Link Community, please see www.io-link.com.
@deviceId	Corresponds to the mandatory standard parameter DirectParameterPage 1, Subindex 10 (high byte), 11 (mid byte), 12 (low byte). Decimal value, vendor specific.

With the combination of vendorId and deviceId, an IO-Link device's identification is unique throughout the IO-Link world. Those values are offline values but shall anyway correspond to the information on the designated subindices as described above.

@vendorName	vendor specific name.
VendorText/@textId	The element VendorText is used to give language dependant additional information to vendorName. e. g.: VendorName = "IO-Link Community" VendorText(en) = "Breakthrough in communication" VendorText(de) = "Durchbruch in Sachen Kommunikation" VendorText shall not repeat the vendorName This text does not correspond to any standard parameter of the device, thus it will not be checked in any way. It is only used for displaying.
VendorUrl/@textId	vendor specific URL, should be applied with prefix "http://www.". This text does not correspond to any standard parameter of the device, thus it will not be checked in any way. It is only used for displaying. The IODD Specification V1.1 allowed a maximum of 1024 characters for the URL string, but was changed through CR16 to a maximum of 255 characters. Do not exceed 255 characters.

VendorLogo/@name see IO Device Description Specification

DeviceName/@textId vendor specific device name
This text does not correspond to any standard parameter of the device, thus it will not be checked in any way. It is only used for displaying.

DeviceFamily/@textId vendor specific device family text
This text does not correspond to any standard parameter of the device, thus it will not be checked in any way. It is only used for displaying.

Note: DeviceFamily/@textId, DeviceName/@textId and @vendorName are often used by tools to create their device catalogues. The vendor should take care on a sensible use of those entries.

DeviceVariantCollection

A DeviceVariantCollection can contain IO-Link Devices with the same deviceId. From IO-Link's point of view, those devices are completely the same. They differ only in e.g. mechanical or approval issues and therefore have different orderNumbers or productIds.

@productId Corresponds to the optional standard device parameter Product ID (index 19). If the device contains this parameter and it is referenced in the IODD, tools can overwrite this entry with the uploaded content.

Name Corresponds to the mandatory standard device parameter Product Name (index 18). If it is referenced in the IODD, tools can overwrite this entry with the uploaded content.

Description Corresponds to the optional standard device parameter Product Text (index 20). If the device contains this parameter and it is referenced in the IODD, tools can overwrite this entry with the uploaded content. The IODD Specification V1.1 allowed a maximum of 255 characters for all Description strings. CR17 changes this limit for texts referenced by DeviceVariant/Description/@textId to a maximum of 1024 characters.

4.6 DirectParameterOverlay

When using a DirectParameterOverlay to describe a structure layed over the 16 bytes of V_DirectParameters_2, you should only reference the RecordItems from the DirectParameter-Overlay from your menus. Do not reference the RecordItems of V_DirectParameters_2, because IO-Link Tools may not be able to handle both views on the DirectParameter page 2 at the same time.

4.7 Types of Texts

Texts can either be a 'Name' or a 'Description'. A 'Name' means a common, short term which is often displayed in the tool's table columns. A 'Description' can be a text which describes the issue properly. Tools show descriptions oftenly as tooltips. The only allowed text formatting is the line break using the LineFeed character (encoded as "
").

4.8 Specialist role menus

All variables required for replicating the Device (usually those with access rights 'rw') shall be referenced from menus visible in the specialist role.

4.9 Languages

Usually all initially supported languages are included in the main IODD. External language files enable delivering additional languages afterwards without touching the original IODD. This is

useful if a company's subsidiary in a foreign country wants to add the local language on their own.

The standard definition files (IODD-StandardDefinitions1.1.xml and IODD-StandardUnit-Definitions1.1.xml) currently only support English, French and German. If you plan to support additional languages in your IODD, please contact the PNO CC/PG1 Technology, Team IODD to add these languages to the standard definition files in a joint effort.

4.10 Using StandardVariables

Mandatory standard variables shall always be referenced in the IODD. E.g.

For a device using only direct parameters:

```
<VariableCollection>
        <StdVariableRef id="V_DirectParameters_1"/>
        <StdVariableRef id="V_DirectParameters_2"/>
        <DirectParameterOverlay id="V_...">
        ...
        </DirectParameterOverlay>
</VariableCollection>
```

For a device using ISDUs:

```
<VariableCollection>
        <StdVariableRef id="V_DirectParameters_1"/>
        <StdVariableRef id="V_DirectParameters_2"/>
        <StdVariableRef id="V_ProductName"/>
        ...
</VariableCollection>
```

All other standard variables are optional, thus they shall only be referenced if the device supports them.

If the device supports them, please refer to the IODD-StandardDefinitions1.1.xml file for the correct ID-designation.

4.11 Dos and Don'ts

- Never write text directly into attributes textId.
- When describing an enumeration as a list of SingleValues, do not include the numerical value into the text referenced by Name/@textId. It is up to the enginering tool to display the value in addition to the name (not necessarily as an additional row, but maybe as tooltip or only in a debug mode).
- @minCycleTime is given in µs. Don't apply the value from DirectParameterPage 1, Subindex 3

  ```
  <TransportLayers>
          <PhysicalLayer baudrate="COM2" minCycleTime="2300" sioSupported="true"/>
  </TransportLayers>
  ```
- schemaLocation is given properly

  OK: xsi:schemaLocation="http://www.io-link.com/IODD/2010/10 IODD1.1.xsd

  KO: xsi:schemaLocation="http://www.io-link.com/IODD/2010/10 ..\..\IODD1.1.xsd
- Assign @releaseDate and @version properly. Increase @version for every release.

