# 57/614/CDV

**COMMITTEE DRAFT FOR VOTE (CDV)**
**PROJET DE COMITÉ POUR VOTE (CDV)**

| | | |
|---|---|---|
| | Project number<br>Numéro de projet | IEC 61850-6 ed. 1 |
| IEC/TC or SC: **57**<br>CEI/CE ou SC: | Date of circulation<br>Date de diffusion<br>**2002-11-15** | Closing date for voting (Voting mandatory for P-members)<br>Date de clôture du vote (Vote obligatoire pour les membres (P))<br>**2003-04-18** |
| Titre du CE/SC: | TC/SC Title: Power system control and associated communications | |
| Secretary: Dr. Andreas Huber (Germany)<br>Secrétaire: | | |
| Also of interest to the following committees<br>Intéresse également les comités suivants<br>IEC TC 13, IEC TC 65, IEC TC 95 | Supersedes document<br>Remplace le document<br>57/569/CD - 57/581/CC | |

Functions concerned
Fonctions concernées

| | | | |
|---|---|---|---|
| ☐ Safety<br>Sécurité | ☐ EMC<br>CEM | ☐ Environment<br>Environnement | ☐ Quality assurance<br>Assurance qualité |

Titre :

Title : Communication networks and systems in substations - Part 6: Configuration description language for communication in electrical substations related to IEDs

Note d'introduction

Introductory note

| ATTENTION | ATTENTION |
|---|---|
| **CDV soumis en parallèle au vote (CEI) et à l'enquête (CENELEC)** | **Parallel IEC CDV/CENELEC Enquiry** |

Draft IEC 61850-6

# Communication Networks and Systems in Substations

Part 6: Configuration description language for communication in electrical substations related to IEDs

## CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

___________

# COMMUNICATION NETWORKS AND SYSTEMS IN SUBSTATIONS

# Part 6: Configuration description language for communication in electrical substations related to IEDs

## FOREWORD

1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, the IEC publishes International Standards. Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. The IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of the IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested National Committees.
3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical reports or guides and they are accepted by the National Committees in that sense.
4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.
5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.
6) Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

The International Standard IEC 61850-6 has been prepared by WG 10, 11, and 12 of IEC technical committee 57: Communication networks and systems in substations.

## INTRODUCTION

Part 6 of this standard specifies a description language for configurations of electrical substation IEDs. This language is called **S**ubstation **C**onfiguration description **L**anguage (SCL). It is used to describe IED configurations and communication systems according to parts 5 and 7-x of this standard. It allows the formal description of the relations between the substation automation system and the substation (switchyard). At the application level the switchyard topology itself and the relation of the switchyard structure to the SAS functions (logical nodes) configured on the IEDs can be described.

SCL allows the description of an IED configuration to be passed to a communication and application system engineering tool, and to pass back the whole system configuration description to the IED configuration tool in a compatible way. Its main purpose is to allow the interoperable exchange of communication system configuration data between an IED configuration tool and a system configuration tool of different manufacturers.

The parts 8-x and 9-x of this standard, which are concerned with mappings of parts 7-x to specific communication stacks, may extend these definitions according to their need with additional parts, or just by restrictions on the way the values of objects have to be used.

The relative position of this part of the standard in the context of other parts is depicted in Figure 1.

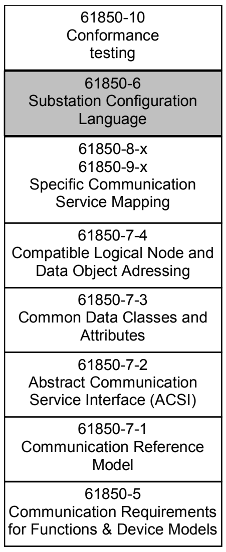

61850-10 Conformance testing
61850-6 Substation Configuration Language
61850-8-x 61850-9-x Specific Communication Service Mapping
61850-7-4 Compatible Logical Node and Data Object Adressing
61850-7-3 Common Data Classes and Attributes
61850-7-2 Abstract Communication Service Interface (ACSI)
61850-7-1 Communication Reference Model
61850-5 Communication Requirements for Functions & Device Models

**Figure 1 - Relative position of this part within IEC 61850**

# COMMUNICATION NETWORKS AND SYSTEMS IN SUBSTATIONS

# Part 6: Configuration description language for communication in electrical substations related to IEDs

## 1 Scope

This part 6 of IEC 61850 specifies a file format for describing communication related IED configurations and IED parameters, communication system configurations, switchyard (function) structures, and the relations between them. The main purpose of this format is to exchange IED capability descriptions, and SA system descriptions between IED engineering tools and the system engineering tool(s) of different manufacturers in a compatible way.

The defined language is called substation configuration description language (SCL). The IED and communication system model in SCL is according to parts 5 and 7-x of this standard. SCSM specific extensions or usage rules may be required in the appropriate parts.

The configuration language is based on the Extensible Markup Language (XML) version 1.0.

This standard does not specify individual implementations or products using the language, nor does it constrain the implementation of entities and interfaces within a computer system. This part of the standard does not specify the download format of configuration data to an IED, although it could be used for part of the configuration data.

## 2 Normative References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All normative documents are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

W3C, Extensible Markup Language (XML) 1.0, http://www.w3.org/TR/2000/REC-xml-20001006

W3C, Name spaces in XML, http://www.w3.org/TR/1999/REC-xml-names-19990114

W3C, XML Schema Part 0: Primer, http://www.w3.org/TR/2001/REC-xmlschema-0-20010502

W3C, XML Schema Part 1: Structures, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502

W3C, XML Schema Part 2: Data Types, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

RFC 1952, GZIP file format specification version 4.3, http://www.ietf.org/rfc/rfc1952.txt

IEC 61850-5, Communication networks and systems in substations – Part 5: Communication Requirements for functions and device models

IEC 61850-7-1, Communication networks and systems in substations – Part Basic communication structure for substations and feeder equipment – Principles and models

IEC 61850-7-2, Communication networks and systems in substations – Part Basic communication structure for substations and feeder equipment – Abstract communication service interface(ACSI)

IEC 61850-7-3, Communication networks and systems in substations – Part Basic communication structure for substations and feeder equipment – Common data classes

IEC 61850-7-4, Communication networks and systems in substations – Part Basic communication structure for substations and feeder equipment – Compatible logical node classes and data classes

IEC 61850-8-1 Communication networks and systems in substations – Part 8-1: Specific communication service mapping (SCSM) – Mappings to MMS (ISO/IEC 9506 Part 1 and Part 2) and to ISO/IEC 8802-3

IEC 61850-9-1: Communication networks and systems in substations – Part 9-1: Specific communication service mapping (SCSM) – Sampled values over serial unidirectional multidrop point-to-point link

IEC 61850-9-2: Communication networks and systems in substations – Part 9-2: Specific communication service mapping (SCSM) – Sampled values over ISO/IEC 8802-3

IEC 61346-1: 1996, Industrial systems, Installation and equipment and industrial products – Structuring principles and reference designations – Part 1: Basic rules

## 3 Definitions

In general the glossary defined in part 2 of this standard apply. The following definitions are used in this part additionally or with some added semantics:

### 3.1

**Access point**

a communication access point to an IED. This may be a serial port, an Ethernet connection, or a client or server address dependent on the used stack. Each access point of an IED to a communication bus is uniquely identified. Each server has exactly one (logical) access point.

### 3.2

**Subnetwork**

a communication system connection between IEDs with serial communication facilities. All devices connected to a subnetwork can directly communicate to each other, without an intervening router. Routers or gateways can connect subnetworks.

### 3.3

**Device**

any physical device, not only IEDs as in the other parts of IEC 61850. In the context of a switchyard these are primary apparatuses like transformers and circuit breakers, in the context of substation automation all IEDs.

### 3.4

**Connectivity node (ConNode)**

a connectivity node is a connection point between terminals of primary devices which has the only task to connect them electrically with no resistance; e.g. a bus bar as connectivity node connects bus bar disconnectors. The connection to a device is done at a device terminal. A connectivity node can connect an arbitrary number of terminals (devices).

### 3.5

**Subdevice**

a part of a PrimaryDevice, which might especially be a phase of a three-phase device

## 4 Abbreviations

In general the glossary and abbreviations defined in part 2 of this standard apply. The following abbreviations are particularly useful for understanding this part and are repeated here for convenience.

| | |
|---|---|
| CIM | Common Information Model of IEC 61970-301 |
| DO | DATA in IEC61850-7-2, Data object class or instance, depending on context |
| DTD | Document Type Definition |
| SCL | Substation Configuration description Language |
| UML | Unified Modelling Language according to Booch / Jacobson / Rumbaugh |
| URI | Universal Resource Identifier |
| XML | Extensible Markup Language |

## 5 Intended engineering process with SCL

Engineering of a Substation Automation system may start either with the allocation of functionally pre-configured devices to switchyard parts / products / functions, or with the design of the process functionality, where the allocation of functions to physical devices is made later based on functional capabilities of devices and their configuration capabilities. Often a mixed approach is preferred: a typical process part such as a line bay is pre-engineered, and then the result is used within the process functionality as often as needed. For SCL this means, that it must be capable of describing:

1. Pre-configured IEDs with a fixed number of logical nodes (LNs), but with no binding to a specific process - only may be a relation to a very general process function part.

2. Pre-configured IEDs with a pre-configured semantic for a process part of a certain structure, e.g. a double busbar GIS line feeder.

3. Complete process configuration with all IEDs bound to individual process functions and primary equipment, enhanced by the access control object definitions (access allowances) for all possible clients.

4. As item 3 above, but additionally all predefined associations and client server connections between logical nodes on data level. This is needed if an IED is not capable of dynamically building associations or reporting connections (either on client or on server side).

The last case is the complete case. Both case 3 and 4 are the result after SAS engineering, while cases 1 and 2 are possible results after IED pre-engineering.

The scope of SCL as defined in this document is clearly restricted to these purposes:

1. IED capability description (points 1 and 2 above)
2. System description (points 3 and 4 above)

for the purpose of **system** communication engineering and description of the engineered system communication for the device engineering tools in a standardised way.

The resulting object model however could also be the base for other engineering tasks, either directly, or with some additions. Therefore, and because of additional needs of SCSMs, this standard considers the language as defined here as the core model, and defines, how extensions of this core model for SCSMs as well as other (engineering) purposes can be done in a standardised way.

Figure 2 explains the usage of SCL data exchange in the above mentioned engineering process. The grey shaded boxes above the dashed line indicate where SCL files are used. The box *IED capabilities* corresponds to a result of steps 1 and 2 above, the other box to steps 3 respective 4 above.

The IED configurator is a manufacturer specific tool that shall be able to import / export data (e.g. files) defined by this part of the standard. It provides IED specific settings and generates IED specific configuration files, or it loads the IED configuration into the IED.

An IED shall only be considered compatible in the sense of this standard series, if

- It is accompanied either by an SCL file describing its capabilities, or by a tool, which can generate this file from the IED.

- It can use a system SCL file to set its communication configuration, as far as setting is possible at all (i.e. as a minimum its needed addresses), ot it is accompanied by a tool which can import a system SCL file to set these parameters to the IED.

The System configurator is an IED independent system level tool that shall be able to import / export data (e.g. files) defined by this part of the standard. It imports configuration files from several IEDs, as needed for system level engineering, and used by the configuration engineer to add system information shared by different IEDs. Then the system configurator generates a substation related configuration file as defined by this part of the standard, which may be fed back to the IED configurator for system related IED configuration.

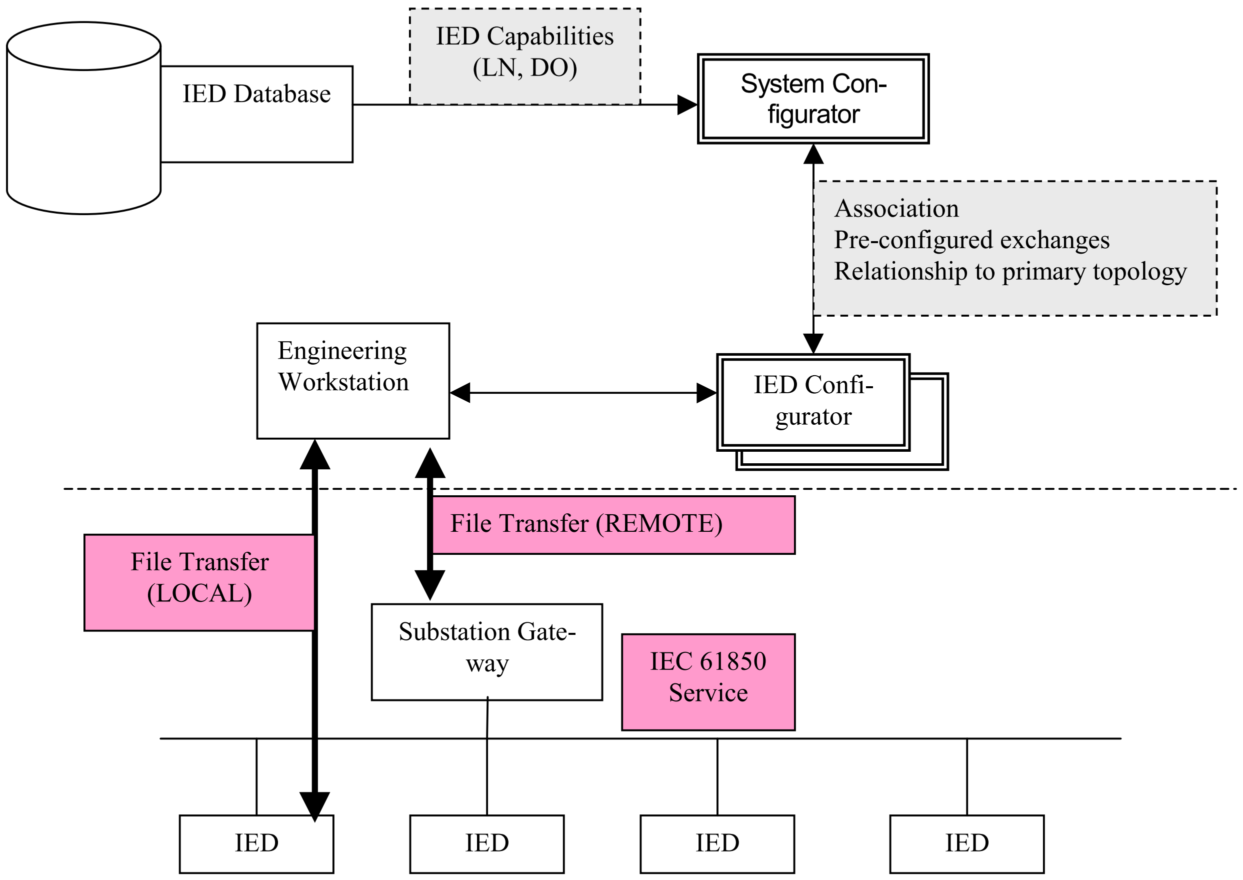


**Figure 2 - Reference model for information flow during the configuration process**

The part below the dashed line of

Figure 2 indicates the ways in which IED configuration data produced by means of the IED configurator can be brought into the IED. This can be done

- By local file transfer from an engineering workstation connected locally to the IED. This file transfer is outside the scope of this standard.

- By remote file transfer e.g. by the file transfer method of this standard. The file format is not defined within this standard, but naturally SCL format is a possible choice.

- By access services to parameter and configuration data defined according to this standard. In this case the standardised methods according to part 7 shall be used.

## 6 The SCL object model

The SCL in its full scope describes a model of

- The primary system structure: which primary apparatus functions are used, and how the apparatuses are connected. This results in a designation of all covered switchgear as substation automation functions, structured according to IEC 61346-1.

- The communication system: how IEDs are connected to subnetworks and networks, and at which of their communication access points (communication ports).

- The application level communication: How data is grouped into data sets for sending, how IEDs trigger the sending and which service they choose, which input data from other IEDs is needed.

- Each IED: which logical devices are configured on each IED, which logical nodes of which class and type belong to each logical device, and which reports with which data and which (pre-configured) associations are available; which data shall be logged.

