# Development of a Universal Model for Description of Intelligent Field Devices through the Life-Cycle

Konrad Gnauck, Andreas Gössling, Roman Frenzel, Martin Wollschlaeger
Industrial Communications, Institute of Applied Computer Science,
Faculty of Computer Science, TU Dresden, 01062 Dresden, Germany
{konrad.gnauck, andreas.goessling, roman.frenzel, martin.wollschlaeger}@inf.tu-dresden.de

Dirk Schulz
ABB Corporate Research Ladenburg
Wallstadter Str. 59, 68526 Ladenburg, Germany
dirk.schulz@de.abb.com

## Abstract

*Integration of field devices is currently performed using different device models which all have a different focus. These models are represented by a vast number of different device description formats. For effectiveness of the integration processes, a comprehensive device description model that catches all relevant information is required.*

*Based on an analysis of important formats, a device description model has been designed that covers all relevant information that was formerly stored in those different formats. In this paper the requirements for such a model are discussed, and a universal model integrating existing definitions is described.*

## 1. Introduction

In recent years, the software and hardware capabilities of automation devices have experienced a significant increase. Thus, modern intelligent field devices offer a functional variety from simple input/output functions over complex measurements of different values to efficient actuators in order to provide a large set of additional information like diverse diagnosis data about the process as well as about themselves. A new potential is unlocked for more efficient production as the data might be used for sophisticated applications such as predictive maintenance or enterprise asset management and can therefore lead to higher availability and durability of plant facilities. A more cost-effective production and a better product quality can be achieved.

To benefit from this potential, it is necessary to provide effective methods to make the appropriate information available for the different use-cases during the (device) life cycle. This ranges from description of device requirements and capabilities for device selection to the integration of devices into enterprise structures without huge engineering efforts. Additionally, mechanisms for simple supervision during operation and for maintenance are required.

During their life cycle, devices are handled by sundry persons working on dissimilar phases and having different goals. Thus, they have a different view on information related to a device. To meet the special requirements of each view, specific description formats have been developed independently, each covering a particular set of information [1].

Since the introduction of the first fieldbus systems, the related fieldbus organizations have standardized methods for describing data on field devices. Description formats like General Station Description (GSD) or Common File Format (CFF) basically contain parameters which are required for tool-based network and communication configuration. As the formats have different origins and a fieldbus-specific scope, they differ in syntax, have different semantics for the same elements, and are not sufficient for all use-cases during device integration.

There are fieldbus-independent concepts like Field Device Tool (FDT) or Electronic Device Description (EDD) but they do not share a common technology base.

Since the various fieldbus-specific and -independent technologies are all established in today's automation systems and are tailored toward different use-cases, device vendors still face the huge effort to support all these different technologies, while engineers must be capable of handling them.

The effort from this large number of diverse description formats is also acknowledged by the automation industry. There are ongoing efforts within the Future Device Integration (FDI) working group toward a harmonization of the next generation of formats.

In summary, almost all stakeholders along the device and plant life-cycle face this diversity in formats and the

978-1-4244-2728-4/09/$25.00 ©2009 IEEE

inconsistencies between them. Device vendors have to provide and maintain several description formats to enable costumers to use the devices in their particular systems (see figure 1).