- In DirectParameterPage 1 or 2, bitOffsets range from 0 to 127. Take care on reserved areas (DirectParameterPage 1: completely reserved, DirectParameterPage 2: nothing reserved)

- Menu entries involving floating point values (of type Float32T or implicitely by using gradient and/or offset) should use attribute displayFormat with "Dec.x" instead of just "Dec" to have a better control over the tool behaviour.

- If a device supports DataStorage the attribute Features/@dataStorage shall be set ‘true’. IO-Link tools shall implicitly use the SystemCommands ParamUploadStart, ParamUploadEnd, ParamDownloadStart, ParamDownloadEnd, ParamBreak and ParamDownloadStore to encapsule the up- or downloaded parameter set. A flowchart in Chapter 7.11 shows the sequence.

5 Examples

5.1 Device with DirectParameterPage

This example shows the usage of the DirectParameterPage for handling device specific parameters. The device does not support ISDU access and has a Boolean value as its process data input.

Please note how the <DirectParameterOverlay> element describes an overlay over the entire DirectParameterPage 2. The provided record covers the whole DirectParameterPage.

For IO-Link Tools which additionally need a description of the individual subindices of the DirectParameterPage 2, the TextRedefine elements are used.

5.2 Device with ISDU

The device in this example supports ISDU access. Thus, it doesn’t use any device specific parameters in the DirectParameterPage 2.

For this example only the primary language is specified and no pictures are referenced.

5.3 Device with ApplicationSpecificTag variable

The standard variable ApplicationSpecificTag is referenced by this device. Since the specified length for the standard variable is 64 bytes, the length of the variable string is overwritten with the fixedLengthRestriction attribute to reduce the needed device memory for storing the value.

5.4 Analog device

This example shows the process data definition of a typical analog device. The process data is a record with an integer value and two signalling bits. Using the lower bit offsets for the signalling bits ensures that the influence on the value of the record is insignificant, when interpreted as an integer.

5.5 Device with events and internal language specification

This device is featuring a teach event and the static text resources of the IO Device Description are localized in English and German.

5.6 IODD with external language and pictures

The example shows the usage of external language files. Please note, although belonging to an IODD file, the external language file has its own stamp. Thus it must be checked in a separate run of the Checker tool.

The picture files may be referenced by more than one IODD file. See the vendor logo file as an example.

5.7 Referenced standard variable with value definitions

The standard variable referenced in this example is V_SystemCommand, which allows the device to use the predefined values of the IO-Link standard and to add device specific commands. After all, only values mentioned in the IODD are valid for this variable. Values and value ranges are not allowed to overlap.

5.8 IODD with menu definition

The example shows the typical usage of a button menu item for a SystemCommand value.

Additionally another menu item displays a hysteresis variable in two different units, by using unitCode and gradient attributes. The offset attribute is not necessary in this case.

The usage of the display format “Dec” with number of fractional digits, and the special display formats “CycleTime”, “OffsetTime” and “RevisionID” is also shown.

5.9 Device with Conditions

The shown device features two different data types for its process data. The V_Condition variable is used to switch between the two process data layouts. Note that the process data length stays fixed while switching.

The standard variable V_ProcessDataInput must be supported by the device in order to be able to reference the process data from the user interface definition. The menu in this example references both sorts of process data according to their definition and their condition.

Process data is available to IO-Link Tools from two sources: Directly from the IO-Link Device via V_ProcessDataInput / V_ProcessDataOutput (optional) or indirectly from the IO-Link Master (optional). The process data from V_ProcessDataInput / V_ProcessDataOutput can be referenced in a menu and formatted using displayFormat, gradient, offset and unitCode. The process data from the IO-Link Master is formatted using ProcessDataRefCollection, which is demonstrated here.

5.10 Device with Diagnostics

This device has error types and events. Some of them are standardized and are thus described in IODD-StandardDefinitions1.1.xml and only referenced here. The others are device-specific and thus described in the IODD.

5.11 Device with Variables with special behaviour

This example show the typical use of the following attributes, which control the behaviour of a variable.

Table 1 – Variable behaviour

Attribute name	Behaviour when false (default)	Behaviour when true
dynamic	The value does not change except when written to.	The value may change spontaneously.
modifiesOtherVariables	Writing to the variable does not influence other variables.	Writing to the variable may change any or even all other variables.
excludedFromDataStorage	The value is stored in Data Storage.	The value is not stored.

5.12 Device with several device variants with their connections

This device comes in several variants which differ in their physical connections (plug, cable).

5.13 Device according to IO-Link Smart Sensor Profile

This device uses a variable representation according to the function classes defined in the specification of the Smart Sensor Profile. These variables are not part of the StandardDefinitions files and thus need to be described in the IODD. Note, that the use of the corresponding indices is restricted to the Smart Sensor Profile only. The IODD checker will generate a corresponding warning, when these indices are referenced in an IODD.

6 Releasing the IODD

6.1 Checker

When you're done editing your IODD, and the schema validation in your favourite XML editor does not report any errors, you are now ready for the mandatory checking and stamping of the IODD.