- Instantiable logical node (LN) type definitions. The logical nodes as defined in part 7 have mandatory, optional and user defined (external) data (DO) as well as optional services, and are therefore not instantiable. In this document instantiable LNTypes are defined, which contain as template the really available DOs and services.

- The relations between instantiated logical nodes and their hosting IEDs on one side and the switchyard (function) parts on the other side.

NOTE*:* SCL allows the specification of user defined DOs as extension of standard LN classes as well as completely user defined LNs according to the rules of parts 7-4. This means, that the appropriate name space attributes shall be defined in the logical node types.

An SCL file describes an instance of the model in a special way. However its semantic can only be fully understood by reference to the model itself. This clause therefore describes the model by using UML notation. The next clauses then define how an instance of the model is formally described in SCL.

The UML object model is contained in figure 3. Note that it is not complete from the modelling sense, i.e. it does not show any superclasses from which the used classes may be derived. It restricts to those concrete object types that are used within SCL, in case of the substation structure mainly for the purpose of functional designation. Further it does not contain the levels below Data (DOs), which are defined e.g. in 7-2.

The object model has three basic object levels:

- Substation: the switchyard devices (process devices), their connection on single line level (topology), and the designation of devices and functions are described. The designations are constructed according to the functional structure of IEC 61346.

- Product: this stands for all SAS product-related objects like IEDs and logical nodes.

- Communication: this contains communication related object types like subnetworks and communication access points, and describes the communication connections between IEDs respective indirectly between logical nodes as clients and servers.

Additionally the logical node type section allows specifying in a type oriented (i.e. reusable) way which data and attributes are really existing on an IED. A logical node type is an instantiable template of the data of a logical node.

More model details contained in SCL, e.g. the structure within the logical nodes, are described in parts 7-x.

The substation object level and the product object level in itself form hierarchies, which are used for naming and can be mapped to the functional and product structures according to IEC 61346. The communication object level model just contains the communication connection relations of devices to subnetworks, between subnetworks by means of routers, and the placement of master clocks at the subnetworks for time synchronisation. The modelling of gateways is not considered here – it is a local issue of the appropriate gateway logical node. Nevertheless, it is recommended that a gateway from another protocol to IEC61850 models all IEDs below as IEC61850 logical devices.

As can be easily seen from Figure 3, the logical node (abbreviated as LN or LNode) is the transition object, which is used to connect the different structures. This means that the LN as a product also has a functional aspect within the switchyard functionality and a communication aspect within the substation automation system.

The substation functional objects as well as the product related objects are hierarchically structured. Each higher level object consists of lower level objects. This hierarchy is reflected in the designation structure of the objects according to IEC 61346-1. The function structure of IEC 61346-1 shall be used for the substation objects, while its product structure should be used for IED designation and structure.

In SCL it is foreseen that within each structure two kinds of designation are possible:

- A **name** is used as (hierarchical part of) a technical key to designate the object. Each object within a hierarchy has an attribute *Name*, which contains its reference part within the hierarchy. Technical keys are used in technical documentation for building and maintaining the system, or for automatic processing of engineering related information. This designation is also used in SCL to describe links between different model objects. In this case, as far as possible the attribute containing the link gets a name of the form *<Targettype>Name,* e.g. *DOName* for a link to a Data object. This *Name* relates to and mostly is identical to what is called *Name* in part 7-2.

- A **description** part is used as (hierarchical part of) an operator related object identification. An object within a hierarchy has an attribute *Desc*, which contains its textual description part within the hierarchy. Textual identifications are for example used in operator interfaces and operator manuals.

A reference (*Ref*) within SCL is, as defined in 7-2, a unique identification of an object, containing as a path the concatenation of all names in the hierarchy levels above until the level of the object. For forming names, also the term *Instance* with abbreviation *Inst* is used. It is a part of a name, making it unique within this level (see examples later on).

The following clauses describe the different parts of the model, their meaning and respective usage. Object attributes are mentioned here only if necessary for the understanding of the model. Further object attributes are described later in the SCL definition. Further model details belonging to parts 7-x and especially explained in 7-1 and 7-2 are purposely not shown here. The name model of the switch yard functionality however is only found in this part, and therefore shown as far as used within this part.

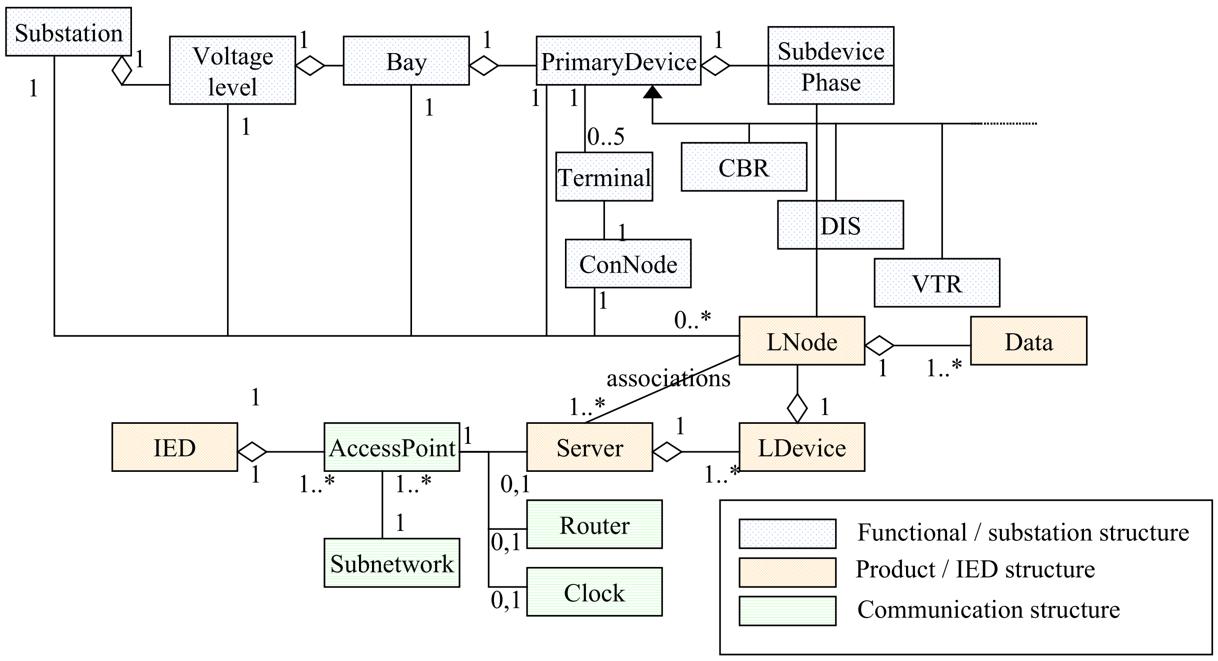


**Figure 3 - SCL object model**

The following Figure 4 shows an instance of this model: a simple example of a SA system used for a switchyard. The naming is performed according to IEC 61346. The switchyard has a 110kV voltage level E1. It is a double bus bar system with two line bays =E1Q1 and =E1Q3, and a bus coupler =E1Q2. The IEDs are already assigned to switchyard functionality (e.g. the bay controller -E1Q1SB1 as a product is assigned to bay =E1Q1, and its LN CSWI1 controls the circuit breaker =E1Q1QA1 via the LN XCBR1 on the IED -E1Q1QA1B1). Observe that in IEC 61346 terms here the bay is a transition object, i.e. it has a function (= sign, at switchyard level), and it is considered to be as product a part of the switch yard. Figure 4 shows with the – (Minus) sign only the product-related designation. The functional name (at switchyard level) is not repeated. The station level communication subnetwork is named W1. There are additional three subnetworks at process level (W2, W3, W4). Access points are seen in the picture, but their designations are not shown. Also logical devices and servers are not shown in the picture. This means especially that dynamic connections like associations are not shown.

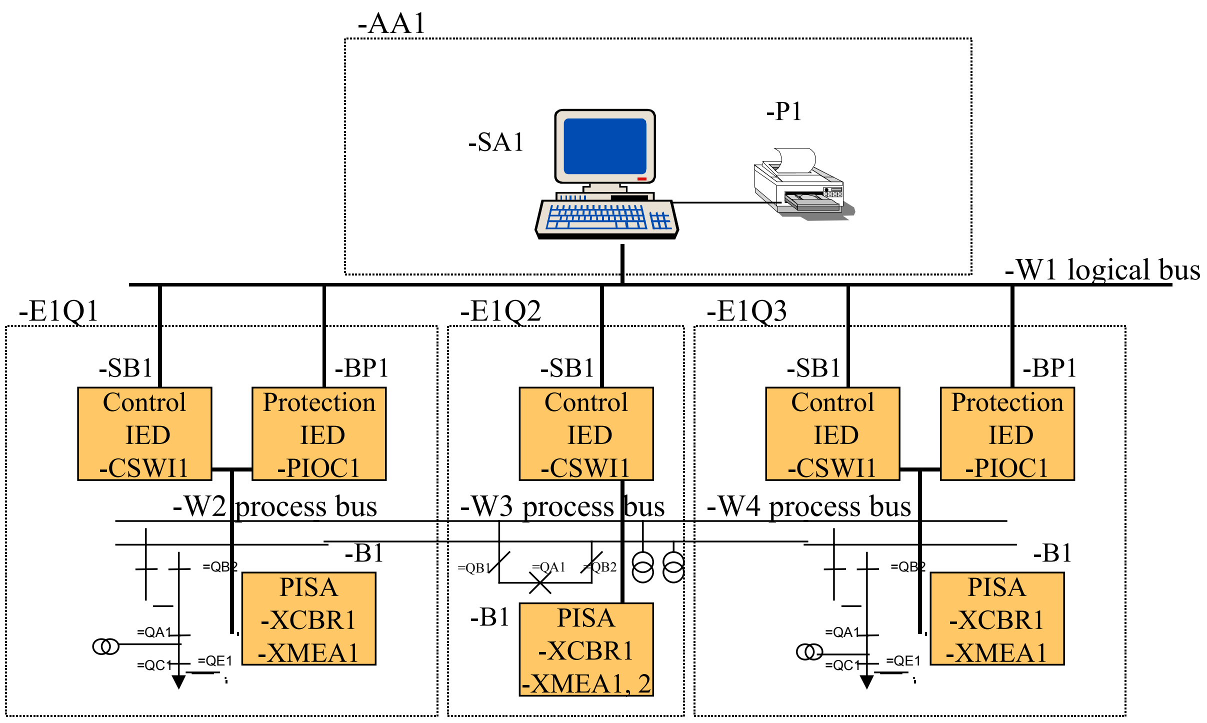


**Figure 4 - Example configuration**

## 6.1 The substation model

The substation model (upper part of Figure 3) is an object hierarchy based on the functional structure of the substation. Although each object is self-contained, its reference designation is derived from its place in the hierarchy. Because LNs perform functions within the complete context of the substation, they can be attached as lower level functional objects on each substation function level. Typically a switch controller LN is attached to a switching device, while a measuring LN is attached to the bay which delivers the measurands.

The purpose of the substation model is

- To relate a logical node to a substation function (substation part or primary device or subdevice)
- To derive a functional designation for the logical node from the substation structure

The following substation objects of the functional structure (in hierarchical order) are used in the SCL model. More background information on these words can be found in the glossary [IEC 61850-2]:

| | |
|---|---|
| Substation: | the object identifying a whole substation. |
| Voltage level: | an identifiable substation part having an identical voltage. If this name structure part is not needed (e.g. for a transformer, or a non electrical part of the substation), then an empty designation string (name) for this level is used, and no voltage value specified. |
| Bay: | an identifiable part or subfunction of the switchyard (substation) within one voltage level. Especially the transformer is a bay, which may either be con- |

| | |
|---|---|
| | tained completely in one voltage level, or preferably outside a voltage level, i.e. within a voltage level with empty designation string and no voltage specified. |
| Device | an apparatus within the switchyard, e.g. circuit breaker, disconnector, voltage transformer, power transformer winding etc. The single line diagram of a switchyard shows the electrical connection of these primary devices. Connectivity node objects model these connections. Therefore, each primary device can contain via terminals links to the connectivity nodes to which it is connected. At single line level normally one or two terminals (connections) are sufficient. |
| Subdevice | a part of a primary Device, which might especially be a phase of a three-phase device. |
| ConNode | the (electrical) connectivity node object connecting different primary devices. Typical connectivity node examples are: connecting nodes within a bay (BayNode), bus bars connecting several bays in the same voltage level, lines connecting bays in different substations. See also Device above. |

**NOTE**: Observe that the hierarchical structure is mainly used for functional designations. In case that substructures of bays are needed, this can be introduced by appropriate bay names. IF e.g. a bay B1 shall be structured into sub-bays SB1 and SB2, this would in the SCL model lead to two bays named B1.SB1 and B1.SB2. If logical nodes shall be attached also to the B1 structure level, then B1 can be introduced as a third bay.

## 6.2 The product (IED) model

Products consisting of hardware or software implement the functions of the switchyard. The scope of SCL from the product side only covers the hardware devices (called IEDs) that form the substation automation system, and therefore restrict the model to them. Primary devices as products are outside the scope of SCL, only their functional side is modelled by the substation structure for functional naming purpose.

| | |
|---|---|
| IED | a substation automation device performing SA functions by means of LNs. It is normally communicating via a communication system with other IEDs in the SAS. Access point objects form the connection between an IED and sub-networks (communication buses) in the communication system. |
| Server | a communication entity within an IED according to part 7 of this standard. It allows access via the communication system and its only access point to the data of the logical devices and logical nodes contained in the server. |
| LDevice | a logical device (LD) according to part 7-2 of this standard, that is contained in a server of an IED. |
| LNode | a logical node (LN) according to parts 5 and 7-2 of this standard contained in a logical device of an IED. The LN contains Data (DO), which other Logical Nodes request, and it may need DOs contained in other LNs to perform its function. The *offered DOs* (server capability) are described in SCL. The *needed DOs* (LN client side) are related to the function implementation and therefore determined by the IED configuration tool respective the engineer. SCL allows optionally to describe them, so that a data flow on data level between LNs can be modelled. |
| DO | the DATA contained in the LNs according to part 7 of this standard. |

This part of the standard introduces additionally

- a *Router* function on an IED. This is a function of the communication network, therefore it is described in the next subclause.

- a *Clock* function to indicate where a subnetwork master clock is located.

## 6.3 The Communication system model

The communication model is, in contrast to the others, not a hierarchical model. It models the logically possible connections between IEDs at and across subnetworks by means of access points. A subnetwork is seen on this description level only as a connecting node between access points, not as a physical structure. A logical device of an IED is connected to a subnetwork by means of an access point, which may be a physical port or a logical address (server) of the IED. Client LNs use the address attribute of the access point to build up associations to servers on other IEDs containing logical devices respective to their contained LNs.

Although subnetworks only model logically possible connections, a correlation to the physical structure can be built up by appropriate naming of subnetworks and access points, and by the relation of access points to (one or more) physical connection points. The access points are the matching elements (transition objects) of both this communication model and the physical implementation of the communication system. The description and maintenance of the physical structure is out of the scope of the core SCL.

| | |
|---|---|
| Subnetwork | a connecting node for direct (link layer) communication between access points. It might contain routing on bridge level, but not on network level. All access points connected to a subnetwork can communicate with all others on the same subnetwork with the same protocol. SCSMs may define restrictions to this e.g. if the stack implements a master-slave bus. The subnetwork as used here is a logical concept. Several logical subnetworks with different higher layer protocol could e.g. be used on the same physical bus to allow mixing of higher level protocols on the same physical (lower) layer(s). |
| Access point | a communication access point of the logical device(s) of an IED to a subnetwork. There is at most one connection between a logical device and a subnetwork on this logical modelling level. An access point may, however, serve several logical devices, and the logical nodes contained in a logical device may, as clients, use several access points to connect to different subnetworks. Typically a switch controller LN may get data as a client from a process bus (part 9-x of this standard), and provide data as a server to the inter bay bus (part 8-1 of this standard). In the terminology of part 7 an access point may be used by a server, a client, or by both. Further the same (logical) access point might support different physical access ports, e.g. an Ethernet connection and a serial PPP based connection to the same higher level (TCP/IP) access point and same server. |
| Router | Normally Clients connected to a subnetwork have only access to servers connected to that subnetwork. The router function extends access to servers connected to another subnetwork at another access point of that IED which hosts the router function. However, a router restricts the access to those services which use a networking layer, all other services like GSE, sampled values and time synchronisation messages are not allowed to cross it. |
| Clock | a master clock at this subnetwork, which is used to synchronize the internal clocks of all (other) IEDs connected to this subnetwork. |

## 7 Configuration Description Files

Configuration description files are used to exchange the configuration data between different tools, possibly from different manufacturers. As already mentioned in section 5 (see also figure 2), there are at least two kinds of configuration files to be distinguished for the data exchange between tools. This is done by means of different file extensions. Nevertheless the contents of each file shall obey the rules of the Substation Configuration Language SCL defined in the next section. Each file should contain a version and revision number to distinguish different versions of the same file. This means that each tool has to keep the version and revision number information of the last file exported, or read back the last existing file to find out its version.