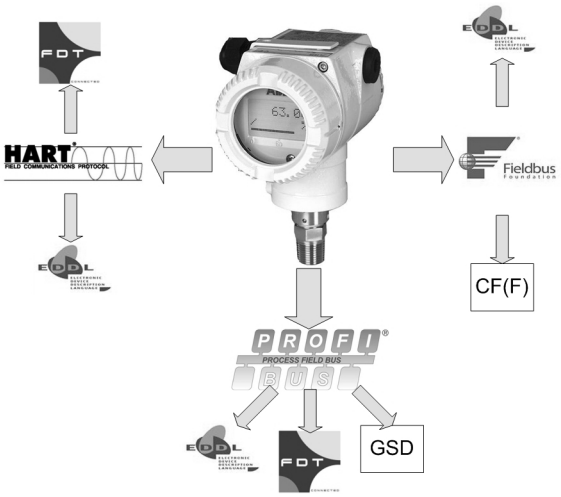


**Figure 1: Exemplary description files for one device**

For a correct representation of the capabilities of their devices, vendors have to deal with the variety of device description models and integration techniques including their ongoing development. Vendors of software solutions like asset management tools or control systems have to provide and maintain interfaces for the import of those file formats. Moreover, it is often necessary to adjust the descriptions to special requirements and capabilities of the different tools. Lastly the end user of the devices has to face these technologies. He has to choose which fieldbus system, which control system, which engineering tools, which device vendor(s) and which devices meet his requirements and cover certain special needs. Based on this, he has to obtain the correct device description(s). Due to different identification and version control mechanisms, this is not generally easy and causes additional costs, an effect contrary to the aspired benefit of the usage of intelligent field devices.

In this paper, we present a consistent and universal description model which covers information for the whole life cycle, different fieldbus interfaces, different classes of devices and more. This is a meant to be a step towards the goal of overcoming all these disadvantages. Such a universal and comprehensive model for device description has the chance to be accepted by device vendors, as it has the advantage of having the need to create, certify and maintain only one description for each device, and, from a users' point of view, the need to obtain only one description which covers all required information.

Section 2 will introduce some related work on this topic. In section 3, requirements are presented as the basis for building a universal description model which is done in section 4. After presenting a proof-of-concept in section 5, section 6 will give a conclusion of this work and closes with an outlook on the next steps for related research.

## 2. Related work

The comparison of current device description models and formats shows that they are all limited to a certain subset of device information regarding the phases within the life cycle. Additionally, no single existing device description format is adequate to provide all required data from the device user's point of view [3]. Even the device description formats that were derived from ISO 15745 "Open Systems Application Integration Framework", which ought to provide a common basis, do not cover the same device-related use-cases [2].

For solving this problem, a common definition of all information that is required during the life-cycle must be found, as already proposed in an article in the computer&automation magazine by Braune and Wollschlaeger [2].

In an atp article by John et al., an approach based on function-oriented use-cases during the device life cycle was used, which results in a (fieldbus) technology independent device model for the usage with OPC UA [1].

In a paper for the SPS/IPC/Drives congress 2008, Thilo Schumann presents a method for device behavior description by state machines and the usage of SCXML [6].

The authors Mühlhause, Diedrich and Riedl state in a paper of the Automation 2008 conference, that there are information gaps along the life cycle of plants, especially between the information for planning DCS or MES systems and during their operation. As a conclusion, they map the information between the different models of NE100 and EDDL to ensure a consistent usage of data from different sources during the life cycle [5].

## 3. Requirements

Before even going into fieldbus-specific details, the first step in developing a common device description model is the collection of general requirements that such a model must fulfill. To support all fieldbuses and use-cases, a future description format must be based on such a common device model.

### 3.1. User requirements

The NAMUR - "User Association of Process Control Technology and Pharmaceutical Industries" has collected several requirements for device integration and thus also for device description from the device users' point of view. These requirements are published in the NE 105 - "Specifications for Integrating Fieldbus

Devices in Engineering Tools for Field Devices". The main focus of the NE 105 is to capture the user requirements that exist for safe and efficient device operation and configuration in a uniform toolset.

In this sense, information about the primary device functions, supported process environments, and ordering information, as described in the NE 100, is of particular interest.

Instead of all the specific description formats and device-related files that are currently in use, a single device package (archive) with clear versioning information would already significantly increase the ease of use of device information.

Another major user requirement is the support of internationalization within the device description model. Language dictionaries may be used to present information that is shown to an operator or other intended human actors.

### 3.2. Generic Requirements

Next to the mentioned basic user requirements, there are further requirements derived through analysis of the state-of-the-art of description formats.

The device description model and its corresponding formats shall be independent from the special characteristics of a certain fieldbus system, but should also be able to describe fieldbus specific information.

Another major point is the fact of continuous and ongoing changes in the field of automation devices. This defines the demand for expandable models that can describe additional device type or vendor specific data without changes at the model or format. Thus, it should provide a method for a device information description that is as generic as possible. This aims at preferably one central, single-point description element. An object-oriented approach shall be preferred, due to its fitness towards the ends mentioned here.