The IODD Checker, available from the download section of http://www.io-link.com/, is used for this. The Checker is continually improved, so please use the latest release of the IODD Checker.

The Checker is a console program. Open the Windows Console, change to the directory where the IODD Checker is:

```
C:
cd "<Programs directory>\IO-Link Community\IODD V1.1 Checker"
```
(<Programs directory> is usually "C:\Program Files").

Make sure your graphics files are present in the same directory as your IODD. Then check your IODD using:

```
IODDChecker <IODD path + file name>
```

In the past, XML parsers did not catch all schema violations. To make sure your IODD is fully schema compliant, it is recommended to use more than one parser.

If you have installed the Xerces-C++ XML Parser, you may include this additional parser by:

```
IODDChecker -xe -xp<Xerces path> <IODD path + file name>
```

When the check does not produce any more errors, stamp your IODD. The command line is the same as above, just add `-s`.

If you created external text documents, put them in the same directory as your main IODD. Check and stamp them now using the command line as shown above. The external text documents are checked against the main IODD file. After the external text document was stamped, there could be changes to the main IODD file which make the external text document invalid. To protect against this, the CRC of the main IODD is included in the CRC calculation of the external text files. This means: Everytime you modified and stamped your main IODD, you have to stamp all external text documents again.

Using the IODD Checker and Xerces as additional parser, when you notice that Xerces produces an error for each XML element and attribute while the .NET Parser does not produce errors, the likely cause is a schemaLocation that uses a path prefix with the IODD schema name. According to the IODD specification, the header shall look like this for the main IODD:

```
<IODevice xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.io-link.com/IODD/2010/10" xsi:schemaLocation="http://www.io-link.com/IODD/2010/10 IODD1.1.xsd">
```

And it shall look like this for an external text document:

```
<ExternalTextDocument xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.io-link.com/IODD/2010/10" xsi:schemaLocation="http://www.io-link.com/IODD/2010/10
IODD-Primitives1.1.xsd">
```

6.2 Deployment

For each combination of Vendor ID and Device ID, there shall be exactly one current IODD. There may be older revisions, but only one is current. An IO-Link Tool detects the current IODD by inspecting DocumentInfo/@releaseDate (which must be the same as the date field in the IODD file name).

Note that an IO-Link V1.1 device that is backward compatible to IO-Link V1.0 has an IODD V1.0.1 in addition to its IODD V1.1.

During development there may be a multitude of IODD releases on the same day carrying the same release date and version. But when it comes to releasing the IODD to the public, there shall not be multiple IODD releases with the same version or date.

The recommended way to offer the IODD (e.g. as a download from a web server) is as a single ZIP archive file. The following files shall be added:

- The main IODD file (XML)
- The graphics files referenced from the main IODD file (PNG)
- The external language files (optional, XML)
- A readme or release notes (optional, e.g. TXT or PDF)

The following files shall not be added:

- The IODD schemas (IODD1.1.xsd, ...)
- The W3C standard schemas (xml.xsd)
- IODD standard XML files (IODD-Standard[Unit]Definitions1.1[language code].xml)
- The output of the IODD Checker (log file)

The following data should be offered as separate files or archives:

- The device manual, assembly instructions, ...
- The manufacturer declaration and other certificates
- FDT Device Type Manager or other device specific tools
- CAD / CAE data
- Firmware update files

7 Things to consider when writing an IO-Link Tool

7.1 IODD Checker releases

According to chapter 5 of the IO Device Description Specification, IO-Link Tools shall compare the checksum in the Stamp with the checksum calculated from the file contents, and shall reject IODDs when there is a mismatch.

That way, IO-Link Tools can rely on the IODD Checker and do not need to re-implement all the complex checks that the IODD Checker performs.

Over time, the IODD Checker is enhanced and checks are added in newer releases.

IO-Link Tools shall accept all IODDs with a correct CRC that were stamped by any officially released IODD Checker.

This means

1. IODDs stamped by a pre-release of the IODD Checker (marked with "beta") do not need to be accepted.
2. IO-Link Tools can only rely on those checks that were present in the first official release of the IODD Checker. All the checks added later must also be implemented in the IO-Link Tool.

7.2 Display Names of User Roles and Menu Names

The following texts are recommended to be used by IO-Link Tools.

Table 2 – User Roles

User Role	ObserverRoleMenuSet	MaintenanceRoleMenuSet	SpecialistRoleMenuSet
English	Operator	Maintenance	Commissioning
German	Bedienen & Beobachten	Wartung	Inbetriebnahme
French			
Spanish			
Italian			

The terms in French, Spanish and Italian have not yet been defined. Contributions are welcome.

Table 3 – Menu Set

Menu	IdentificationMenu	ParameterMenu	ObservationMenu	DiagnosisMenu
English	Identification	Parameters	Observation	Diagnosis
German	Identifikation	Parameter	Beobachten	Diagnose
French	Identification	Paramètres	Observation	Diagnostique
Spanish	Identificación	Parámetros	Observación	Diagnósticos
Italian	Identificazione	Parametri	Osservazione	Diagnosi

7.3 Usage of the StandardDefinition files

Tools should use the IODD-StandardDefinitions1.1*.xml files for creating an internal representation of the IO-Link standard variables. Those files can be interpreted using the same routines as used for interpreting the IODD.

The IODD-StandardUnitDefinitions1.1*.xml should be used to decode any numerical unit codes to text.