NOTE: The version identifies versions of the SCL file, not versions of the data models used within the tools. This is a private issue of the tools.

The following types of SCL files are distinguished:

- Data exchange from the IED configurator tool to the system tool (corresponding to points 1 and 2 of clause 5). This file describes the capabilities of an IED. It shall contain exactly one IED section for the IED whose capabilities are described. Further it shall contain the needed logical node type definition, and may contain an optional substation section.

  The file extension shall be **.**ICD for IED Configuration Description.

- Data exchange from the system configurator tool to IED configurator tools (corresponding to points 3 and 4 of clause 5). This file contains all IEDs, a communication configuration section and a substation description section.

  The file extension shall be **.**SCD for Substation Configuration Description.

The IED Configuration Description file again may be used in different ways.

- If the name of the IED (Ref attribute, see next clause) is left blank, then this is a file for a device type, i.e. it describes the capabilities of a certain device type, and can be used as a start of engineering for each device of this type. This file does not contain a communication system section, but may contain an optional substation section that has the substation and voltage level names (Ref attributes) undefined, i.e. set to an empty string.
- If the name of the IED is given, this refers to an instantiated IED within a project. In this case the communication section contains the current address of the IED. The substation section related to this IED may be present and have name values assigned according to the project specific names.

# 8 The SCL language

The SCL language is based on XML (see normative references).

## 8.1 Specification method

The XML language contains a method to define the allowed vocabulary of XML document types, the Document Type Definition (DTD). The later developed XML Schema is an enhancement to DTD which additionally allows to define data types and data type dependent value coding. The XML Schema language can express more details syntactically, and is therefore used here for the normative syntax definition of SCL. The appropriate XML Schema definition is enhanced by appropriate (incomplete) examples illustrating the use of the specific feature defined, and additional written requirements, restrictions, and relations to the object model, which shall be used or checked by the application reading or building an SCL file. The complete normative XML schema definition is contained in the appendix.

In the following schema definition parts it is assumed that the SCL Schema definition file starts as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--W3C Schema for IEC61850-6 (SCL)-->
<xs:schema  xmlns:xs="http://www.w3.org/2001/XMLSchema"  elementFormDe-
fault="qualified">
```

and then ends with

```
</xs:schema>
```

The basic XML syntax is itself specified in a special form of extended Backus - Naur form (EBNF). In certain cases reference is taken to elements defined there, by indicating them as XML-EBNF elements.

In some cases the data format of values is important. Wherever possible, here the data type coding (lexical presentations) of the XML Schema shall be used. If not explicitly said, all element values are XML Schema *string*s, and all attribute values are of the XML Schema type *normalizedString*, i.e. they are not allowed to contain tab, carriage return and line feed characters.  Further restrictions may be stated either in this document or in other parts of this standard, mostly parts 7, 8 and 9. If any XML schema data type is used, it is referenced with the prefix *xs:*, e.g. *xs:decimal* for decimal number coding. For convenience an overview about coding of the most types used in SCL is given in Table 2 - Data type mapping

## 8.2 Private Data

*Private* data entities appear on several levels of the SCL. The contents of these XML elements is, as seen from the SCL, transparent text (XML format CDATA). If the Private part shall contain XML data, then this has to be bracketed with **<![CDATA[** at the start, and **]]>** at the end.

Private data for different purpose can be distinguished by the value of its Type attribute.

The handling within tools shall be as follows:

The private data is owned by a tool / tool category. The owner is allowed to modify its contents, and normally is the only one able to interpret the data. All other tools which read private data have to preserve (store) its contents on SCL import, and regenerate it if an SCL file containing this part is produced / exported.

The appropriate IED Configurator tool owns all private data within IED sections. The System configurator tool owns all other private data (on top level, within the substation part, and within the communication part). Only the LNodeType section might contain Private data from the IED tool to which it belongs, as well as from the system configurator tool. Therefore Private data for different purpose here shall be distinguished by the value of its Type attribute. If manufacturers use it, this Type attribute value should start with a manufacturer specific string part.

## 8.3 SCL Language extensions

Extensions of the data model with semantically new LNs and DOs are covered by the rules stated in part 7-x for extensions, and by the SCL approach as a meta language to the data model, i.e. data model element identifications do not appear in the language syntax itself. The name scope of logical node classes, DATA and CDC attributes are described in SCL by stating the appropriate name space values within the appropriate DATA attributes.

The core SCL as defined here is designed for a specific purpose described in clause 5. It can however be used with smaller or bigger extensions like additional attributes for additional (engineering) tasks. Further it leaves some communication stack dependent definitions to the SCSMs. Therefore the following describes SCL extension possibilities.

### 8.3.1 Additional semantics to existing syntax elements

Some language elements of SCL like *Desc* and *Text* have a weakly defined semantic, some like the parameter *P* have purposely been left open. An SCSM shall or a further standard can define (additional) semantics to these elements. This can be done e.g. by defining a Type value for a P parameter with an own semantic.

### 8.3.2 Data type constraints

The usage of XML schema based data types allows already on the syntactic level to further restrict the range of some values. A restriction shall use one of the allowed subtypes of the types defined in this core language.

### 8.3.3 XML name spaces

For all tag elements (sub-)tags and attributes can be added. These shall however belong to a defined XML name space with a defined semantics for all these elements. The used name spaces shall be defined at the main tag (SCL). For private name spaces the used internal name space abbreviation shall start with the character **e**. An example of a standard extension for single line or communication diagram layouts is given in the appendix. The name space URI of this version of the core SCL (which is normally used as default name space in case additional elements are used in an SCD or ICD file) is xmlns:scl = http://www.iec.ch/61850/scl001.

All tools compliant to this standard shall be able to import an SCL file with name space definitions, at least for the core SCL as the default name space. Name spaces other than the SCL core, which are not understood by the tool, shall be ignored by it. This means especially, that an IED tool which exports data in its own name space to an ICD file, can NOT expect that this information is contained resp. preserved in a SCD file coming from the system configurator tool.

### 8.3.4 Private parts

For small manufacturer or project specific extensions the Private parts can be used (see 8.2). The advantage of private parts is that the data content is preserved at data exchange between tools.

### 8.3.5 Another XML syntax

Naturally a completely new standardised or private XML based syntax can be used to extend the SCL data model with additional objects or attributes. In this case references to the objects contained in the SCL model shall be defined in this new XML file, and the naming philosophy of this standard shall be followed to be able to identify the objects. An example for the CIM model of IEC 61970-301 is given in the appendix.

### 8.3.6 Summary: Standard conformance for extension handling

A tool claiming conformance with this part of the standard shall handle any extensions as a minimum as follows:

- Import and export the SCL core syntax as a default name space; understand all parts of the core syntax referring to the capabilities of the handled IED and the intended functionality of the tool.
- Keep all data in Private sections and Text elements from Import to Export (accept if purposely modified within the tool). Keep all data of not handled IEDs, if an SCD file is exported.
- Accept syntactically correct name space extensions on import.

## 8.4 General structure

An SCL - XML document starts with the XML-EBNF element *prolog,* and continues then with elements as defined in the following XML schema definition part. The whole SCL definition part is contained in the SCL element:

```
<xs:element name="SCL">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Header"/>
            <xs:choice maxOccurs="unbounded">
                <xs:element ref="Substation"/>
                <xs:element ref="Communication"/>
                <xs:element ref="IED"/>
                <xs:element ref="LNodeType"/>
                <xs:element ref="Enumtype"/>
            </xs:choice>
            <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

The SCL element shall contain a header section, and at least one of the sections Substation, IED, LNodeType, Communication, which are further explained below. Each of these sections except the header may appear more than once. At the end a Private section may follow which belongs to the system configurator tool.

```
<xs:element name="Private">
      <xs:complexType>
            <xs:simpleContent>
                  <xs:extension base="xs:string">
                        <xs:attribute name="Type" type="xs:string"/>
                  </xs:extension>
            </xs:simpleContent>
      </xs:complexType>
</xs:element>
```

The format of the contents of the private section is free. The Type attribute allows a possible distinction between different Private parts already at SCL level

The *prolog* shall contain the identification of the XML version to allow automatic detection of the character coding used. UTF-8 coding is the preferred coding. In case that verification against a DTD is wanted a reference to a DTD definition file can be added. The preferred way of validation is against the schema definition, which is identified by namespace and location at the SCL element tag.

An example SCL file verifiable with XML Schema as defined above looks as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<SCL xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
        xs:noNamespaceSchemaLocation="Mypath/scl-001.xsd">
    <!-- here come the Header / Substation / IED / LNodeType / Communication sections as
        defined later -->
</SCL>
```

The DOCTYPE specification shall not be used, if a schema verification can be performed by the file user. In this case the *xs:noNamespaceSchemaLocation* references the place where the schema definition file is located. A complete XML schema file for the core SCL is contained in the annex.

In case that instead of Schema verification a DTD verification shall be made, the SCL tag looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE SCL SYSTEM "Mypath/scl-001.dtd">
    <!-- contains the reference to the file with the DTD definitions – skipped if schema verifica-
        tion is usable -->
<SCL xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://www.iec.ch/61850/scl001" >
    <!-- here come the Header / Substation / IED / LNodeType / Communication sections as
        defined later -->
</SCL>
```

The DOCTYPE specification defines the DTD to be used, the xmlns – attribute defines the default name space to be SCL, but without verifying its syntax.

## 8.5 Object and Signal designation

The SCL model allows two kinds of object designation:

1. A technical key which is used on engineering drawings and for signal identifications. This is contained in attribute *Name* as identification of each object. If this value is used as reference to an object it is contained in an attribute name starting with a string denoting the reference target object type, and ending with the string "*Name*". The technical key is used within SCL for referencing other objects.

2. A user oriented textual designation. This is contained in attribute *Desc*. Attributes are not allowed to contain carriage return, line feed or tab characters.

Further a general description tag *Text* can be used to add descriptive textual data. The meaning of this data is purposely not specified further. Each tool shall preserve imported text data for export.

### 8.5.1 Object designations in an object hierarchy

In case of the hierarchically structured objects of the substation structure and the product structure, both *Name* and *Desc* attribute for each object contain only that part which identifies the object within this level of the hierarchy. Their value may be an empty string, e.g. if there is only one object within the hierarchy level, and the hierarchy level shall not be visible in the name. The full object reference consists of the concatenation of all name parts of higher hierarchy levels up to this level. It is up to the configuring engineer to ensure that the references are unique after concatenation. This shall be reached by using a designation (syntax) convention as specified in IEC 61346. This means especially, that all levels can be directly concatenated, if the higher level ends with a number and the lower level starts with an alpha character – else an intervening character, preferably a dot (.), shall be put between them. In case that the name is the empty string, no delimiting character is needed at this level. Other separation characters may be specified for name mapping in SCSMs or according to IEC61346.

Transition objects, i.e. objects appearing in more than one hierarchical structure, may be designated by several references, one in each structure. In the case of SCL this applies especially to LNs, which are in the substation functional structure as well as in the IED product structure.

### 8.5.2 Signal identifications to be used in the communication system

According to part 7 signal identifications are built from the following parts (Figure 5):

1. A User defined part identifying the logical device LD in the process (LDName).
2. A (function related) part to distinguish several LNs of the same class within the same IED / LD (LN-Prefix).
3. The standardised LN class name and the LN instance Id, which distinguishes several LNs of the same class and prefix within the same IED.
4. A signal identification consisting of data and attribute name as defined in the standard.

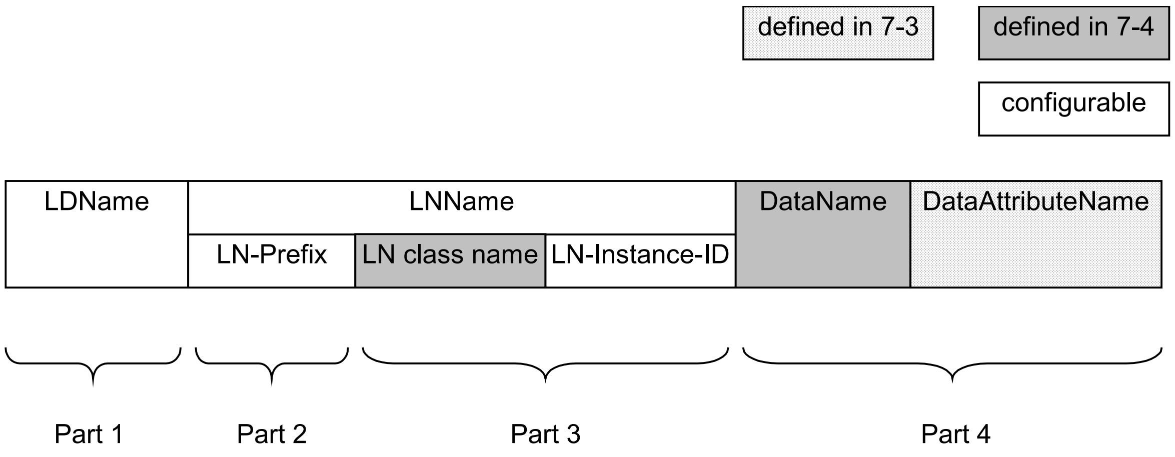


**Figure 5 – Elements of the signal identification as defined in IEC 61850-7-2**

Name parts 2 and 3 above together form the LN name and distinguish different LN instances within the same LD of an IED. Both can be used freely. A function related LN-Prefix is preferably used during functional engineering, or to bind an instantiated LN to some process semantics. The instance Id of part 3 shall be used to distinguish instantiated LNs which are not (already) bound to a process semantic (e.g. a CSWI which is not bound to some specific switch type).

The mapping of these signal name parts to actual signal names is stack and mapping related and therefore contained in parts 8 and 9 of this standard. From the SCL point of view, it is sufficient to determine the contents of these parts for a specific SAS. However, parts 8 and 9 may contain further restrictions on length and contents of name parts.

The LNodeTypes definition section of the SCL and the standardised names as defined in part 7-3 and 7-4 of this standard define the possible values for name parts 3 and 4. The LN instance Id and the prefix are defined in the IED section of the SCL.

For name parts 1 and 2 there are two options (see also Figure 6 and Figure 7). For both a separation of part two into an IED name (reference) IEDName and a LD instance name LDInst within this IED is used.

1. **Function related naming**: Part 1 is the name of the object of the substation section, to which the LN is attached. If it is a PrimaryDevice, use the name parts from Substation name to Bay name as part 1, and use the PrimaryDevice name (followed eventually by a Subdevice name) as part 2. Concatenate the IED LD Inst to part 1 (Figure 6). In case that LNs are attached to higher levels than the bay level, naturally the part 1 has to be shortened appropriately, and the part 2 stays empty, or can be used for the level where the LN is attached to.