The model should furthermore be able to define relationships within the described information. This will allow, for example, the mapping of configuration data to the part of the device that may be configured with them.

### 3.3. Required Information

In addition to the general requirements, the analysis of existing description formats reveals the following more specific information domains that must be covered.

The most important elements are related to a *Functional Device View*. The model shall, for example, provide elements that describe the process data of the device, or also the fact that a device may contain several function blocks or any other function-oriented device type specific element. In addition, the description of the device behavior or device business logic is required at this point. Furthermore, device status and diagnosis information need to be described here.

Elements summarized as *General Device Information* are necessary information that describes for example the device identity like device type identifier, device instance identifier, identifier for device structural parts or device vendor information. Also very important is information concerning device version control like for the device hardware or parts of the device as well as for the device software. For device ordering, it is necessary to include elements for order data description. Furthermore, additional so-called non-functional information like device certification, device description certification or references to device manuals, etc. is necessary.

As already mentioned in the generic requirements section, it is necessary that the model is able to describe relationships between the information and the *Device Structure.* Therefore, it shall contain elements that represent the physical structure of the device, e.g. for the mapping of possible hardware modules to the device slots in hardware-modular devices like Remote-I/Os. Besides the physical device structure, the model shall also describe the logical device structure in a function-oriented way. Logical structure elements are for instance entities that include the process management, communication management or function blocks.

For *Device Communication*, a generic description of the communication capabilities and a communication service description, that is as generic as possible, are necessary. Moreover, a mapping of the accessible device variables to the communication services like I/O data, parameter data and diagnosis data is required. Communication function blocks may be used here.

Necessary *Device Configuration* information for application specific device parameterization and configuration shall also be describable by the model. A mapping of the configuration information to a device part is useful, like the mapping of communication relevant information to a physical or logical communication entity.

As already described in the user requirements, it is necessary that the model is able to describe elements or parts of the *Device User Interface* or *GUI*. Basic GUI information is for example labels for the representation of the described elements in the User Interface or references to a graphical device representation for the usage in various HMI components, ranging from engineering and parameterization to operation and maintenance. Besides the GUI itself, this includes several for a description of device specific workflows (which may in other contexts be described by the term "business logic").

## 4. Development

According to the requirements, a proposal for a universal device description model is introduced here.

### 4.1. Device Data - first Classification

Inside and around field devices, a huge cloud of data exists which should be covered by the device description. Thus, a classification of the possible data is useful.

As shown in figure 2, a first rough classification could be to differentiate between data inside a device and data outside of the device that still is related to any aspect of the device. Data in the device might be for example a process value; data outside the device might be the order number, the power consumption or the documentation of the device. The distribution of the device data between "inside" and "outside" may vary. A next classification can be made by distinguishing data that is accessible over the fieldbus, which needs a mapping to accessing methods and services of the fieldbus system, and data that is not accessible over the fieldbus but fulfill any function in the device. Data that is accessible over any fieldbus services may furthermore be classified into process and control relevant data which is transferred according to a certain schedule of real-time data exchange, and data that provides additional information that is transferred besides the scheduled data transfer like diagnosis information, alarms or device configuration parameters.