7.4 Using the IODD schemas for code generation

To facilitate tool development, you might consider generating source code from the IODD schemas.

For C# / .NET read here:

- **CodeXS**
  http://www.bware.biz/DotNet/Development/CodeXS/Article/Article_web.htm or
  http://www.codeproject.com/KB/cpp/CodeXS.aspx
- **XML Schema Definition (Xsd.exe) tool**
  http://msdn.microsoft.com/en-us/library/x6c1kb0s%28v=VS.100%29.aspx
- **Sample Code Generator (XSDObjectGen) tool**
  http://www.microsoft.com/downloads/details.aspx?FamilyID=89e6b1e5-f66c-4a4d-933b-46222bb01eb0&DisplayLang=en

Even when writing code by hand, the hierarchy of the complex types in the IODD schemas is a good prototype for your class hierarchy.

7.5 Handling of defaults

Reading an IODD with or without schema validation influences the handling of default values, so care must be taken:

- With schema validation, optional attributes that have a default value in the schema will be automatically added with their default value. In case you want to know whether the attribute actually was in the IODD or was added by the default in the schema, the post-schema-validation infoset (PSVI) must be consulted.
- Without schema validation, optional attributes that have a default value in the schema will not be added. All defaults must be programmed in the IO-Link Tool.
- Some attributes have defaults which can't be expressed in XML schema. These have to be programmed in the IO-Link Tool regardless of schema validation.

7.6 Variable handling in IO-Link Tools

Tools shall only read / write the variables accessible in the current role without condition dependency.

Variables which are not referenced in a menu will not be accessed. If a variable is described in the IODD but can not be accessed (e.g. Index or Subindex not available), it is up to the IO-Link Tool, whether it continues uploading with the following variable or whether it stops the upload process and discards all values. This means the IODD should only reference variables which are always accessible.

7.7 Read-write variables without @defaultValue

An IO-Link Device is instantiated (taken from the catalog and placed into the project). The IODD contains variables with accessRights="rw" but without a @defaultValue.

In this situation, IO-Link Tools should leave the field for the value of such variables empty and set the state of the variable to "empty". As long as the state is "empty", the value should not be written to the device.

If the IO-Link Tool is not able to leave a variable empty, it should select the "natural" default value according to the data type of the variable (e.g. 0 or the empty string). If the "natural" default is not included in the allowed values (expressed via SingleValues and ValueRanges), the IO-Link Tool shall pick one of the allowed values, e.g. the lowest or the first value.

7.8 V_ProductID

See also chapter 3.7. If a default value is given for V_ProductID and the IODD contains more than one DeviceVariant, the default value may contain a pattern that can be understood by users. In this case the default value of V_ProductID does not match with any of the values of DeviceVariant/@productId. Tools have to get along with this.

7.9 Replicating a device

Tools should implement a function for replicating a device which works independently of the current user role. This function should upload all variables which are refenced in the SpecialistRole with access rights 'rw' into a temporary storage, prompt the user for device replacement, and then download those variables to the other device.

7.10 Preparametrization of a device

Use case:

A system has a data storage enabled IO-Link master port. An IO-Link device which is not connected to the system shall be preparametrized with an external service tool. The data of the IO-Link device shall be uploaded when the IO-Link device is connected to the IO-Link master port of the system. Therefore, the external service tool should implement a user command (e. g. button) which sets the IO-Link device into the required state.

Tool implementation:

If a tool supports this functionality, the tool shall give a warning to the user, if the IO-Link device is set to the state where the variables would be uploaded from the device, if it is connected to the master port in the system. The warning shall give a hint for the user to label the device “preparametrized”.

Caution:

If such a device is used accidently for a normal device exchange, not the expected behaviour will appear. Not a download from the data storage will be performed, but an upload from the device.