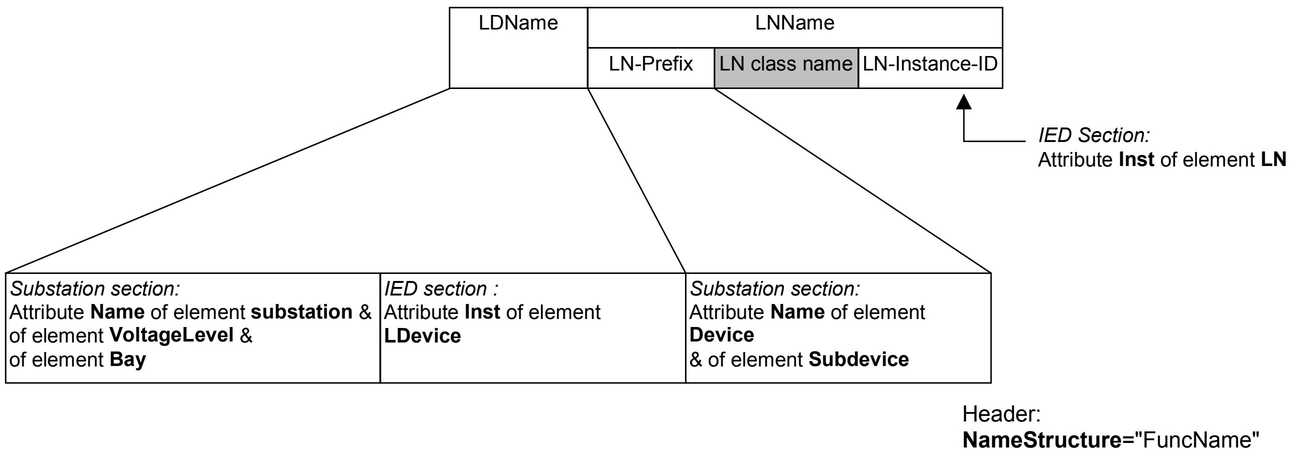

LDName
LNName
LN-Prefix
LN class name
LN-Instance-ID
*IED Section:* Attribute **Inst** of element **LN**
*Substation section:* Attribute **Name** of element **substation** & of element **VoltageLevel** & of element **Bay**
*IED section :* Attribute **Inst** of element **LDevice**
*Substation section:* Attribute **Name** of element **Device** & of element **Subdevice**
Header: **NameStructure**="FuncName"

**Figure 6 – Elements of the signal name using functional naming**

2. **Product related naming**: Part 1 is the name of the IED in the IED (product) section, on which the LN is configured, concatenated with the LD Instance Id. Part 2 stays as predefined within the IED (Figure 7).

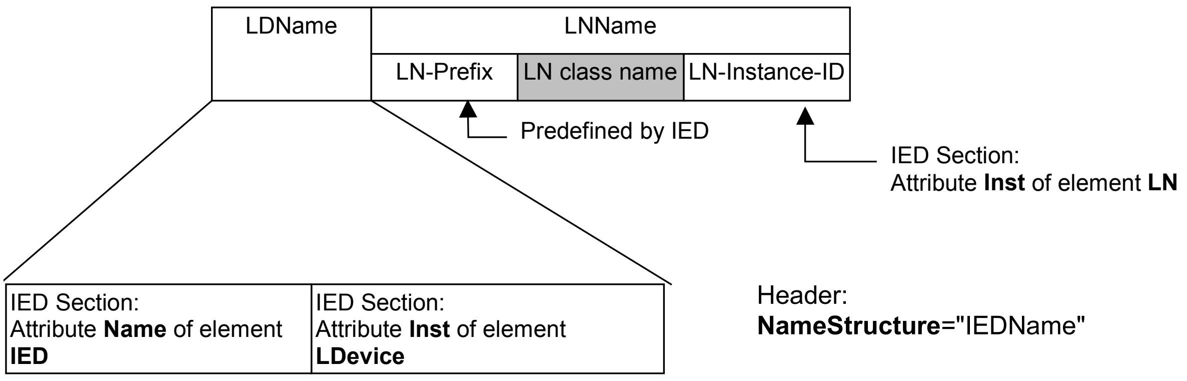

LDName
LNName
LN-Prefix
LN class name
LN-Instance-ID
Predefined by IED
IED Section: Attribute **Inst** of element **LN**
IED Section: Attribute **Name** of element **IED**
IED Section: Attribute **Inst** of element **LDevice**
Header: **NameStructure**="IEDName"

**Figure 7 – Elements of the signal name using product naming**

The SCL model leaves both options open, but allows the header part to specify if option 1 (functional naming) or option 2 (product naming) is taken for signal naming at communication time. It is recommended to use the LN instance Id in such a way that the LN class and LN instance ID together are always unique. This allows the way of naming (with / without prefix) to be changed at a later time, and even to later replace preconfigured prefixes by prefixes related to the functional structure.

### 8.5.3 Naming example

Figure 8 below shows an example of an IED with LNs, which control a circuit breaker QA1. The naming is chosen according to IEC 61346. In this example the IED as a product has the same higher level product designation part according to the bay (-E1Q1) as the controlled circuit breaker QA1 has in its functional designation (=E1Q1QA1). The figure shows the resulting references within different structures, and the resulting LN reference for communication.

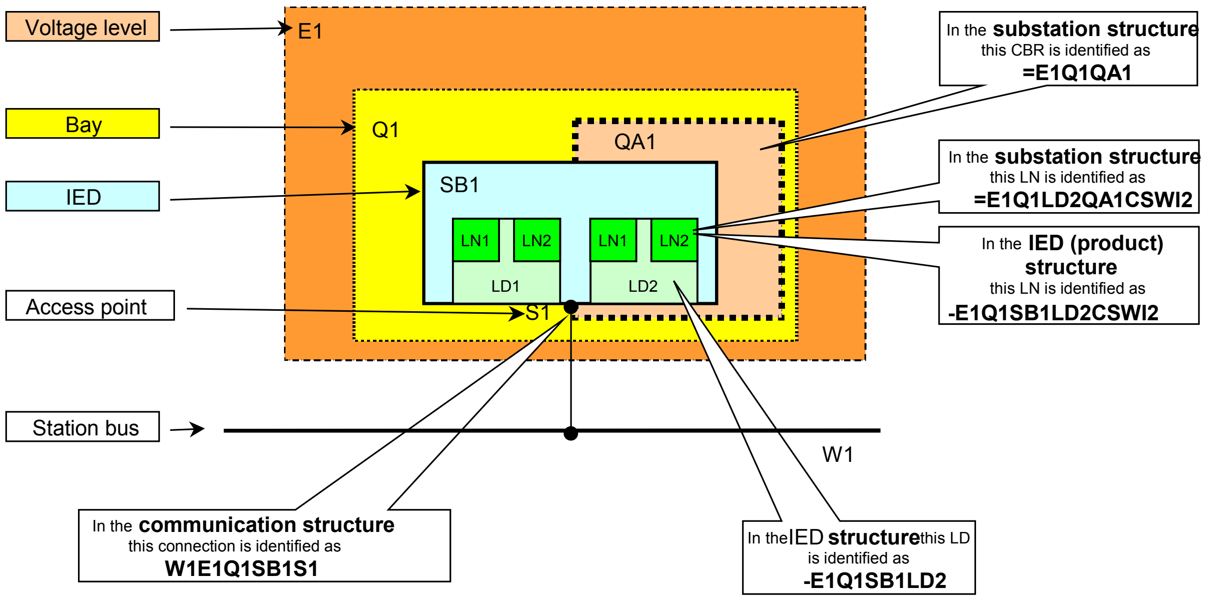

Voltage level
Bay
IED
Access point
Station bus
E1
Q1
QA1
SB1
LN1
LN2
LN1
LN2
LD1
LD2
S1
W1
In the substation structure this CBR is identified as =E1Q1QA1
In the substation structure this LN is identified as =E1Q1LD2QA1CSWI2
In the IED (product) structure this LN is identified as -E1Q1SB1LD2CSWI2
In the communication structure this connection is identified as W1E1Q1SB1S1
In the IED structure this LD is identified as -E1Q1SB1LD2

**Figure 8 - Names within different structures of the object model**

If now data objects of LN2 of LN class CSWI within LD2 shall be named with names from the function structure, then the LN reference according to part 7-2 would be E1Q1LD2/QA1CSWI2. If the reference shall be taken from the product structure, it would be E1Q1SB1LD2/CSWI2. Observe that the whole name in each case shall be unique within the system, which is the case for both names above. However, in case of the functional name the LD reference E1Q1LD2 alone is **not necessarily unique within the system** (only within the IED), because there could be another IED within bay E1Q1 having an LD2. Only the relation of E1Q1QA1CSWI2 to the IED E1Q1SB1 **in the model** allows finding the correct IED for this LD, and E1Q1LD2 then identifies uniquely the LD within this IED.

**NOTE**: if the reference is taken from the functional structure, and if there could be several IEDs within the functional part before the LD name part, it is recommended that the LDs are identified with names from the functional structure. If, for example, there are protection and control IEDs within the same bay, the LD name part could identify the protection and control subfunctionswithin the bay.

**NOTE:** If already a LN prefix is used at a preconfigured IED, then this shall always be taken as name part. In case of functional naming the engineering process has to assure that the prefix and the device / subdevice identification are identical.

# 9 The SCL syntax elements

## 9.1 Header

The header serves to identify an SCL configuration file and its version, and to specify options for the mapping of names to signals.

```
<xs:element name="Header">
      <xs:complexType>
            <xs:sequence>
                  <xs:element ref="History" minOccurs="0"/>
                  <xs:element ref="Text" minOccurs="0"/>
            </xs:sequence>
            <xs:attribute name="Ref" type="xs:string" use="required"/>
            <xs:attribute name="Version" type="xs:string"/>
            <xs:attribute name="Revision" type="xs:string" default=""/>
            <xs:attribute name="ToolId" type="xs:string" default=""/>
            <xs:attribute name="NameStructure" use="required">
                  <xs:simpleType>
                        <xs:restriction base="xs:NMTOKEN">
                              <xs:enumeration value="FuncName"/>
                              <xs:enumeration value="IEDName"/>
                        </xs:restriction>
                  </xs:simpleType>
            </xs:attribute>
      </xs:complexType>
</xs:element>
```

| | |
|---|---|
| Ref | a reference identifying this SCL file |
| Version | the version of this SCL configuration file |
| Revision | the revision of this SCL configuration file |
| ToolId | the manufacturer specific identification of the tool that was used to create the SCL file |
| NameStructure | element indicating if communication system signal names are built from the substation function structure (FuncName) or from the IED product structure (IEDName) |

NOTE: the attribute *Ref* is mandatory, while *Revision* is by default the empty string meaning the original revision of the configuration.

```
<xs:element name="Text" type="xs:string"/>
```

NOTE: the *Text* syntax element for describing text will be used in several places.

The revision history is optional. The same syntax can be used also for other documents requiring a revision history. If present, it should have the following form:

```
<xs:element name="History">
    <xs:complexType>
        <xs:sequence maxOccurs="unbounded">
            <xs:element ref="Hitem"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:element name="Hitem">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="Version" type="xs:string"
use="required"/>
```

```
              <xs:attribute name="Revision" type="xs:string"
use="required"/>
              <xs:attribute name="When" type="xs:string" use="required"/>
              <xs:attribute name="Who" type="xs:string"/>
              <xs:attribute name="What" type="xs:string"/>
              <xs:attribute name="Why" type="xs:string"/>
           </xs:extension>
       </xs:simpleContent>
    </xs:complexType>
</xs:element>
```

The History contains several history item entries. Each item identifies a (previously) approved version of this SCL file. A text within the items can be used to explain further details to this version.

| | |
|---|---|
| Version | the version of this history entry |
| Revision | the revision of this history entry |
| When | date when the version / revision was released |
| Who | who made / approved this version / revision |
| What | what has been changed since the last approval |
| Why | why the change has happened. |

The following shows a completely filled header example without history, where the signal names are taken from the substation function structure:

```
<Header Ref=”1KHL1000546” Version=”1” Revision=””
       ToolId=”MySystemTool V1.2” NameStructure=”FuncName”>My SA Project
</Header>
```

## 9.2 Substation description

The substation section serves to describe the functional structure of a substation, and to identify the primary devices and their electrical connections. For an industrial process or to describe whole power networks it is possible to have several Substation sections, one for each substation served by the SAS.

Note that the *Name* attribute is always mandatory, although it may have an empty string as a value, e.g. if there is only one object within the appropriate hierarchy level. This Name value is also a reference of the substation, because it shall be unique for all substations contained in the SCL file.

If the *Desc* attribute is missing, it gets as default value an empty string.

Logical nodes (LNode) can be attached at each level of the structure. Logical node instances at the same level shall have different identifications.

```
<xs:element name="Substation">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="VoltageLevel" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string"/>
        <xs:attribute name="Desc" type="xs:string"/>
    </xs:complexType>
</xs:element>
```

The *Private* element here and in all following hierarchical levels allow the system configurator tool to add private data to the appropriate objects. The syntax definition of *Private* is in the SCL subclause.

### 9.2.1 Voltage level

```
<xs:element name="VoltageLevel">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="Text" minOccurs="0"/>
         <xs:element ref="LNode" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Bay" maxOccurs="unbounded"/>
         <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="Name" type="xs:string"/>
      <xs:attribute name="Desc" type="xs:string"/>
      <xs:attribute name="Voltage" type="xs:string"/>
   </xs:complexType>
</xs:element>
```

The voltage level attributes are as follows

| | |
|---|---|
| Nam | a name identifying this voltage level |
| Desc | a descriptive text for this voltage level |
| Voltage | the voltage, a decimal number followed by the unit (e.g. 110kV) |

### 9.2.2 Bay level

```
<xs:element name="Bay">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="Text" minOccurs="0"/>
         <xs:element ref="LNode" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="ConNode" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Device" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Private" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="Name" type="xs:string"/>
      <xs:attribute name="Desc" type="xs:string"/>
   </xs:complexType>
</xs:element>
```

The ConNode element allows to explicitely define connectivity nodes within this bay, and also to attach logical nodes to them. Its text part can be used to contain some freely usable description. Its Name attribute identifies the ConNode instance within the bay.

```
<xs:element name="ConNode">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="Text" minOccurs="0"/>
         <xs:element ref="LNode" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="Name" type="xs:string" use="required"/>
      <xs:attribute name="Desc" type="xs:string" />
   </xs:complexType>
</xs:element>
```

**NOTE**: If a bus bar does not contain any primary devices it can be modelled as a bay that only contains connectivity nodes.

**NOTE**: A transformer is modelled as a bay. Its VoltageLevel might have the empty string as Name, if the transformer shall be outside any voltage level.

### 9.2.3 Primary device level

```
<xs:element name="Device">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="Connection" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="Subdevice" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Desc" type="xs:string" />
        <xs:attribute name="Type" type="xs:NMTOKEN" use="required"/>
    </xs:complexType>
</xs:element>
```

Connections within the SCL model the substation topology on the level of a single line, i.e. the number of phases and special connections between phases are not considered here. The maximum number of possible connections to connectivity nodes depends on the terminals available for a device function type. The following type codes for attribute *Type* are selected according to part 7-4 of this standard as far as possible:

**Table 1 Primary apparatus device type codes**

| **Type code** | **Meaning** | **Number of terminals (connections to different connectivity nodes)** |
|---|---|---|
| CBR | Circuit breaker | 2 |
| DIS | Disconnector or earthing switch | 2 |
| VTR | Voltage transformer | 1 |
| CTR | Current transformer | 2 |
| PTW | Power transformer winding | 1 |
| GEN | Generator | 1 |
| CAP | Capacitor bank | 1 / 2 |
| REA | Reactor | 1 / 2 |
| CON | Converter | 1 / 2 |
| MOT | Motor | 1 |
| EFN | Earth fault neutralizer (Peterson coil) | 1 |
| PSH | Power shunt | 2 |

| | | |
|---|---|---|
| BAT | Battery | 1 |
| BSH | Bushing | 2 |
| CAB | Power cable | 2 |
| GIL | Gas insulated line | 2 |
| LIN | Power overhead line or line segment; line segments connected by connectivity nodes form a line. A line segment within a substation could be used to attach e.g. special LNs, or physical line properties. For a GIS line segment also GIL could be used instead. | 2 |
| RRC | Rotating reactive component | 1 |
| SAR | Surge arrestor | 1 |
| TCF | Thyristor controlled frequency converter | 2 |
| TCR | Thyristor controlled reactive component | 2 |
| IFL | Infeeding line; substation limiting object; models a possibly infeeding power network line outside the substation at the single line border | 1 |

Additionally private types may be used. To allow compatibility with future enhancements of this standard, they shall start with character E.