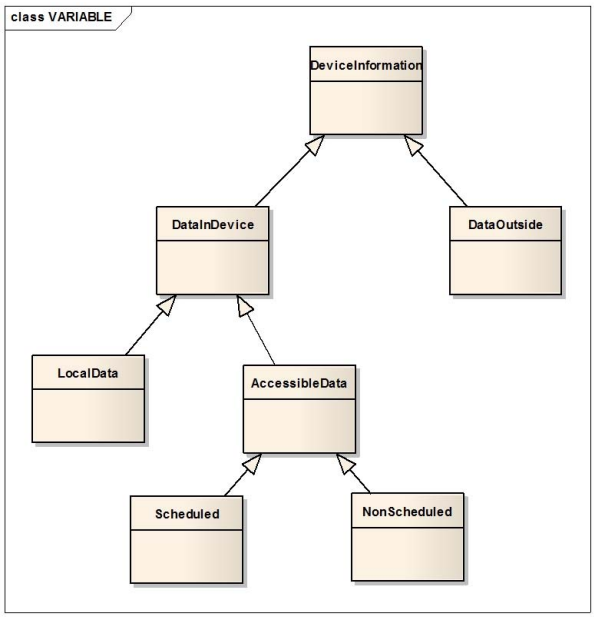


**Figure 2. Rough Classification of Existing Device Data**

### 4.2. Existing Data – Current Device Descriptions

Within the current device description models, a heterogeneous set of information is described by each standard. In the following, an analysis of the main focus of several formats is given. Each of them are deemed important and representative for device description.

With *GSD*, device type-specific information like General Information, Configuration Information, Diagnosis / Alarm Data and cyclic Data (Process Data) is describable.

The focus of *CFF* lies on type-specific and instance-specific General Information and Configuration Information.

*EDDL* describes type-specific and profile-specific information based on the generic Variable element. The class attribute of this element determines the sort of the described data like Diagnosis / Alarm Data, Local Device Data, Process Data (as for example within Function Blocks), EDD-Application Data and also Service / Maintenance Data.

*FDCML* distinguishes between General Information, Local Device Data, Process Data, Configuration Data, Device Parameter and Arguments for remote callable Methods.

### 4.3. Existing Data - "Meta Information"

A device variable can be characterized by several "Meta Information". These include for example a possible name of the data item, its data type, and further information that is required for accessing the data item via the fieldbus system.

In order to obtain the "Meta Information", the observed formats of the previous section have been analyzed. Due to the limited space, only an example will be presented here.

In *FDCML* description elements containing Meta information are for instance UniqueID, Label, Help, Description, Datatype, Refererence, Access, ArgumentType, LocalAccess, Range, Unit, and Format.

### 4.4. The Basis of the model: Variable / Device Data Item synthesis

Within figure 3, a synthesis of the Meta information from the last section is given. This structure shall be the basic structure of the universal model. Because of focusing to the most important perspective, the functional view is the main view. It considers the main device functions implemented by algorithm and operating on device variables. Thus, the core of the new model is the "*Variable"* element. This element and its optional child elements (see table 1) are intended to describe all relevant device information. The Meta information of the device variable is explained in table 2 and 3. Table 2 contains the description of the Variable Attributes and table 3 contains the description of the associated elements.