7.11 Tool behaviour during Up- and Downloading sequence

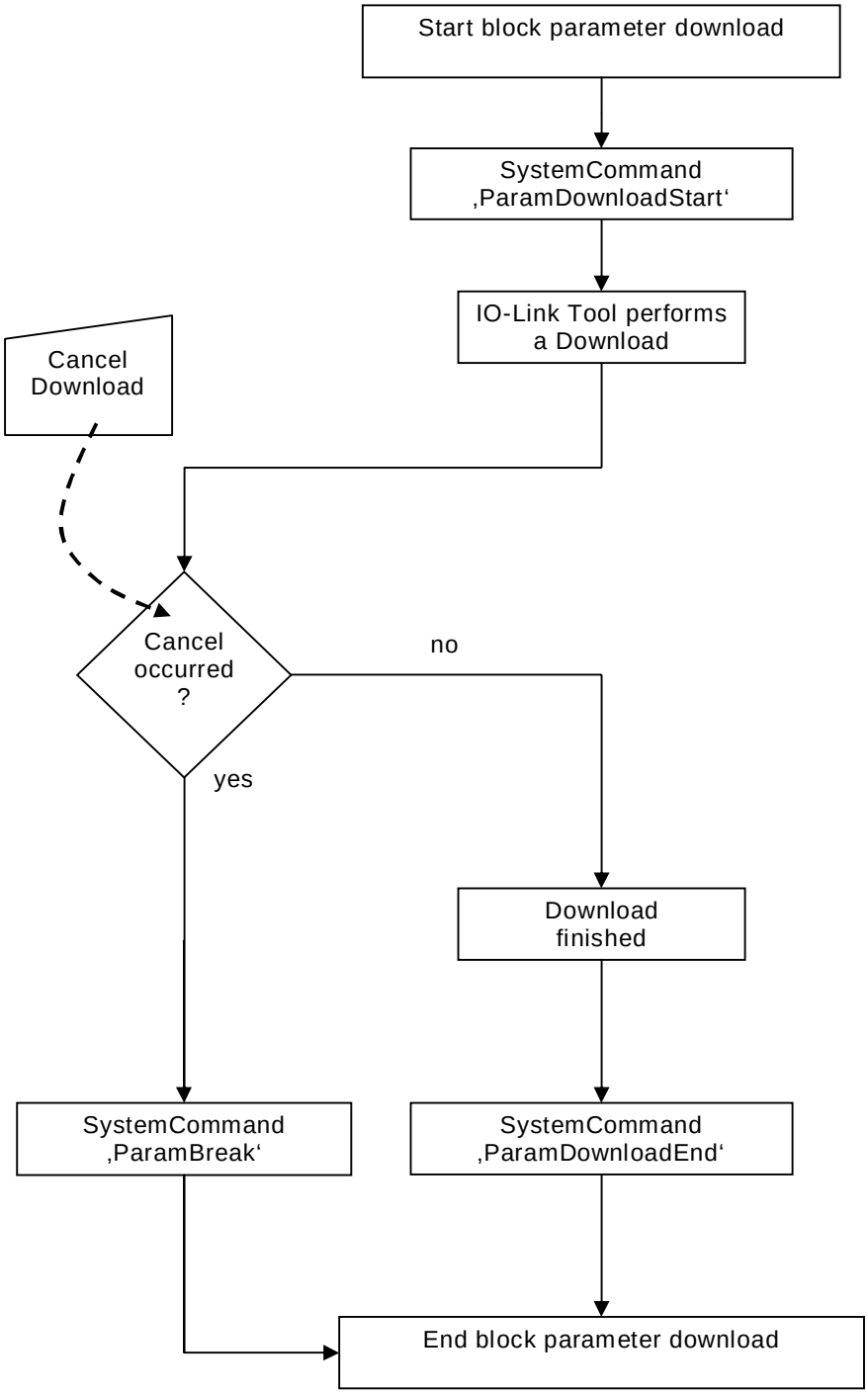

Figure 1 – Block parameter download

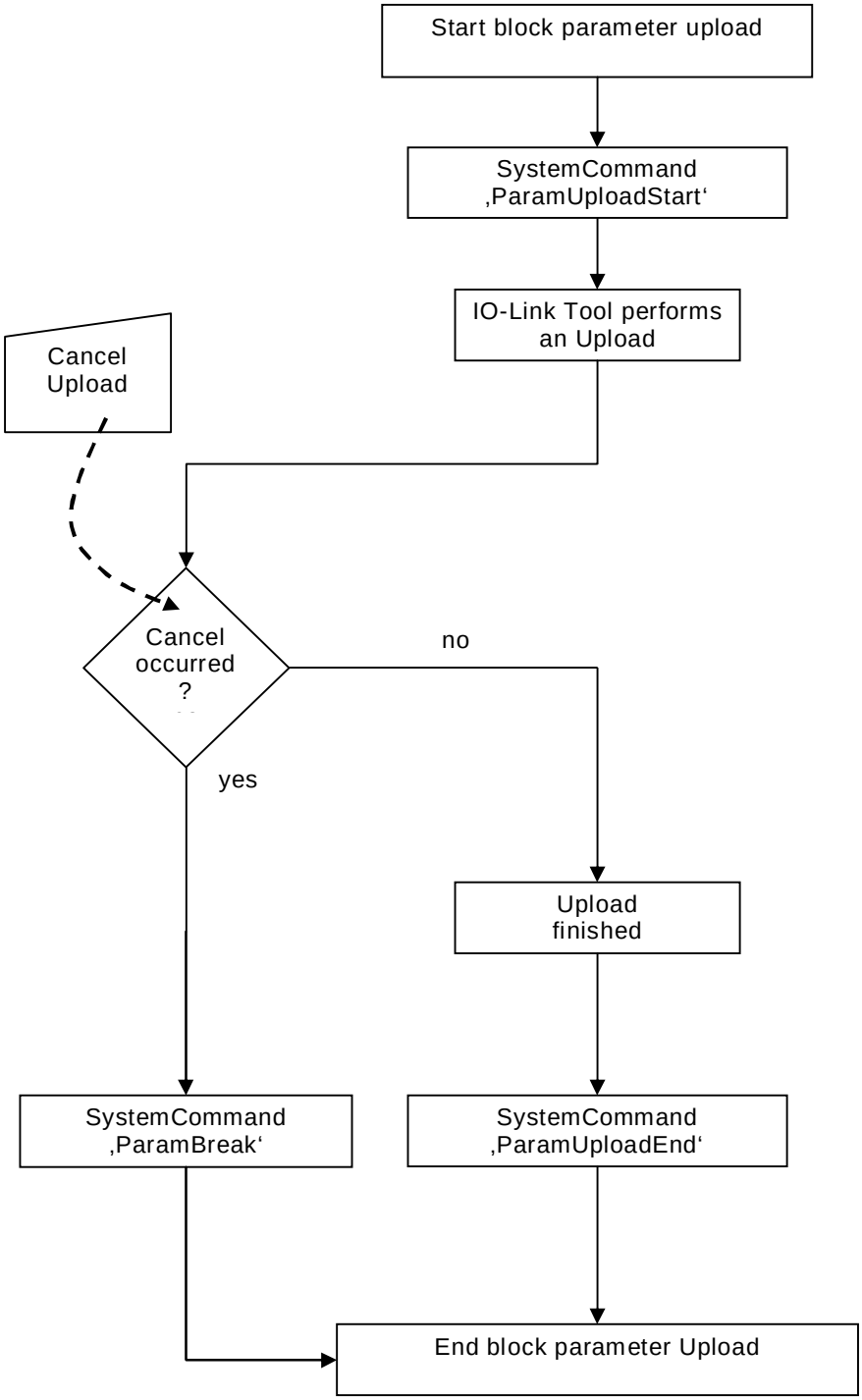

Figure 2 – Block parameter upload

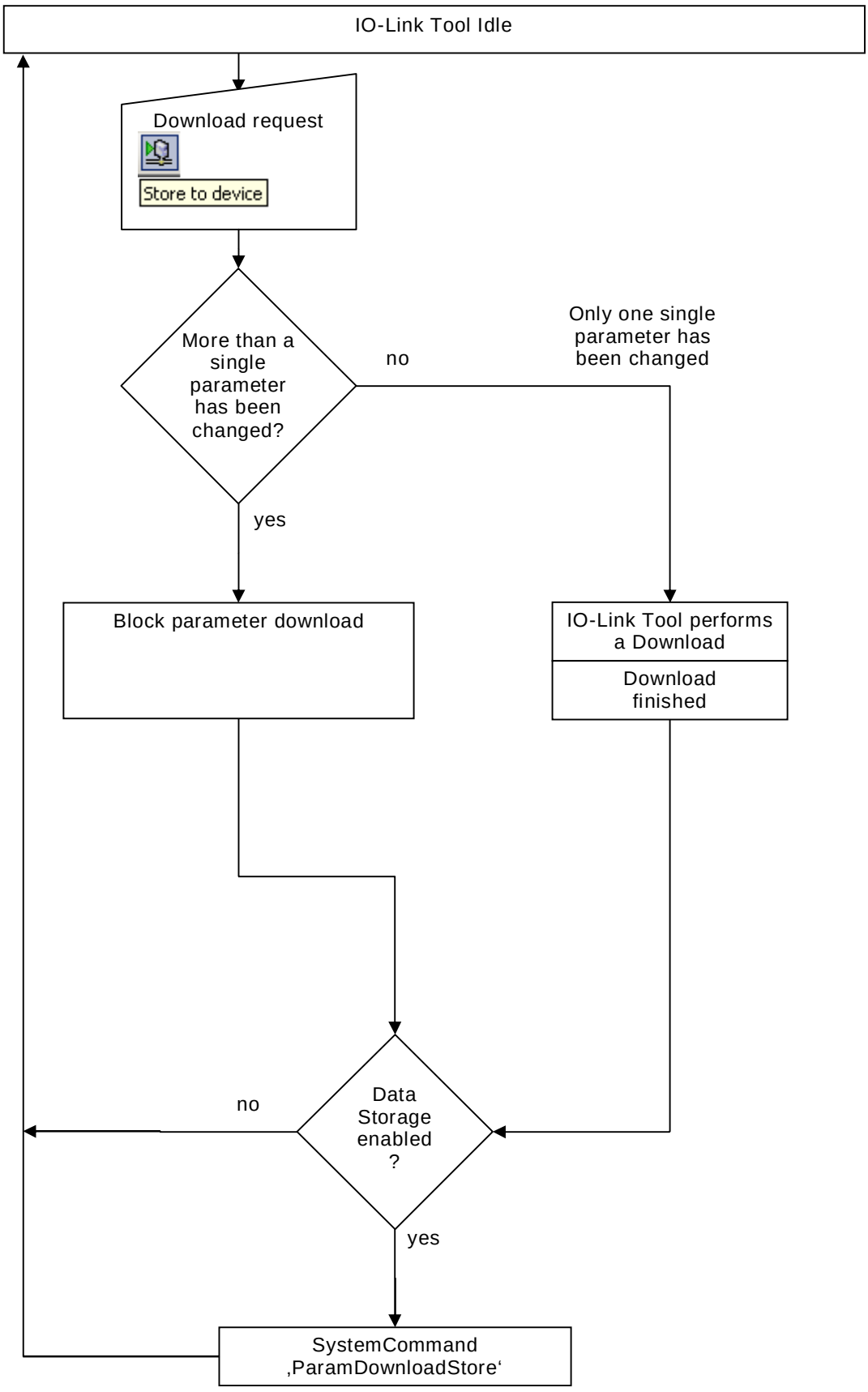

Figure 3 – Download

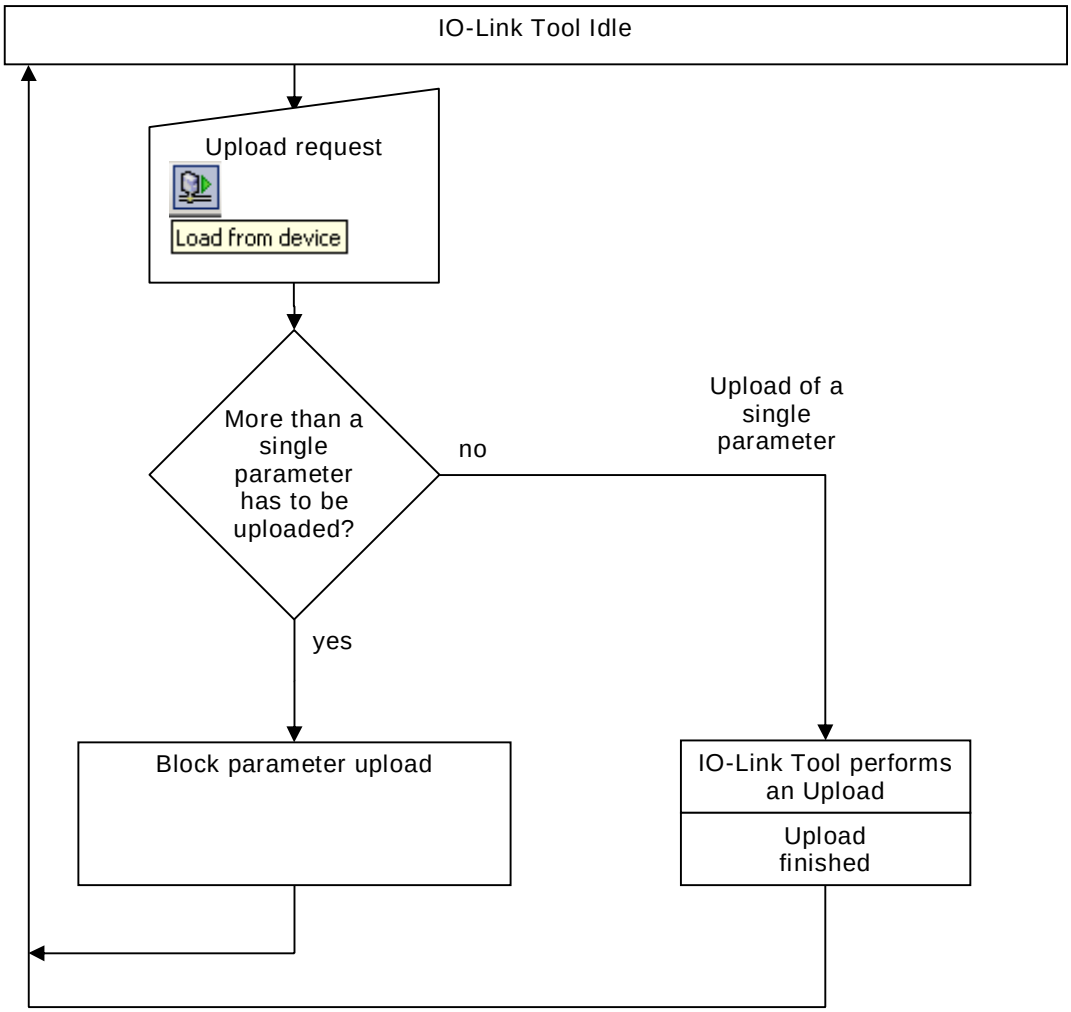

Figure 4 – Upload

7.12 Finding a DeviceVariant or a Connection by its 'productId'

Finding the correct DeviceVariant after reading V_ProductID from the device, or finding the corresponding Connection for a DeviceVariant is done by searching the appropriate element by the value of the 'productId' attribute.

This is the only reference within the IODD that uses the type 'xsd:string' instead of 'RefT'. When navigating, do not use the value of 'productId' directly in an XPath expression. This would result in a security hole (XPath Injection).

7.13 additionalDeviceIds

The attribute DeviceIdentity/@additionalDeviceIds contains a list of device IDs which are supported by this device. For an IO-Link Tool, the only use of this list is to display it to the user. There is no action connected with it.