A connection definition contains the reference to one connectivity node to which the device is connected (ConNode in our model of figure 3), and optionally the name of the device terminal which connects to this connectivity node. The reference is hierarchically structured. In SCL only the node name part CNodeName is mandatory, all other parts are optional. If they are not present they get the value of the appropriate hierarchy levels. This means that if only the node part CNodeName is defined, it is a node in the same Substation / Voltage level / Bay as the device itself.

```
<xs:element name="Connection">
    <xs:complexType>
        <xs:attribute name="CNodeName" type="xs:string" use="required"/>
        <xs:attribute name="Terminal" type="xs:string"/>
        <xs:attribute name="BayName" type="xs:string"/>
        <xs:attribute name="VLName" type="xs:string"/>
        <xs:attribute name="SSName" type="xs:string"/>
    </xs:complexType>
</xs:element>
```

The attribute meaning in detail:

| | |
|---|---|
| CNodeName | the node identification within the bay |
| Terminal | the device terminal which connects to the connectivity node |
| BayName | the bay name (e.g. to bus bar) |
| VLName | the voltage level name |
| SSName | the substation name (e.g. for lines to other substations) |

Device terminals are in general only needed if the device polarizes the power flow, i.e. the connections are not interchangeable. If the terminal attribute is left empty, but a terminal designation needed, then the default is the device identification (SSName VLName BayName DevName) together with the connectivity node identification. If the connected connectivity node is in the same bay, then CNodeName is sufficient as identification, else the terminal has to be identified additionally by the full connectivity node reference.

There is one predefined connectivity node with CNodeRef GROUNDED. This is used to model earth potential. Thus an earthing switch is an isolator (type DIS) that is connected on one side to the connectivity node GROUNDED. It is up to the tool to decide if GROUNDED is one single node for the whole substation, or a separate node at each place where connected, or something in between e.g. per bay or voltage level.

### 9.2.4 Subdevice level

Subdevices are parts of the primary devices, like a pump is part of a switch, or like a phase of a switch is a part of the whole switch. Especially they allow specifying a phase relation of LNs.

```
<xs:element name="Subdevice">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOccurs ="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOccurs ="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Name″ type="xs:string" use="required"/>
        <xs:attribute name="Desc" type="xs:string"/>
        <xs:attribute name="Phase" default="all">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="A"/>
                    <xs:enumeration value="B"/>
                    <xs:enumeration value="C"/>
                    <xs:enumeration value="N"/>
                    <xs:enumeration value="all"/>
                    <xs:enumeration value="none"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
</xs:element>
```

Attribute definition:

| | |
|---|---|
| Name | the identification of the subdevice relative to the device designation (e.g. L1, if related to phase A) |
| Desc | a textual description of the subdevice relative to the device |
| Phase | the phase to which the subdevice belongs. |

The following phase values are allowed: A, B, C, N (neutral), *all* meaning all three phases, and none (default), menaing not phase related.

### 9.2.5 Substation function logical nodes

The logical node (LN, LNode) defines the SA function part performed at the appropriate level of the hierarchy. The LNode element identifies the SA function by specifying a logical node (LN) as defined in parts 5 and 7 of this standard. *Desc* contains some operator-related text describing the LN and its usage.

```
<xs:element name="LNode">
   <xs:complexType>
      <xs:sequence minOccurs="0">
         <xs:element ref="Text"/>
      </xs:sequence>
      <xs:attribute name="Inst" type="xs:string"/>
      <xs:attribute name="Desc" type="xs:string"/>
      <xs:attribute name="LNClass" type="xs:NMTOKEN" use="required"/>
      <xs:attribute name="Prefix" type="xs:string"/>
      <xs:attribute name="LDInst" type="xs:string"/>
      <xs:attribute name="IEDName″ type="xs:string"/>
   </xs:complexType>
</xs:element>
```

The attribute definitions are

| | |
|---|---|
| Inst | the LNode instance identification |
| LNClass | the LN class as defined in parts 5 and 7 |
| IEDName | the name of the IED which contains the LN |
| LDInst | the LD instance on the IED which contains the LN |
| Prefix | the LNode prefix used in the IED (if needed) |

During functional engineering the LNode element defines a (functional) LN instance in the plant. In this case the IEDName shall be the empty string, and Inst then is an arbitrary number, as long as LNClass and Inst together are unique within the substation object level of the LN. If the allocation of logical nodes to IEDs is needed, IEDName and LDInst as a reference to the implementing IED shall be specified. At this process Inst as well as Prefix values have to be taken from the predefined values of the IED for this LN instance. If none is specified, the current values are kept.

**NOTE**: For LLN0 the value of Inst is the empty string. In all other cases it is an unsigned number.

The IEDName identifies the IED on which the LN resides, the LDInst the LD within this IED to which the LN belongs. LNClass and Inst (meaning the LN instance Id according to part 7) then identify the logical node within that IED. In case that the IED uses prefixes to identify semantic binding, also this prefix additionally to Inst identifies the LN. In this way the binding between the substation function and the SAS is defined.

### 9.2.6 Substation section example

The following incomplete SCL example contains a substation section with one bay E1Q1, which contains a circuit breaker QA1 and a disconnector QB1, both electrically connected together at connectivity node L1. The connectivity node within the same bay is explicitely defined. A logical node of type CSWI controls each switch. No link to IEDs is defined.

```
   <Substation Name="">
      <VoltageLevel Name="E1">
         <Bay Name="Q1">
            <ConNode Name="L1"/>
            <Device Name="QA1" Type="CBR">
               <LNode Inst="1" LNClass="CSWI"/>
               <Connection CNodeName="L1"/>
           </Device>
            <Device Name="QB1" Type="DIS">
               <LNode Inst="2" LNClass="CSWI"/>
               <Connection CNodeName="L1"/>
           </Device>
      </Bay>
    </VoltageLevel>
  </Substation>
```

## 9.3 IED description

The IED section describes the (pre-) configuration of an IED: its access points, the logical devices, and logical nodes instantiated on it. Further it defines the capabilities of an IED in terms of offered communication services and, together with its LNType, instantiated data (DO). There shall be an IED section for each IED. IED names (Name attribute) shall be unique within the file. In case that just the descriptions of pre-configured IEDs are contained in the file, Name shall be the empty string to indicate that the IED has not been bound to a place in the project. The system configurator tool should handle this like an IED type, i.e. a pre - configured product type, from which an arbitrary number of product (hardware) instances can be produced.

**NOTE:** Because the IEDName is unique within a system, it is also a reference.

A special IED function ***Router*** is introduced. An IED containing a router function connects different subnetworks by means of its access points. The router IED may have no logical devices and no logical nodes. In this case it is managed and supervised by a separate network management system, outside the application scope of this paper. A router is a limiting border for real time related message types, which are not allowed to cross it:

- Time synchronisation messages
- GSE messages
- Sampled measurement values

All other messages are routed through with some time delay.
Additionally to the stand-alone router IED described above, the router function can reside on an IED containing additionally clients or servers.

An access point may belong to a server with logical devices, which contain logical nodes. In this case the server of the access point provides access to the LDs and LNs, while the LNs as clients may use all IED access points (not only that of the server) to access data (on LNs on servers) on other IEDs. An access point always has a server, if the IED shall be supervised remotely, because the LN0 of the server's logical device is used to supervise the IED. Only if all LNs on an IED are using an access point as client only, and the IED shall not be supervised, then an IED without a server is needed. It is recommended that an IED contains at least one server. An access point without server may then be used to get data from 'lower level' busses, i.e. a bay unit from process bus. However, this data from the lower level bus cannot be seen directly on the higher level bus unless a router function also resides on this IED. The following figure gives a typical example of an IED connected to station bus and process bus.

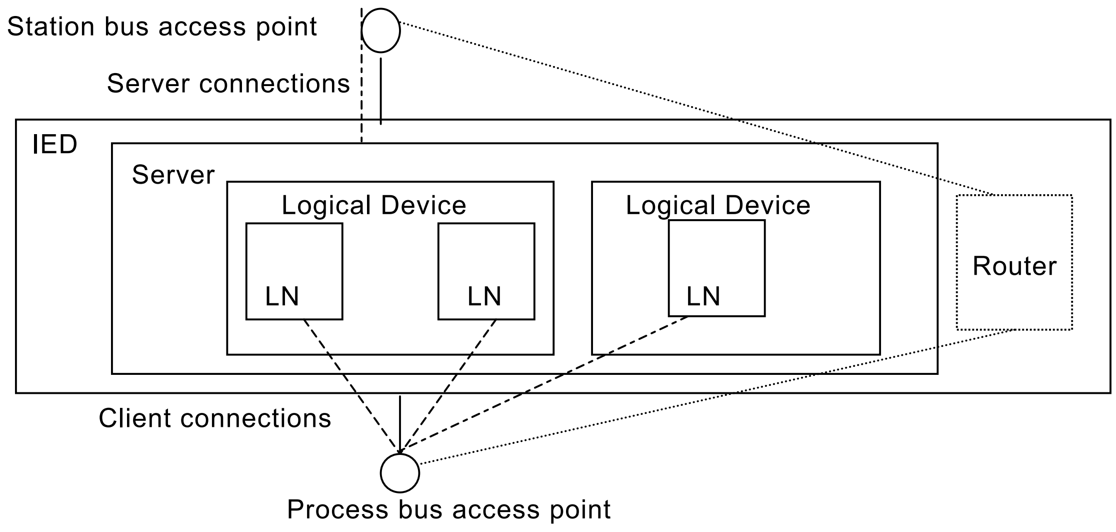


**Figure 9 - IED structure and access points**

In case that short addresses can or shall be used, it is possible to define a translation of logical names to short addresses on a data attribute basis.

**NOTE**: The usage and meaning of short addresses may be defined in an SCSM (stack mapping). In this case the system configuration tool handles them. If an SCSM does not define this, the short address related attributes may be used by the IED tools as reference to its internal addresses. In this case the IED tool handles them. In all other cases their contents shall be just imported and reexported by all tools.

More about short addresses is found in 9.5.2.1.

The SCL syntax to describe an IED is as follows:

```
<xs:element name="IED">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="Services" minOccurs="0"/>
            <xs:element ref="AccessPoint" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOccurs ="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string"/>
        <xs:attribute name="Desc" type="xs:string"/>
        <xs:attribute name="Type" type="xs:string"/>
        <xs:attribute name="Manufacturer" type="xs:string"/>
        <xs:attribute name="ConfigVersion" type="xs:string"/>
    </xs:complexType>
</xs:element>
```

The attribute definition is

| | |
|---|---|
| Name | a reference / name identifying this IED in the SAS |
| Manufacturer | the manufacturers name |
| Type | the (manufacturer specific) IED product type |
| ConfigVersion | the basic configuration version of this IED configuration. |

The IED ConfigVersion above only identifies the IED basic configuration (its capabilities), and not its individual configuration after instantiation into a project. This is a parameter of the IED, or of its logical nodes. It shall be contained in an SCL file as attribute value of the attribute *LLN0.NamPlt.configRev*.

The services element of the IED defines the available services.

```
<xs:element name="Services">
     <xs:complexType>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
               <xs:element ref="DynAssociation"/>
               <xs:element ref="GetDirectory"/>
               <xs:element ref="GetDataObjectDefinition"/>
               <xs:element ref="DataObjectDirectory"/>
               <xs:element ref="GetDataSetValue"/>
               <xs:element ref="SetDataSetValue"/>
               <xs:element ref="DataSetDirectory"/>
               <xs:element ref="ConfDataSet"/>
               <xs:element ref="DynDataSet"/>
               <xs:element ref="ReadWrite"/>
               <xs:element ref="TimerActivatedControl"/>
               <xs:element ref="DynReportControl"/>
               <xs:element ref="SetControlValue"/>
               <xs:element ref="GetControlValue"/>
               <xs:element ref="SGEdit"/>
               <xs:element ref="GSEDir"/>
               <xs:element ref="GOOSE"/>
               <xs:element ref="GSSE"/>
               <xs:element ref="FileHandling"/>
          </xs:choice>
     </xs:complexType>
```

```
</xs:element>
```

Appropriate services for control blocks shall be identically implemented for all kinds of control blocks. Observe that the service definition in an ICD file can be used to describe the maximum capabilities of a device as well as wanted restrictions (e.g. for security reasons).

Service classes may appear in arbitrary order. If they do not appear, then the services are not available at the IED. If the same service name appears several times, this is of no meaning. For the meaning of the services please refer to part 7-2.

**NOTE**: Setting group services belong to the setting group control block. If this control block is available, then also the setting group service for activating a setting group is available. If editing is possible is decided with the SGEdit element.

```
<xs:element name="SGEdit">    <xs:complexType/> </xs:element>

<xs:element name="TimerActivatedControl"> <xs:complexType/> </xs:element>
```

this element specifies that timer activated control services are supported. All other control related services are specified directly at a DO with the ctlModel attribute.

```
<xs:element name="DynAssociation"> <xs:complexType/> </xs:element>
```

all services for dynamic building of associations

```
<xs:element name="DynDataSet">         <!--Services to dynamically create and delete data sets-->
     <xs:complexType>
          <xs:attribute name="Max" type="xs:unsignedInt" use="required"/>
     </xs:complexType>
</xs:element>
```

| | |
|---|---|
| Max | the maximum number of dynamically creatable data sets |

```
<xs:element name="DynReportControl">      dynamic creation of report control blocks
     <xs:complexType>
        <xs:attribute name="Max" type="xs:unsignedInt" use="required"/>
     </xs:complexType>
</xs:element>
```

| | |
|---|---|
| Max | the maximum number of dynamically creatable report control blocks |

```
<xs:element name="GetDirectory"> <xs:complexType/> </xs:element>
```

read the server , LD and LN directories (all LDs, LNs and DATA of the LNs)

```
<xs:element  name="GetControlValue">  <xs:complexType/>  </xs:element>
```

read control block values

```
<xs:element name="GetDataObjectDefinition"> <xs:complexType/> </xs:element>

<xs:element name="GetDataSetValue"> <xs:complexType/> </xs:element>

<xs:element name="DataObjectDirectory"> <xs:complexType/> </xs:element>
```

get the DATA defined in a LN

```
<xs:element  name="SetControlValue">  <xs:complexType/>  </xs:element>
```

change control block values

```
<xs:element name="SetDataSetValue"> <xs:complexType/> </xs:element>

<xs:element name="DataSetDirectory"> <xs:complexType/> </xs:element>
```

```
<xs:element name="GSEDir"> <xs:complexType/> </xs:element>
```

GSE directory services acc. 7-2

```
<xs:element name="GOOSE">
       <xs:complexType>
            <xs:attribute name="Max" type="xs:unsignedInt"/>
       </xs:complexType>
</xs:element>
```

This element shows that the IED can be a GOOSE server and/or client acc 7-2. The Max attribute defines the maximum number of GOOSE control blocks, which are configurable. Max = 0 means the device is only a GOOSE client.

```
<xs:element name="GSSE">
       <xs:complexType>
            <xs:attribute name="Max" type="xs:unsignedInt"/>
       </xs:complexType>
</xs:element>
```

This element shows that the IED can be a binary data GSSE server and/or client acc 7-2. The Max attribute defines the maximum number of GSSE control blocks, which are configurable.

```
<xs:element name="ConfDataSet">
    <xs:complexType>
        <xs:attribute name="Max" type="xs:unsignedInt"
                           use="required"/>
        <xs:attribute name="Modify" type="xs:boolean" default="true"/>
    </xs:complexType>
</xs:element>
```

If ConfDataSet is not specified, then the value of Max is equal to the number of configured data sets, and they may be modified. If it is specified, it is possible to configure new data sets or modify existing ones at configuration time via SCL. The attribute meaning is as follows:

| | |
|---|---|
| Max | the maximum number of data sets possible |
| Modify | TRUE means that preconfigured data sets may be modified |

**NOTE:** Within an IED capability description the maximum numbers specified above must be a guarantied maximum, i.e. this number must be possible to get also if some dynamic memory allocation allows sometimes to even have more dynamic data sets at the expense of another category.

```
<xs:element name="ReadWrite"> <xs:complexType/> </xs:element>
```

basic data read and write facility; includes GetData, SetData, and the Operate service, if appropriate data objects exist.

```
<xs:element name="FileHandling"> <xs:complexType/> </xs:element>
```

all file handling services

```
<xs:element name="AccessPoint">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Router" minOccurs="0"/>
            <xs:element ref="Clock" minOccurs="0"/>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:choice minOccurs="0">
                <xs:element ref="Server"/>
                <xs:element ref="LN" maxOccurs="unbounded"/>
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:normalizedString"
                           use="required"/>
    </xs:complexType>
</xs:element>
```

a communication access point of the IED.