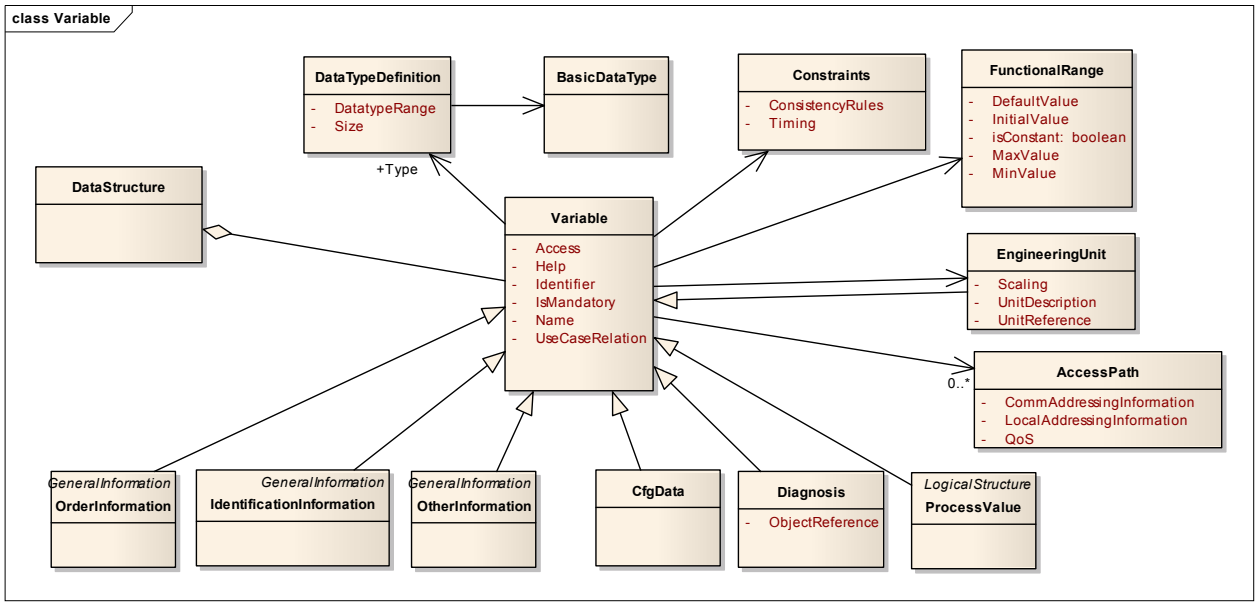


**Figure 3. Basic elements of the Universal Model**

| Element | Description |
|---|---|
| CfgData | This element shall be used to describe the device configuration information |
| Diagnosis | This element shall be used to describe Diagnosis and Alarm information. Alarm information is a special case of diagnosis information with a special communication type within the Access-Path |
| ProcessValue | This element shall be used to describe process values |
| Engineering-Unit | This is an Engineering Unit for a process value |
| Order-Information | These are special information for device ordering |
| Identification-Information | These are information for device identification and version control |

**Table 1. Child elements of Variable.**

| Attribute | Description |
|---|---|
| Access | Determines if and how a variable may be accessed over the fieldbus; possible values might be for example: read, write, read/write ,local, external |
| Help | Further textual description of the Variable |
| Identifier | Unique identifier of the variable (for example for referencing within the description) |
| IsMandatory | Determines if a Variable shall be contained in a device type e.g. for description of device profiles |
| Name | Name of the variable (might be for example used in a GUI as a label of the data item) |
| Size | Size of the variable (e.g. in byte) |
| Type | Data type of the variable |
| UseCase-Relation | Determines for which use cases the Variable is used for (for example ordering, maintenance or engineering.) |

**Table 2. Attributes of Variable.**

| Element | Description |
|---|---|
| DataType | The type attribute of the variable shall have a reference to a data type definition. This may be a device specific data type or for example also a basic data type (e.g. from IEC 61158 or IEC 61131 etc.) |
| Constraints | This element might for example contain consistency information, information about the Variable timing or if it has another dependency |
| Functional-Range | This represents a possible range of the variable like a min and max value, also a default value, initial value or also the information if a variable is constant |
| Engineering-Unit | This is the possible engineering unit of the variable (if applicable) |
| AccessPath | This elements contains information how to access the variable over the fieldbus like addressing information, the type of transaction (e.g. cyclic, acyclic, alarm, etc) |

**Table 3. Associated Variable information.**

### 4.5. Existing Data – Relationships

Between the described device data and the existing device structure relations are defined within the analyzed description models. Since all these existing models are widely used and thus cannot be neglected, the content they are describing needs to be related to the new approach. This leads to the definition of relationships as mentioned below.

*62390*

- Parameter: Mapping to Functional Element or the whole Device

*GSD*

- Diagnosis: Mapping to Device, Module, Channel
- CfgData: Mapping to Device, Module
- ProcessData: Mapping to Module

*CFF*

- General information: Mapping to related logical device part

*EDDL*

- Variable: Mapping to a Block or Menu

*FDCML*

- CfgData: Mapping to related Device Part
- ProcessData: Mapping to (logical / physical) Channel
- Argument: Mapping to ProcessData / Cfg. Data etc.
- Device Parameter: Mapping to Communication Services and thus to logical / physical Communication Interfaces

### 4.6. Structure of the universal model: Relations Synthesis

*Device:*

As shown in figure 4, a device has a physical and a logical structure. The physical structure shall represent the hardware parts of a device and the logical structure shall represent functional oriented logical (software) structure of a device.