7.14 Testing an IO-Link Tool

The manufacturer of an IO-Link Tool shall of course test it, but the IO-Link Community currently has no test specification and no manufacturer declaration for tools.

As an aid for testing the IODD import, there is a set of interpreter test IODDs available on www.io-link-projects.com. These cover the quantity structure, the different data types, plus more. Over time this set will be enlarged.

All the IODDs from this set use a different deviceId, so they may be imported in parallel.

It is recommended to test the import of IODDs by automatically importing all of these IODDs, and to selectively use some of these IODDs for more thorough (probably manual) tests.

8 IO Device Description Specification Version 1.1 errata

CR3: SingleValues and ValueRanges do not need xsi:type

Chapter 7.5.3 Data types: The examples use xsi:type within the SingleValue and ValueRange elements. This is not necessary and should be avoided because the type is already determined by xsi:type within the superordinated DataType element.

CR4: IODD quantity structure must be detailed

Annex B Definitions of IODD quantity structure: The entries "Number of <element name>" are not meant as the total number of XML elements with that name in the whole IODD. Instead, they are meant as "Number of XML elements of that name within the superordinated element".

E.g. "Number of Datatypes" is the number of Datatype elements within the DatatypeCollection element. This does not constitute the total number of data types which may appear in an IODD, because each Variable and ProcessDataIn/ProcessDataOut element may additionally use an embedded data type.

"Number of Variables" is the sum over all StdVariableRef, DirectParameterOverlay and Variable elements within the VariableCollection.

In addition, there is an editorial error in the lower part of the table: A slash is missing between the element names Variable and Datatype and the attribute name @id.