The Name attribute is a reference identifying this IED access point within the IED. Together with the Name of the IED this gives a unique reference for the access point within the SA system.

If neither a router, nor a clock, nor a server, nor a LN list is specified, the access point may only be used by client LNs in the same IED to access the bus to which it is connected. This is typical for a bay level device with a process bus access point, where the LNs offer their data via a server to the station bus only.

Project specific access point attributes like the address within a communication system are contained in the SCL Communication section.

```
<xs:element name="Router"> <xs:complexType/> </xs:element>
```
the presence defines this IED to have a router function.

```
<xs:element name="Clock"> <xs:complexType/> </xs:element>
```
the presence defines this IED to be a master clock at this bus

Note that

- an IED can be a router or a clock only, if it does not contain any other element (especially server),
- a router or clock function may exist additionally on a server access point,
- the IED contains only the server – this is the normal case.
- The IED contains only a LN list – these are clients only and can not be supervised, because no server offers the appropriate data. An additional router or clock function is possible.

### 9.3.1 The IED server

```
<xs:element name="Server">
     <xs:complexType>
          <xs:sequence>
               <xs:element ref="Text" minOccurs="0"/>
               <xs:element ref="Authentication" minOccurs="0"/>
               <xs:element ref="LDevice" maxOccurs="unbounded"/>
               <xs:element ref="Association" minOccurs="0" maxOc-
                                   curs="unbounded"/>
               <xs:element ref="Private" minOccurs="0" maxOc-
                                   curs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="Timout" type="xs:unsignedInt" de-
                                   fault="30"/>
     </xs:complexType>
</xs:element>
```
a communication server of the IED.

The *Timout* attribute contains a time out in seconds: if a started transaction (e.g. selection of a setting group) is not completed within this time, it is cancelled and reset.

The server is identified within the system by the access point to which it belongs. Its communication wise identification (address) is contained in the SCL communication section (9.4).

```
<xs:element name="Authentication">
     <xs:complexType>
          <xs:choice maxOccurs="unbounded">
               <xs:element ref="none"/>
               <xs:element ref="password"/>
               <xs:element ref="weak"/>
               <xs:element ref="strong"/>
               <xs:element ref="certificate"/>
          </xs:choice>
     </xs:complexType>
</xs:element>
```

The Authentication element defines in case of a device description the authentication possibilities, in case of a device instantiated in a plant the method(s) to be used for authentication. If the element is missing, this means none (i.e. no authentication). The exact meaning of the other methods, especially weak and strong, is defined in the stack mappings (SCSMs).

### 9.3.2 The logical device

The *LDevice* element defines a logical device of the IED reachable via the access point. It shall contain at least one LN and the LN0, and may contain pre - configured report, GSE and SMV definitions.

```
<xs:element name="LDevice">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LN0"/>
            <xs:element ref="LN" minOccurs="1" maxOc-
                    curs="unbounded"/>
            <xs:element ref="AccessControl" minOccurs="0"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
                    curs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Inst" type="xs:string"/>
        <xs:attribute name="Desc" type="xs:string"/>
    </xs:complexType>
</xs:element>
```

The attributes have the following meaning:

| | |
|---|---|
| Inst | The identification of the LDevice within the IED; may be an empty string if only one LDevice exists. The full LD reference according to part 7 contains an additional part before this Ref (see also chapter 8.5) |
| Desc | optional describing text for the logical device. |

### 9.3.3 LN0 and Data sets

```
<xs:element name="LN0">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="Text" minOccurs="0"/>
         <xs:element ref="DataSet" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="ReportControl" minOccurs="0" maxOc-
                        curs="unbounded"/>
         <xs:element ref="LogControl" minOccurs="0" maxOc-
                        curs="unbounded"/>
         <xs:element ref="GSEControl" minOccurs="0" maxOc-
                        curs="unbounded"/>
         <xs:element ref="SampledValueControl" minOccurs="0" maxOc-
                        curs="unbounded"/>
         <xs:element ref="SettingControl" minOccurs="0"/>
         <xs:element ref="DO" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Inputs" minOccurs="0"/>
         <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="LNType" type="xs:string" use="required"/>
   </xs:complexType>
</xs:element>
```

The LN0 attributes have the following meaning:

| | |
|---|---|
| LNType | the instantiable type definition of this logical node, reference to a LNodeType definition. |

The LN0 LNName is always LLN0, so no Inst attribute is needed. For referencing links to LN0, LNInst shall be the empty string, and LNClass shall be LLN0.

```
<xs:element name="LN">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="Text" minOccurs="0"/>
         <xs:element ref="DataSet" minOccurs="0" maxOc-
                  curs="unbounded"/>
         <xs:element ref="ReportControl" minOccurs="0" maxOc-
                  curs="unbounded"/>
         <xs:element ref="LogControl" minOccurs="0" maxOc-
                  curs="unbounded"/>
         <xs:element ref="DOI" minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Inputs" minOccurs="0"/>
         <xs:element ref="Private" minOccurs="0" maxOc-
                  curs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="Inst" type="xs:unsignedInt"/>
      <xs:attribute name="LNClass" type="xs:NMTOKEN" use="required"/>
      <xs:attribute name="Prefix" type="xs:string"/>
      <xs:attribute name="LNType" type="xs:string" use="required"/>
   </xs:complexType>
</xs:element>
```

The LN attributes have the following meaning:

| | |
|---|---|
| Inst | the LN instance id identifying this LN – an unsigned integer. |
| LNClass | the LN class according to part 7 of this standard. |
| Prefix | the LN prefix part |
| LNType | the instantiable type definition of this logical node, reference to a LNodeType element. |

The optional DOI elements in a LN definition can be used to define special instance related values for data objects and their attributes by means of DAI elements per attribute (see DOI definition in 9.3.4). The DOIs referenced here shall however already be defined within the LNodeType definition of the LN, referenced with the LNType attribute. The DOI elements at this place for this instance shall NOT define new DOs, which are not contained in the LNodeType. *Example*: the pulse length configuration parameter of a DPC CDC, specified with 100 ms in the LNodeType, is overwritten here with a value of 300 ms for this special DO.

### 9.3.4 DATA (DOI) definition

```
<xs:element name="DOI">
  <xs:complexType>
      <xs:sequence>
        <xs:element ref="DAI" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Private" minOccurs="0" maxOc-
                                  curs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="Name″ type="xs:string" use="required"/>
      <xs:attribute name="AccessControl" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

The DOI attributes are used as follows:

| | |
|---|---|
| Name | a standardised DO name e.g. from part 7-4. |
| AccessControl | access control definition for this DO. The empty string (default) means that the higher-level access control definition applies. |

The DAI attribute within the DOI defines the attributes and the related values to be set. Again, all attributes shall also be contained in the LnodeType definition of this LN. Only those are repeated here, where some aditional (attribute or element) values shall be set or individually be overwritten.

The DAI allows describing instance values for an IED. This can be used at engineering time by other IEDs / LNs which need to know configuration related values, e.g. if they have no services to read the values, or if the IED does not support their reading. Alternatively it can be used by the IED itself to set these values, either to offer them via the communication protocol, or at least consider them in its internal functions.
The DAI element contains a subset of the DA attributes, and shall be used within an IED DO specification in case that some instance specific attribute values shall be set or typical attribute values overwritten.

```
  <xs:element name="DAI">
    <xs:complexType>
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="Val"/>
      </xs:sequence>
      <xs:attribute name="SAdr" type="xs:string"/>
      <xs:attribute name="Valkind" default="Set">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="Spec"/>
            <xs:enumeration value="Conf"/>
            <xs:enumeration value="RO"/>
            <xs:enumeration value="Set"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="Name″ type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
```

The meaning of the attributes is as follows:

| | |
|---|---|
| SAdr | an optional short address of this DO attribute |
| Valkind | the meaning of the value from the engineering phases, if any value is given |
| Name | the name of the DA attribute whose value is given |

### 9.3.5 Data set definition

```
<xs:element name="DataSet">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="FCDA" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Name″ type="xs:string" use="required"/>
        <xs:attribute name="Desc" type="xs:string"/>
    </xs:complexType>
</xs:element>
```

This data set definition of the LN has the following attributes

| | |
|---|---|
| Name | a name identifying this data set in the LN where it is defined. |
| Desc | an informative text to the data set |

```
<xs:element name="FCDA">
    <xs:complexType>
        <xs:attribute name="LDInst" type="xs:string" default="" />
        <xs:attribute name="Prefix" type="xs:string" default="" />
        <xs:attribute name="LNClass" type="xs:NMTOKEN" default="" />
        <xs:attribute name="LNInst" type="xs:string" default="" />
        <xs:attribute name="DOName″ type="xs:string" default="" />
        <xs:attribute name="DAName" type="xs:string" default="" />
        <xs:attribute name="FC" type="xs:string" default="" />
    </xs:complexType>
</xs:element>
```

The FCDA element defines the name of a functionally constraint data or functionally constraint data attribute according to part 7-2 of this IED / LD to be contained in the data set. The order of the names in the data set is important for the communication messages. The element has the following attributes:

| | |
|---|---|
| LDInst | the LD where the DO resides; empty means this LD |
| Prefix | Prefix identifying together with LNInst the LN where the DO resides |
| LNInst | Instance Id of the LN where the DO resides |
| LNClass | LN class of the LN where the DO resides |
| DOName | a name identifying the DO (within the LN). A name standardized in part 7. If DOName is empty, then FC can contain a value, selecting the attribute category of all DOs of the defined LN. |
| DAName | the attribute – if empty (and *FC* empty) all attributes with characteristic *Process value (ST, MX)* are selected. |
| FC | all attributes of this functional constraint are selected. Possible constraint values see part 7-2 resp. the FC definition in 9.5. |

**NOTE:** IF Attr and FC both contain a non empty value, then the FC value shall be valid for the attribute (i.e. defined identically at the appropriate LnodeType definition), else the SCL file processing shall be stopped. If all attributes of the FCDA are empty, then this corresponds to an empty string in a GSSE DataLabel definition – in all other data sets this is not allowed.

**NOTE:** all control blocks, which reference a data set, shall be contained in the same LN as the data set definition. Therefore the data set reference within all control blocks only contains the LN relative data set name (*Name* attribute at *DatSet* element), and not its full name (which should in general also contain the LDInst and LNName according to part 7-2).

### 9.3.6 Report Control block

A report control block definition of the LN looks as follows:

```
<xs:element name="ReportControl">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="TrgOpEna" minOccurs="0"/>
            <xs:element ref="OptFlds" minOccurs="0"/>
            <xs:element ref="RptEnabled" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="RCName" type="xs:string" use="required"/>
        <xs:attribute name="RptID" type="xs:string" use="required"/>
        <xs:attribute name="DatSet" type="xs:string" use="required"/>
        <xs:attribute name="ConfRev" type="xs:unsignedInt"
                                  use="required"/>
        <xs:attribute name="Buffered" type="xs:boolean" de-
                                  fault="FALSE"/>
        <xs:attribute name="BufTim" type="xs:unsignedInt" de-
                                  fault="0"/>
        <xs:attribute name="IntgPd" type="xs:unsignedInt" de-
                                  fault="0"/>
    </xs:complexType>
</xs:element>
```

The following attributes are used:

| | |
|---|---|
| RCName | a reference identifying this report control block (part of URCName resp. BRCName in part 7-2) |
| DatSet | the name of the data set to be sent by the report control block |
| Buffered | specifies if reports are buffered or not – see part 7-2. |

| | |
|---|---|
| BufTim | buffer time – see part 7-2 |
| IntgPd | integrity period in milliseconds – see part 7-2 |

```
<xs:element name="TrgOpEna">
        <xs:complexType>
            <xs:choice maxOccurs="4">
                <xs:element ref="dchg"/>
                <xs:element ref="qchg"/>
                <xs:element ref="dupd"/>
                <xs:element ref="period"/>
            </xs:choice>
        </xs:complexType>
    </xs:element>

<xs:element name="OptFlds">
    <xs:complexType>
        <xs:choice maxOccurs="8">
            <xs:element ref="SeqNum"/>
            <xs:element ref="TimeStamp"/>
            <xs:element ref="DatSet"/>
            <xs:element ref="ReasonCode"/>
            <xs:element ref="BufOvfl"/>
            <xs:element ref="DataRef"/>
            <xs:element ref="EntryId"/>
        </xs:choice>
    </xs:complexType>
</xs:element>

<xs:element name="RptEnabled">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="ClientLN" minOccurs="0" maxOc-
                                      curs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Max" type="xs:unsignedInt" default="1"/>
    </xs:complexType>
</xs:element>
```

The RptEnabled element contains the list of client LNs for which this report is enabled (e.g. at IED startup on pre-established associations).

The attribute Max defines the maximum number of report control blocks of this type, which are instantiated at configuration time in the LN (and then used online). According to part 7-2 only one client can use a report control block at a time. This means that if Max>1 is given here, more than one report control block (RCB) of this type has to be instantiated in the IED. Observe that for all buffered control blocks a ClientLN must be preconfigured, i.e. Max is identical to the number of ClientLNs given. If ClientLNs are preconfigured for unbuffered RCBs, then the *Resv* attribute of the RCB shall be set to true additionally to the *RptEna* attribute in the IED. The URCName or BRCName of the control block according to part 7-2 consists of the RCName attribute above followed by a number between 1 and Max. If ClientLNs are defined, the index (position) of the ClientLN in the list contained in the RptEnabled element is used as this number for this client (the first has index 1). This means that a report control block definition in SCL has to be considered as a type, and not as an instance.

The ClientLN element defines the name of a LN in the system, which is a client to this report CB type.

```
<xs:element name="ClientLN">
    <xs:complexType>
       <xs:attribute name="IEDName″ type="xs:string" use="required"/>
       <xs:attribute name="LDInst" type="xs:string" use="required"/>
       <xs:attribute name="Prefix" type="xs:string"/>
       <xs:attribute name="LNInst" type="xs:string"/>
```

```
        <xs:attribute name="LNClass" type="xs:NMTOKEN" use="required"/>
    </xs:complexType>
</xs:element>
```

It has the following attributes:

| | |
|---|---|
| IEDName | the name of the IED where the LN resides |
| LDInst | the instance identification of the LD where the LN resides |
| Prefix | the LN prefix |
| LNInst | the instance id of this LN instance of below LN class in the IED. |
| LNClass | the LN class according to part 7 of this standard |

Note that to identify a LN within the system here the IED based designation is used, even if the communication name generation will be based on the substation functional structure. It is recommended that a tool assures that the defined client is really accessible across the defined communication system.

For pre-established associations the AssociationId corresponding to the referenced LN can be found in the Association definition section of this IED.

### 9.3.7 Log Control block

A log control block is defined with the following element:

```
<xs:element name="LogControl">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="TrgOpts" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="LCBName" type="xs:string" use="required"/>
        <xs:attribute name="LogEna" type="xs:boolean" default="TRUE"/>
        <xs:attribute name="IntgPd" type="xs:unsignedInt" de-
                                 fault="0"/>
        <xs:attribute name="DatSet" type="xs:string" use="required"/>
    </xs:complexType>
</xs:element>
```

The meaning of the attributes is mostly identical to the appropriate control block attributes defined in 7-2.

| | |
|---|---|
| LCBName | the name of this log control block within this LN (LCBName in 7-2) |
| LogEna | TRUE enables immediate logging; FALSE prohibits logging until enabled online. |
| IntgPd | the integrity scan period (cycle time) in milliseconds |
| DatSet | the name of the data set whose values shall be logged |

### 9.3.8 GSE control block

The following GSE control element is only allowed in the logical node LLN0.