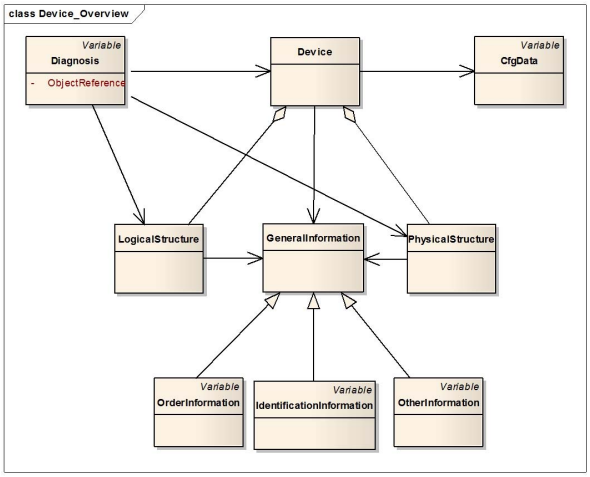


**Figure 4. Basic device structure of the Universal Model**

The device and also its structural parts have related identification, order information, and may contain additional information parts (“OtherInformation”). There might also be Diagnosis or Alarm information that is related to the device or also to its parts.

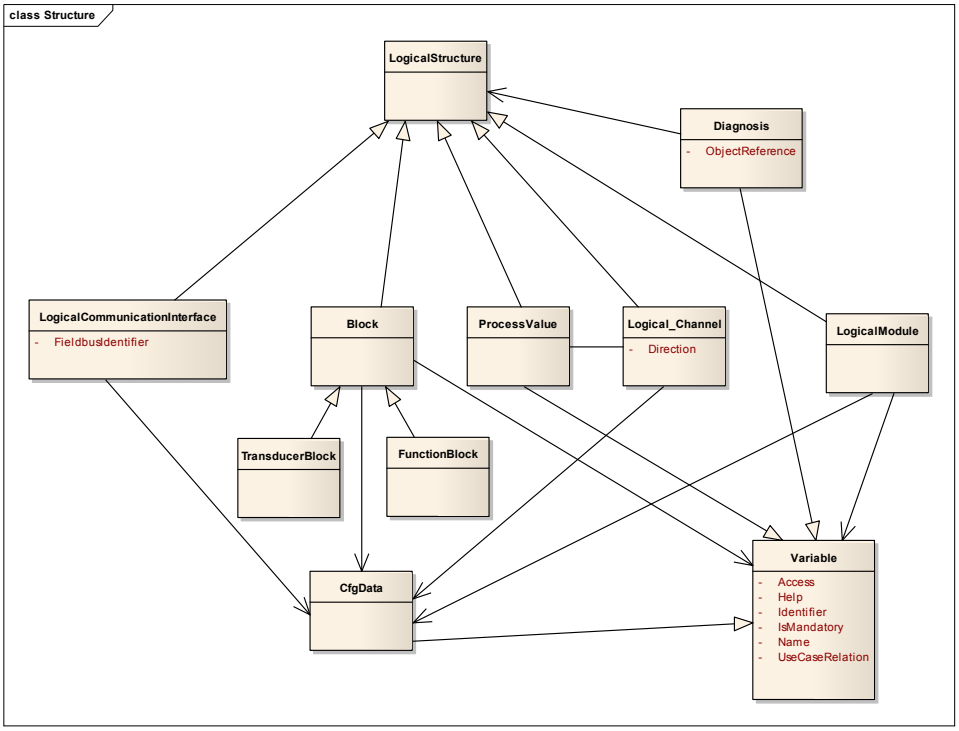


**Figure 5. Logical device structure**

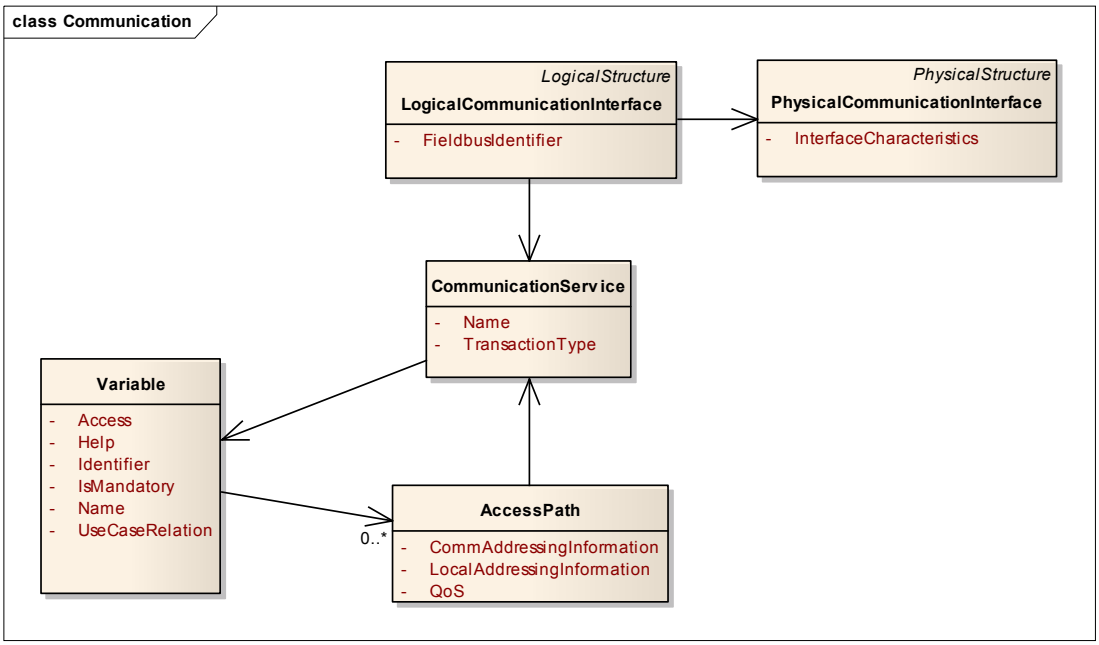


**Figure 6. Communication view**

*Physical Structure*

From a hardware point of view, a device consists of several parts. A physically modular device like a Remote I/O has slots where physical modules can be attached, which may have also an internal logical structure. A device has also physical communication interfaces and physical connections to the process. Additional physical elements might be easily added as a child element of the "PhysicalStructure" element.

Physical elements may have related diagnosis information or configuration parameters.

*Logical Structure*

In figure 5, possible elements for logical and functional device structuring are given. From this point of view, a device (or also a physical module) may consist of several Block elements, logical channels, logical modules, or also process values, which are always related to a logical channel which again is related to a physical channel. For every logical element, related diagnosis and configuration information may exist.

*Communication*

As already mentioned above, a device has a logical communication interface, which is related to a physical communication interface. Through this interface, communication services are executed. Parts of each such service (and its description) are the references to those variables that it transfers. Furthermore, service- or service-class-specific data are provided here. This information is important for network design and configuration.

## 5. Proof-of-Concept

By the developed universal model, information and their relations become visible which have been invisible so far because they were distributed in different formats or hidden in background knowledge. The model makes them available for the usage throughout the life cycle. Thus, these relations may be created and used in engineering tools, but also for operation, and in maintenance applications. Furthermore, the relations of the information fragments are already visible in the device definition at the vendor, or during specification of application profiles.

The way the model was developed, by an analysis of state-of-the-art of description formats and the synthesis of the thereby obtained information, the universal model covers all information that previously was distributed among the different formats.

To show the correctness of the model, a comparison of the description of existing devices has been done as shown in the following example. A PROFIBUS PA device typically contains a large number of variables that determine the function of the device. Most of these variables are mapped to a description expressed in EDDL. This description focuses on parameterization and on operation and is relevant for application relations between the device and a configuration tool or a visualization system. The logical structure of the device can be separated into different blocks. The actual configuration of the blocks is set by writing appropriate data into selected parameters.

Additional descriptions of the same device are required in a controller, e.g. a PLC or a DCS system.