CR5: If data storage is not supported, excludedFromDataStorage may be omitted

Chapter 7.5.4.1 – 3: If Features/@dataStorage is set to "false" (data storage is not supported), then read-write variables do not need the attribute excludedFromDataStorage. It is implicitely "true".

CR6: V_DirectParameters_1 has index 0 instead of 1

Chapter 7.5.8.4, Table 5 – Standard variables with special display: V_DirectParameters_1 is listed several times with index 1. This is wrong, the index is 0.

CR7: Enhance description for Features/@profileCharacteristic

Chapter 7.5.1 Features: The description of the attribute profileCharacteristic lists "o" (use == optional). Make clear what this means here:

This attribute shall be given if and only if the device adheres to the IO-Link Profile Smart Sensor Specification.

CR8: How much of the IODD must an IO-Link Tool support

Chapter 7 Device Description: "IO-Link Tools shall interpret and use the content of IODDs completely". Actually, this requirement is too strong.

An IO-Link Tool shall support:

- At least one catalog entry for each IODD (derived from DeviceName)
- All data types, including names of ValueRanges

- Variables
- Menus, including Buttons
- ErrorTypes and Events
- Conditions
- StandardDefinitions (StandardVariables, StandardErrorTypes, StandardEvents) – the newest version at the time when the Tool is developed
- User Role "Specialist"
- Texts in the PrimaryLanguage (English)
- Refresh of dynamic variable values (on demand or cyclic)
- When variables marked with "modifiesOtherVariables" are changed, the tool shall either notify the user that other variables are possibly changed, or the tool shall automatically reload.
- When parameters are up- or downloaded, any ErrorTypes that occur shall be displayed in a way that the user is able to associate them with the parameter that triggered it. Either by displaying it at the parameter or by telling the parameter name in the error log.

An IO-Link Tool should support, but is not obliged to:

- Separate catalog entries for each DeviceVariant
- A separate display of the Process Data (in addition to V_ProcessDataIn / V_ProcessDataOut)
- Display of the Connection
- Texts in other Languages than the PrimaryLanguage
- User Roles other than "Specialist"
- Vendor logo, pictures and icons.
- For menu entries having gradient and/or offset: additional display of the raw value
- ErrorTypes and Events: additional display of the raw values

CR9: SupportedAccessLocks versus V_DeviceAccessLocks

Chapter 7.5.4.1 StdRecordItemRef, add the following rule: “For StdVariableRef id="V_DeviceAccessLocks", StdRecordItemRef is only allowed for those subindices which refer to an access lock that is supported, i.e. where the respective attribute in Features/SupportedAccessLocks is set to "true".

Note: With V_DeviceAccessLocks StdRecordItemRef is only needed if you want to specify a defaultValue for a specific lock. It is not suited to indicate which of the access locks are supported.

Chapter 7.5.8.5 RecordItemRef, add the following rule: “A RecordItemRef with variableId="V_DeviceAccessLocks" is only allowed for those subindices which refer to an access lock that is supported, i.e. where the respective attribute in Features/SupportedAccessLocks is set to "true".

CR10: Special floating point values

Chapter 7.5.3.1.5 Float32T lexical representation, change this to:

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

NaN is removed because:

1. It is not comparable and thus can't be used for ValueRange/@lowerValue and @upperValue

2. It could be used for SingleValue/@value or @defaultValue, but NaN can be represented by a lot of bit combinations (see IEEE Std 754-2008, *IEEE Standard for Floating-Point Arithmetic*). It is unclear which value should be sent to an IO-Link Device by an IO-Link Tool.

CR12: Document Type Definition (DTD) shall not be used

Chapter 5 Files, add the following rule to the second paragraph: "IODD XML files shall not use any DTD (Document Type Definition, see http://www.w3.org/TR/xml/#sec-prolog-dtd)."

CR13: Condition Variable in ProcessData shall be referenced in Menu

Chapter 7.5.5 Process data collection, add the following rule to the list after Figure 32: "For each user role there shall be a menu (could be the same) that contains at least a read-only reference to the Variable / RecordItem used in the ProcessData/Condition elements."

CR16: Max. Length of URL

Annex B Definitions of IODD quantity structure: The maximum length of an URL is restricted to 1024 characters. There is no standard defining this number. Although according to sources in the internet, current clients and servers seem to support at least 2048 characters, this limit seems exaggerated. This limit will be reduced to 255 characters in the next version of the IODD Specification. Up to then, the IODD Checker will warn on lengths exceeding 255 chars.

CR17: Allow 1024 instead of 255 characters for strings referenced by DeviceVariant/Description/@textId

Annex B Definitions of IODD quantity structure: The maximum length of strings referenced by Description/@textId is 255. In the special case DeviceVariant/Description/@textId this limit is enlarged to 1024 to enable the exact same sales text for the device variant as in other media.

CR18: CRC calculation must be detailed

Chapter 7.3.4 File validation: The CRC calculation must be detailed.

The IODD file is read in binary mode. The stream of bytes is fed into the CRC algorithm until the string `<Stamp crc="` has been processed. The value of the attribute crc is skipped, and the CRC calculation continues with the closing quotation mark.

The same is done with external language documents, but after the end-of-file has been reached, the CRC of the main IODD file is converted to decimal representation (no leading zeroes) and the character codes for the digits are fed into the CRC algorithm.