```
<xs:element name="GSEControl">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="IEDName" minOccurs="0" maxOc-
                                 curs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="GCName" type="xs:string" use="required"/>
        <xs:attribute name="ConfRev" type="xs:unsignedInt"/>
        <xs:attribute name="Type" default="GOOSE">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="GSSE"/>
```

```
                        <xs:enumeration value="GOOSE"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="AppId" type="xs:string" use="required"/>
            <xs:attribute name="DatSet" type="xs:string" use="required"/>
        </xs:complexType>
</xs:element>
```

It has the following attributes:

| | |
|---|---|
| GCName | a reference identifying this GOOSE control block |
| Type | If format is *GSSE*, then only single indication and double indication data types are allowed for the data items referenced in the data set, else all data types are allowed. Note that on stack level each type might be mapped differently to message formats. |
| ConfRev | the configuration revision number of this control block |
| AppId | an identification of the application sending the GOOSE message in the system |
| DatSet | the name of the data set to be sent by the GSE control block. For Type=GSSE the FCDA definitions in this data set shall be interpreted as DataLabels according to 7-2. |

`<xs:element name="IEDName" type="xs:string"/>` the name of an IED in the system, to which GSE or SAMPLE data shall be sent

### 9.3.9 Sampled Value Control block

The following Sampled Value control block element is only allowed in the logical node LLN0.

```
<xs:element name="SampledValueControl">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="SmvOpts" minOccurs="0"/>
            <xs:element ref="IEDName" minOccurs="0" maxOc-
                                    curs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="SVCName" type="xs:string" use="required"/>
        <xs:attribute name="ConfRev" type="xs:unsignedInt"/>
        <xs:attribute name="SmvId" type="xs:string" use="required"/>
        <xs:attribute name="Multicast" type="xs:boolean" default="TRUE"/>
        <xs:attribute name="DatSet" type="xs:string" use="required"/>
        <xs:attribute name="SmpRate" type="xs:unsignedInt"
                                    use="required"/>
    </xs:complexType>
</xs:element>
```

The attributes mean the following:

| | |
|---|---|
| SVCName | the name of the sampled value control block within this LN |
| Multicast | FALSE indicates Unicast SMV servces only |
| ConfRev | the configuration revision number of this control block |
| SmvId | the communication level Id MsvID for the Sampled value definition for a multicast CB, the value buffer identification UsvID for a unicast CB. |
| DatSet | the name of the data set whose values shall be sent |
| SmpRate | sample rate as defined in 7-2 |

If Multicast is FALSE, i.e. this is a Unicast control block, then for each IEDName a control block instance shall be created, UsvCBName shall be set to this IEDName concatenated with SVCName above (which might then be empty), and the Resv attribute of the CB initialized to TRUE. In this case the definition has to be considered a type.

If Multicast is TRUE, then MsvCBName is set to SVCNam.

The following options can be set:

```
<xs:element name="SmvOpts">
    <xs:complexType>
        <xs:choice maxOccurs="unbounded">
            <xs:element ref="refreshtime"/>
            <xs:element ref="samplesynchronised"/>
            <xs:element ref="samplerate"/>
        </xs:choice>
    </xs:complexType>
</xs:element>
```

The meaning of the options is described in 7-2.

### 9.3.10 Setting Control Block

The following defines the definition for a setting control block. Note that the SGC name, i.e. its relative name within the LN0, is SGCB according to 7-2.

```
<xs:element name="SettingControl">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="NumOfSGs" type="xs:unsignedInt"
                                    use="required"/>
        <xs:attribute name="ActSG" type="xs:unsignedInt" default="1"/>
    </xs:complexType>
</xs:element>
```

The attributes are identical to those of the Setting group control block in 7-2.

| | |
|---|---|
| NumOfSGs | the number of setting groups available. |
| ActSG | the number of the setting group to be activated when loading the configura-tion. |

### 9.3.11 Binding to external signals

The *Inputs* section defines all external signals, which are needed by the LN logic to fulfill its function. The section also allows binding the signal to an internal address IntAdr.

```
<xs:element name="Inputs">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="ExtRef" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Each ExtRef element references one external item, either at DO or DA level. If IntAdr is needed, it has to be used appropriately.

```
<xs:element name="ExtRef">
    <xs:complexType>
        <xs:attribute name="IEDName" type="xs:string" use="required"/>
        <xs:attribute name="LDInst" type="xs:string" use="required"/>
        <xs:attribute name="Prefix" type="xs:string"/>
        <xs:attribute name="LNClass" type="xs:NMTOKEN"
                                    use="required"/>
        <xs:attribute name="LNInst" type="xs:string" use="required"/>
        <xs:attribute name="DOName″ type="xs:string" use="required"/>
        <xs:attribute name="DAName" type="xs:string"/>
        <xs:attribute name="IntAdr" type="xs:string"/>
    </xs:complexType>
</xs:element>
```

The attributes have the following meaning:

| | |
|---|---|
| IEDName | the name of the IED where the input comes from |
| LDInst | the LD instance name where the input comes from |
| Prefix | the LN prefix |
| LNClass | … etc. – identification of input item at source |
| LNInst | |
| DOName | |
| DAName | the attribute designating the input. The IED tool should use an empty value if it has some default binding (IntAdr) for all process input attributes of a DO inclusive t and q. |
| IntAdr | the internal address to which the input is bound. Only the IED tool of the concerned IED shall use the value. All other tools shall preserve it. |

An empty DAName value means the operational value attribute(s) of the DO, i.e. stVal, mag, etc. In this case also IntAdr can specify the addresses of all operational attributes in some IED tool specific way.

If the same input data can be received by the IED on different communication services (e.g. by report and by GOOSE), it is up to the IED or its tool implementation to decide which one shall be taken.

### 9.3.12 Associations

```
<xs:element name="AccessControl"> <xs:complexType mixed="true"/>  </xs:element>
```

an access control definition. Meaning and eventual refinement of the definition are stack specific issues.

It is recommended that all authorization and access control is done by private implementation within the interface LNs. In this case no access control definitions are necessary within SCL.

Each association definition defines one pre-configured association between this server and a client logical node. Two kinds of pre-configuration are possible. *Predefined* means that this association is defined, but not yet opened, the client has to open it. *Pre-established* means, that the association is defined and considered to be open directly after IED start up.

```
<xs:element name="Association">
    <xs:complexType>
        <xs:attribute name="Kind" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="pre-established"/>
                    <xs:enumeration value="predefined"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="AssociationId" type="xs:string"/>
        <xs:attribute name="IEDName″ type="xs:string" use="required"/>
        <xs:attribute name="LDInst" type="xs:string" use="required"/>
        <xs:attribute name="Prefix" type="xs:string"/>
        <xs:attribute name="LNInst" type="xs:string" use="required"/>
        <xs:attribute name="LNClass" type="xs:NMTOKEN" use="required"/>
    </xs:complexType>
</xs:element>
```

The attributes mean the following:

| | |
|---|---|
| Kind | the kind of pre-configured association, pre-established or predefined |
| IEDName | a reference identifying the IED on which the client resides |
| LDInst | the reference to the client logical device |
| LNClass | the Class of the client LN |
| Prefix | |
| LNInst | the reference number of the client LN |
| AssociationId | the Id of a pre-configured association (else empty) |

An empty association Id as given by the default value means that the association Id is not yet defined. For a completed SCL file and a pre-established association the association Id shall be set, so that the client LNs and the server can verify it correctly. The same client may use the same association to different LNs on the same server. Uniqueness requirements as well as value range of the association Id (e.g. a 32 bit integer, unique at the server, or at server IED and client Id, or system wide) are set up in the SCSMs.

## 9.4 Communication system description

This clause describes the direct communication connection possibilities between logical nodes by means of logical buses (subnetworks) and IED access points. The IED sections already describe which LDs and LNs are reachable across a certain access point. The communication section now describes which IED access points are connected to a common subnetwork. This is done in a way that reflects the hierarchical name structure within the IED, which is based on IED relative names for access points, LDs and LNs.

```
<xs:element name="Communication">
        <xs:complexType>
                <xs:sequence>
                        <xs:element ref="Subnetwork" maxOccurs="unbounded"/>
                        <xs:element ref="Private" minOccurs="0" maxOccurs ="unbounded"/>
                </xs:sequence>
        </xs:complexType>
</xs:element>

<xs:element name="Subnetwork">
        <xs:complexType>
                <xs:sequence>
                        <xs:element ref="Text" minOccurs="0"/>
                        <xs:element ref="ConnectedAP" maxOccurs="unbounded"/>
                        <xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="Name" type="xs:string" use="required"/>
                <xs:attribute name="Bitrate" type="xs:decimal"/>
                <xs:attribute name="Type" type="xs:string"/>
        </xs:complexType>
</xs:element>
```

The Text part can be used for free text.

The attributes are defined as follows:

| | |
|---|---|
| Ref | a reference identifying this bus |
| Bitrate | the bit rate used on this subnetwork in MB/s (Megabit per second) |
| Type | the bus protocol type |

Protocol types are defined in parts 8 and 9. Those of part 8 start with “8-“, those of part 9 with “9-“ (except if they are identical; in this case part 9 defines the part 8 type to be used).

```
<xs:element name="ConnectedAP">
```

```
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="Address"/>
				<xs:element ref="GSE" minOccurs="0" maxOccurs="unbounded"/>
				<xs:element ref="SMV" minOccurs="0" maxOccurs="unbounded"/>
				<xs:element ref="PhysConn" minOccurs="0" maxOccurs="unbounded"/>
				<xs:element ref="Private" minOccurs="0" maxOccurs="unbounded"/>
			</xs:sequence>
			<xs:attribute name="IEDName" type="xs:string" use="required"/>
			<xs:attribute name="APName" type="xs:string" use="required"/>
		</xs:complexType>
</xs:element>
```

The ConnectedAP is the IED access point connected to the bus.

It has the following attributes:

| | |
|---|---|
| IEDName | a reference identifying the IED |
| APName | a name identifying this access point within the IED |

```
<xs:element name="Address">
	<xs:complexType>
		<xs:sequence>
			<xs:element ref="P" minOccurs="0" maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
</xs:element>
```

The Address element defines the address parameters of this access point at this bus. The different parameters are defined within the contained P elements. The Type attribute if P identifies the meaning of the value. The meaning of the P parts is depending on the subnetwork protocol type and therefore specified in the appropriate SCSM. The Access point address shall be filled with a unique value at least for server type access points to get a complete SCD description.

```
<xs:element name="P">
	<xs:complexType>
		<xs:simpleContent>
			<xs:extension base="xs:string">
				<xs:attribute name="Type" type="xs:string" use="required"/>
			</xs:extension>
		</xs:simpleContent>
	</xs:complexType>
</xs:element>
```

The GSE element defines the address for a GSE control block in this IED. It must be at LLN0.

```
<xs:element name="GSE">
	<xs:complexType>
		<xs:attribute name="Addr" type="xs:string"/>
		<xs:attribute name="LDInst" type="xs:string"/>
		<xs:attribute name="CBName" type="xs:string"/>
		<xs:attribute name="Mintime" type="xs: unsignedInt"/>
		<xs:attribute name="Maxtime" type="xs: unsignedInt"/>
	</xs:complexType>
</xs:element>
```

The attributes have the following meaning:

| | |
|---|---|
| LDInst | the LD to which the control block belongs |

| | |
|---|---|
| CBName | the identification of the control block within the LD (always in LLN0) |
| Addr | the GOOSE multicast destination address. The format is determined by the SCSM. |
| Mintime | the maximal allowed sending delay on a data change in ms. |
| Maxtime | the source supervision repetition time in ms (supervision heart beat cycle) |

Mintime and Maxtime may influence SCSM parameters. Which and how is defined in the appropriate SCSM.

The SMV element defines the address for a sampled value control block

```
<xs:element name="SMV">
        <xs:complexType>
                <xs:attribute name="Addr" type="xs:string"/>
                <xs:attribute name="LDInst" type="xs:string"/>
                <xs:attribute name="CBName" type="xs:string"/>
        </xs:complexType>
</xs:element>
```

The attributes are defined as follows:

| | |
|---|---|
| LDInst | the LD to which the control block belongs |
| CBName | the identification of the control block within the LD (in LLN0) |
| Addr | the SMV multicast destination address |

The element PhysConn defines the type(s) of physical connection for this access point. The parameter values are depending on the type of the physical connection, and their types (meaning) have to be defined in the stack mapping.

```
<xs:element name="PhysConn">
        <xs:complexType>
                <xs:sequence>
                        <xs:element ref="P" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="Type" type="xs:string" use="required"/>
        </xs:complexType>
</xs:element>
```

The Type attribute specifies the type of physical connection of this access point to the bus, while the value then specifies the instance of this type (e.g. Type="Plug", value is "BFOC". Allowed types and values shall be defined in the stack mapping. The P element can be repeated if one value is not sufficient.

The following SCL part shows a communication section with one subnetwork XW1, to which two IEDs are connected with their access points S1. The Addresses are partly left empty, i.e. the description is not complete. The PhysConn and address types are just examples.

```
<Communication>
   <Subnetwork Ref="XW1" Type="8-MMS">Station bus
      <ConnectedAP IEDName="E1Q1KA1" APName="S1">
            <Address><P Type="IP">10.0.0.12</P></Address>
      </ConnectedAP>
      <ConnectedAP IEDName="E1Q2KA1" APName="S1">
            <Address><P Type="IP">10.0.0.23</P></Address>
            <PhysConn Type="Cable">10BaseT</PhysConn>
            <PhysConn Type="Plug">RJ45</PhysConn>
      </ConnectedAP>
   </Subnetwork>
</Communication>
```

## 9.5 LNodeType definitions

This clause defines instantiable logical node types. A logical node type is an instantiable template of the data of a logical node. It is referenced each time that this instantiable type is needed within an IED. The LN type names (Ref attribute) together with the IEDType attribute of the LN type shall be unique within each SCL file. Therefore on generation of the system SCD file from IED ICD files the names may change to keep uniqueness across all IED definitions. To keep possible semantic information of the type names, it is recommended to use the IEDType attribute to define the relation of a specific LN type to an IED type. If this is not sufficient, generate a new LN type name by concatenating the IED name (which must be unique within the file) with the old type name (which must be unique at least per IED). If a LN type shall be generally valid for several IEDs, then the IEDType attribute shall be defined as empty string.

The LN type (LNodeType element) contains a list of Data, its attributes, and possible default values for configuration parameters.

```
<xs:element name="LNodeType">
     <xs:complexType>
          <xs:sequence>
               <xs:element ref="Text" minOccurs="0"/>
               <xs:element ref="DO" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="Ref" type="xs:string" use="required"/>
          <xs:attribute name="IEDType" type="xs:string"/>
          <xs:attribute name="Desc" type="xs:string"/>
          <xs:attribute name="LNClass" type="xs:NMTOKEN" use="required"/>
     </xs:complexType>
</xs:element>
```

The attributes have the following meaning:

| | |
|---|---|
| Ref | a reference identifying this LN type within this SCL section; used by the LN attribute LNType to reference this definition |
| IEDType | the manufacturer IED type to which this LN type belongs. |
| Desc | an additional text describing this LN type |
| LNClass | the LN base class of this type |

### 9.5.1 DATA (DO) definition

```
<xs:element name="DO">
  <xs:complexType>
       <xs:sequence>
         <xs:element ref="Transient" minOccurs="0"/>
         <xs:element ref="DA" minOccurs="1" maxOccurs="unbounded"/>
         <xs:element ref="Private" minOccurs="0" maxOc-
                                  curs="unbounded"/>
       </xs:sequence>
       <xs:attribute name="Name″ type="xs:string" use="required"/>
       <xs:attribute name="CDC" type="xs:string" use="required"/>
       <xs:attribute name="AccessControl" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

The DO attributes are used as follows:

| | |
|---|---|
| Name | a standardised DO name e.g. from part 7-4. |
| CDC | the standard type (Common Data Class, CDC) of the data object e.g. as defined in part 7-3. |
| AccessControl | access control definition for this DO. The empty string (default) means that the higher-level access control definition applies. |

The existence of the *Transient* element indicates that this is transient data as defined for the appropriate DATA / DO definition. The Transient element itself is empty.