These descriptions are needed to configure the structure of the runtime data. A GSD file contains this description. Within GSD, configuration data are defined, representing the device in a modular structure, even if the device is a compact one. These logical modules are defined with respect to the runtime data only, the have no relation to the logical structure of device described in EDDL. No consistency mechanisms are existing guaranteeing the logical structure defined by parameterization using EDD is matching the configuration data expressed in GSD.

In the model described in the paper, this relation is clearly defined. Thus, the parameters have to be defined only once, a definition tool can guarantee consistency. Such a tool was developed as a prototype.

The tools used during engineering can also refer to such a relation and can implement functions for an automatic pre-selection of the corresponding configuration data. So the possible sources for errors are reduced.

Another example has been investigated from a device vendor's point of view. If a vendor wants to support more than one fieldbus, he needs to create, deliver and maintain the appropriate fieldbus-specific descriptions (like GSD for PROFIBUS, GSDML for PROFINET, EDS for CANopen, EDS for DeviceNet, FDCML for INTERBUS-S etc.). This is a tremendous effort for a manufacturer, even if the data are very similar. Without a general model, definitions have to be done separately and repeated without any chance of re-using previously made specifications.

The model described above contains such a relation. Thus, definitions made once can be re-used and can be quite easy transformed into different formats. This improves consistency again, reducing efforts for function-related software tools (that are independent from communication). Within the context of profile definitions, such transformations have been developed and have been applied. They show the feasibility of the workflow described above.

## 6. Conclusion and Future work

In this paper, a generic device description model has been presented. The model is based on a survey of existing device description formats and their underlying models. The object-oriented approach which was used for the model allows the later adaptation and extension of this generic model.

The first goal that can be achieved by employing this model is of course the generation of a new format for device descriptions that is based off this model. The idea would be to have a very universal format, since the underlying model did take a lot of experience from other formats into account. This, however, is not possible without a lever that will lift this format above all the other formats on the market.

Another advantage that can be taken of this model is the usage in semantic device description. In order to define semantics for automation devices, a common information model is a very useful standard. Based off this model, more sophisticated descriptions including an ontology for field devices could be defined. This can then be used in current attempts towards automatic configuration or even firmware generation of field devices [8].

Last but not least, the definition of a generic device description model allows experts and designers from the industry to have a common frame of reference that covers all the different interests that arise over the whole life cycle of a field device. This can facilitate discussions, ease data exchange between different expert groups and therefore form a unified understanding of information inside and around field devices.

**References**


[1] John, D.; Topp, U.; Lin, Y.; Fay, A.: Universal Device Model for the whole lifecycle - Concept and realization with OPC UA. In: *Automation technology in practice (atp)* 49 (2007), Nr. 7, S. 45–51

[2] Braune, A.; Wollschlaeger, M.: XML in der Automation: Jedem seine Sprache? *Computer&Automation*, 9/2007, WEKA Fachzeitschriften-Verlag GmbH Poing, ISSN 1615-8512, S. 32-38. *(german)*

[3] Schwibach, M.; Pelz, M.: EDD and FDT/DTM - wide or narrow differences. In: *Automation technology in practice (atp)* 2 (2006), S. 52–55

[4] The VAN consortium: Engineering of VAN platform for Embedded Automation Systems - Task T8.3 Engineering Related Product Data Descriptions / European Community - "Information Society Technology" Programme. 2002-2006.

[5] Mühlhause, M.; Diedrich, C.; Riedl, M.: Integration von Planungs- und Instrumentierungsdaten in den operativen Betrieb. *Inproceedings*: *Automation* 2008 *(german)*

[6] Schumann, Thilo: Bus-independent device description. *Inproceedings*: *SPS/IPC/DRIVES* 2008

[7] NAMUR - User Association of Process Control Technology in Chemical and Pharmaceutical Industries, NAMUR-Workgroup 2.6 "Fieldbus": NE105 – Specifications for Integrating Fieldbus Devices in Engineering Tools for Field Devices, August 2004

[8] Hahn, C.; Gössling, A.; Frenzel, R.; Wollschlaeger, M.: Towards an Automated Generation of Device Firmware Components for Intelligent Field Devices. *Inproceedings*: *International Conference on Industrial Technology – IEEE ICIT* 2009