```
      <xs:element name="Transient" />
```

### 9.5.2 Data Attribute definition

Each data object has attributes related to it. The DA element defines the attributes and describes its (default) values. Each instantiable atribute except the mandatory attributes t (TimeStamp) and q (Quality) shall be defined in the LNodeType definition.

The DA element contains just a basic value, i.e. no structure values or array values can be given. In case of an attribute with a structure like *ScaledValueConfig*, for each element of the structure a DA element shall be used. Its Name attribute then contains the concatenation of attribute name and its subelement names, separated by a period. Thus the scale factor of a scaled value configuration is addressed by a name of *ScaledValueConfig.iScaleFactor*. Similarly, for an array the index in parenthesis [ ] is used. Thus the x coordinate of curve point 2 in a curve shape description DO is addressed by the name *crvPts[2].xval*. This means that all values only have a simple data type.

Part 7 defines the value type for a certain attribute based on the CDC of the DO. The value coding in the Val element then has to follow the XML Schema data type definition. The type mapping is as follows.

**Table 2 - Data type mapping**

| **Part 7 base type** | **XML Schema (xs) data type** | **Value representation** |
|---|---|---|
| INT8, INT16, INT32,<br>INT8U, INT16U, INT32U | Integer | An integer number , no decimal fraction (99999) |
| FLOAT32, FLOAT32 | Decimal | A number with or without a decimal fraction (999.99999). |
| BOOLEAN | Boolean | *false*, *true* or 0, 1 |
| ENUMERATED | NormalizedString | The enumeration element names as defined in part 7 as string values. |
| Octet string | Base64Binary | Coding according clause 6.8 of RFC 2045 |
| VisibleString | NormalizedString | A character string without tabs, linefeeds and carriage return |

The meaning of the value for an IED configuration tool can be different depending on the device capabilities, the functional characteristic of the attribute, and the stage of the engineering process. The DA attribute Valkind allows specifying this meaning. It is ignored, if no value is given, and for all cases not specified below (e.g. for the q and t attributes). The following table explains how it shall be interpreted.

**Table 3 - Attribute value kind meaning**

| Valkind value | Functional constraints | Engineering process stage | Meaning |
|---|---|---|---|
| Spec | Non operational (CF, DC) | Specification phase | The wanted value determined at specification phase typically in an SCD file |
| Conf | CF, DC, operational attribute of a CDC used for settings | IED template, after IED engineering | This value is not available at the IED. The IED is engineered such that this value is used. |
| RO | Operational process state attribute | IED template | The default value for the attribute to be used if *q.source* is set to *defaulted* – not changeable |
| RO | CF, DC, operational attribute of data used for settings | IED template, after IED configuration | Read only value at an IED – can only be set at configuration time |
| Set | CF, DC | At / after IED configuration | A determined setting value. The value is / shall be set within the IED. |
| Set | Operational process values (except time and quality) | At / after IED configuration (possibly RO changed to Set) | The default value for the operational attribute, to be used if *q.source* is set to *defaulted* |
| Set | Operational setting value (SP, SG for all data used as setting) | At / after IED configuration | The setting value for the set point |

This allows e.g. defining IED capabilities (which attributes are available, which are read only), the default values an IED is delivered with (readable, changeable, or not visible at all), or the setting values for operative (e.g. protection) parameters.

```
<xs:element name="DA">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Enum" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Val" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="SAdr" type="xs:string"/>
      <xs:attribute name="FC" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="ST"/>
            <xs:enumeration value="MX"/>
            <xs:enumeration value="CO"/>
            <xs:enumeration value="SP"/>
            <xs:enumeration value="SG"/>
            <xs:enumeration value="SE"/>
            <xs:enumeration value="SV"/>
            <xs:enumeration value="AX"/>
            <xs:enumeration value="CF"/>
            <xs:enumeration value="DC"/>
            <xs:enumeration value="EX"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="TrgOpt" default="none">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="dchg"/>
            <xs:enumeration value="qchg"/>
            <xs:enumeration value="dupd"/>
            <xs:enumeration value="none"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="BType" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="BOOL"/>
            <xs:enumeration value="INT8"/>
            <xs:enumeration value="INT16"/>
            <xs:enumeration value="INT32"/>
            <xs:enumeration value="INT8U"/>
            <xs:enumeration value="INT16U"/>
            <xs:enumeration value="INT32U"/>
            <xs:enumeration value="FLOAT32"/>
            <xs:enumeration value="FLOAT64"/>
            <xs:enumeration value="Enum"/>
            <xs:enumeration value="CodedEnum"/>
            <xs:enumeration value="VisString32"/>
            <xs:enumeration value="VisString64"/>
            <xs:enumeration value="VisString255"/>
            <xs:enumeration value="Octet64"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="Valkind" default="Set">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="Spec"/>
            <xs:enumeration value="Conf"/>
            <xs:enumeration value="RO"/>
            <xs:enumeration value="Set"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="Name″ type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
```

The meaning of the attributes is as follows:

| | |
|---|---|
| SAdr | an optional short address of this DO attribute |
| FC | specification of the functional characteristic as defined in e.g. 7-3 |
| TrgOpt | the (report or log) trigger option used for this attribute |
| BType | the basic data type of this attribute at application level (acc. 7-2) |
| Valkind | the meaning of the value from the engineering phases, if any value is given |
| Name | the name of the DA attribute whose value is given |

FC, TrgOpt and BType shall be defined. All instantiable attributes contained within a DO shall be defined.

A special BType is the enumeration (Enum and CodedEnum). In this case also the enumeration values have to be defined by means of the EnumType element (9.5.2.3).

The following example defines the stVal attribute of a DPC CDC without value, according to 7-3:

```
<DA Name=”stVal” FC=”ST” TrgOpt=”dchg” BType=”Enum” EnumType=”Position”>
```

An IED tool shall fill in the value of the functional characteristic FC when exporting an IED description. For standardised common data class attributes it is defined in part 7-3.

#### 9.5.2.1 Short addresses

The SAdr attribute allows allocating a short address to DO attributes. Short addresses can be used within the communication to be more efficient either in the communication, or in the handling of messages at client or server. Further they can be used as IED internal identification for the attribute. To be able to use short addresses in the communication,

- the stack mapping must allow them and define their meaning,
- and the IED must allow them.

The detailed syntax of a short address depends on the stack if the stack defines their usage, else on the IED tool. SCL foresees a two level hierarchy for short addresses used in communication:

- The communication address of the IED / server / access point
- The short address of a data item at attribute level

It is possible to use the short address instead of the (symbolic) IED communication address in case that the short address is unique system wide, and the stack allows this. Otherwise the short address scope is private to the IED.

#### 9.5.2.2 Values

The value definition contains a possible value. For attributes with FC = SG the SGroup attribute specifies to which setting group this value belongs. There may be a value for each setting group defined. The meaning of the value in the engineering process is defined at the DA / DAI level by means of the Valkind attribute.

```
<xs:element name="Val">
     <xs:complexType>
          <xs:simpleContent>
               <xs:extension base="xs:string">
                    <xs:attribute name="SGroup" type="xs:integer"/>
               </xs:extension>
          </xs:simpleContent>
     </xs:complexType>
</xs:element>
```

| | |
|---|---|
| SGroup | the number of the setting group (if FC is SG) to which this value belongs |

The SGroup value used within an IED should be checked against an existing setting group definition on this IED, where the maximum allowed number is specified (SettingControl.NumOfSGs). If the SGroup is left empty, then either this attribute is in no setting group (FC # SG), or the data value (if any) applies to all setting groups.

#### 9.5.2.3 Enumeration types

Enumerations are in general used in more than one LNodeType. Therefore for them an enumeration type definition is made.

```
    <xs:element name="EnumType">
       <xs:complexType>
          <xs:sequence>
             <xs:element ref="EnumVal" minOccurs="2" maxo-
curs=”unbounded"/>
          </xs:sequence>
          <xs:attribute name="Ref" type="xs:string" use="required"/>
       </xs:complexType>
    </xs:element>
```

The Ref attribute is the SCL file unique identification of the enumeration type. It is referenced by a DA definition, in case the basic data type BType is an Enum or codedEnum.

If extensions to an enumeration are made, this shall be considered as a new EnumType.

The values of the enumeration are defined as follows:

```
<xs:element name="EnumVal">
     <xs:complexType>
       <xs:simpleContent>
         <xs:extension base="xs:string">
         <xs:attribute name="Ord" type="xs:unsignedInt" use="required"/>
       </xs:extension>
     </xs:simpleContent>
    </xs:complexType>
</xs:element>
```

The Ord attribute contains the order of the values, starting from 0.

The following is an example of the EnumType definition for switch positions (attribute stVal of DPC or DPS CDCs).

```
<EnumType Ref="Position">
      <EnumVal Ord="0">intermediate-state</EnumVal>
      <EnumVal Ord="1">off</EnumVal>
      <EnumVal Ord="2">on</EnumVal>
      <EnumVal Ord="3">bad-state</EnumVal>
</EnumType>
```

## Annex A
(normative)

## SCL syntax XML Schema Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<!--W3C Schema for SCL language version 1  (CDV9) 2002-09-18 -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDe-
fault="qualified">
   <xs:element name="AccessControl">
      <xs:complexType mixed="true"/>
   </xs:element>
   <xs:element name="AccessPoint">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="Router" minOccurs="0"/>
            <xs:element ref="Clock" minOccurs="0"/>
            <xs:choice minOccurs="0">
               <xs:element ref="Server"/>
               <xs:element ref="LN" maxOccurs="unbounded"/>
            </xs:choice>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:normalizedString"
use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Address">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="P" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="Association">
      <xs:complexType>
         <xs:attribute name="Kind" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="pre-established"/>
                  <xs:enumeration value="predefined"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="AssociationId" type="xs:string"/>
         <xs:attribute name="IEDName" type="xs:string"
use="required"/>
         <xs:attribute name="LDInst" type="xs:string" use="required"/>
         <xs:attribute name="Prefix" type="xs:string"/>
         <xs:attribute name="LNInst" type="xs:string" use="required"/>
         <xs:attribute name="LNClass" type="xs:NMTOKEN"
use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Authentication">
      <xs:complexType>
         <xs:choice maxOccurs="unbounded">
            <xs:element ref="none"/>
            <xs:element ref="password"/>
            <xs:element ref="weak"/>
```

```
                <xs:element ref="strong"/>
                <xs:element ref="certificate"/>
            </xs:choice>
        </xs:complexType>
    </xs:element>
    <xs:element name="Bay">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Text" minOccurs="0"/>
                <xs:element ref="LNode" minOccurs="0" maxOc-
curs="unbounded"/>
                <xs:element ref="ConNode" minOccurs="0" maxOc-
curs="unbounded"/>
                <xs:element ref="Device" minOccurs="0" maxOc-
curs="unbounded"/>
                <xs:element ref="Private" minOccurs="0"/>
            </xs:sequence>
            <xs:attribute name="Name" type="xs:string"/>
            <xs:attribute name="Desc" type="xs:string"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="BufOvfl">
        <xs:complexType/>
    </xs:element>
    <xs:element name="ClientLN">
        <xs:complexType>
            <xs:attribute name="IEDName" type="xs:string"
use="required"/>
            <xs:attribute name="LDInst" type="xs:string" use="required"/>
            <xs:attribute name="Prefix" type="xs:string"/>
            <xs:attribute name="LNInst" type="xs:string"/>
            <xs:attribute name="LNClass" type="xs:NMTOKEN"
use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Clock">
        <xs:complexType/>
    </xs:element>
    <xs:element name="Communication">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Text" minOccurs="0"/>
                <xs:element ref="Subnetwork" maxOccurs="unbounded"/>
                <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="ConNode">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Text" minOccurs="0"/>
                <xs:element ref="LNode" minOccurs="0" maxOc-
curs="unbounded"/>
                <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="Name" type="xs:string" use="required"/>
            <xs:attribute name="Desc" type="xs:string"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="ConfDataSet">
        <xs:complexType>
```

```
         <xs:attribute name="Max" type="xs:unsignedInt"
use="required"/>
         <xs:attribute name="Modify" type="xs:boolean" de-
fault="true"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="ConnectedAP">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Address"/>
            <xs:element ref="GSE" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="SMV" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="PhysConn" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="IEDName" type="xs:string"
use="required"/>
         <xs:attribute name="APName" type="xs:string" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Connection">
      <xs:complexType>
         <xs:attribute name="CNodeName" type="xs:string"
use="required"/>
         <xs:attribute name="Terminal" type="xs:string"/>
         <xs:attribute name="BayName" type="xs:string"/>
         <xs:attribute name="VLName" type="xs:string"/>
         <xs:attribute name="SSName" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:simpleType name="ValkindType">
      <xs:restriction base="xs:NMTOKEN">
         <xs:enumeration value="Spec"/>
         <xs:enumeration value="Conf"/>
         <xs:enumeration value="RO"/>
         <xs:enumeration value="Set"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:element name="DA">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Val" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="SAdr" type="xs:string"/>
         <xs:attribute name="FC" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="ST"/>
                  <xs:enumeration value="MX"/>
                  <xs:enumeration value="CO"/>
                  <xs:enumeration value="SP"/>
                  <xs:enumeration value="SG"/>
                  <xs:enumeration value="SE"/>
                  <xs:enumeration value="SV"/>
                  <xs:enumeration value="AX"/>
                  <xs:enumeration value="CF"/>
                  <xs:enumeration value="DC"/>
                  <xs:enumeration value="EX"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="TrgOpt" default="none">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="dchg"/>
                  <xs:enumeration value="qchg"/>
                  <xs:enumeration value="dupd"/>
                  <xs:enumeration value="none"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="BType" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="BOOL"/>
                  <xs:enumeration value="INT8"/>
                  <xs:enumeration value="INT16"/>
                  <xs:enumeration value="INT32"/>
                  <xs:enumeration value="INT8U"/>
                  <xs:enumeration value="INT16U"/>
                  <xs:enumeration value="INT32U"/>
                  <xs:enumeration value="FLOAT32"/>
                  <xs:enumeration value="FLOAT64"/>
                  <xs:enumeration value="Enum"/>
                  <xs:enumeration value="CodedEnum"/>
                  <xs:enumeration value="VisString32"/>
                  <xs:enumeration value="VisString64"/>
                  <xs:enumeration value="VisString255"/>
                  <xs:enumeration value="Octet64"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="Valkind" type="ValkindType" de-
fault="Set"/>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="EnumType" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DAI">
      <xs:complexType>
         <xs:sequence minOccurs="0" maxOccurs="unbounded">
            <xs:element ref="Val"/>
         </xs:sequence>
         <xs:attribute name="SAdr" type="xs:string"/>
         <xs:attribute name="Valkind" type="ValkindType" de-
fault="Set"/>
         <xs:attribute name="Name" type="xs:string" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DO">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Transient" minOccurs="0"/>
            <xs:element ref="DA" minOccurs="1" maxOc-
curs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="Desc" type="xs:string" />
         <xs:attribute name="CDC" type="xs:string" use="required"/>
         <xs:attribute name="AccessControl" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DOI">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="DAI" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="Desc" type="xs:string"/>
         <xs:attribute name="AccessControl" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DatSet">
      <xs:complexType/>
   </xs:element>
   <xs:element name="DataObjectDirectory">
      <xs:complexType/>
   </xs:element>
   <xs:element name="DataRef">
      <xs:complexType/>
   </xs:element>
   <xs:element name="DataSet">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="FCDA" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="Desc" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DataSetDirectory">
      <xs:complexType/>
   </xs:element>
   <xs:element name="Device">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Connection" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Subdevice" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="Desc" type="xs:string"/>
         <xs:attribute name="Type" type="xs:NMTOKEN" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DynAssociation">
      <xs:complexType/>
   </xs:element>
   <xs:element name="DynDataSet">
      <xs:complexType>
         <xs:attribute name="Max" type="xs:unsignedInt"
use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DynReportControl">
```

```
      <xs:complexType>
         <xs:attribute name="Max" type="xs:unsignedInt"
use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Enhanced">
      <xs:complexType/>
   </xs:element>
   <xs:element name="EntryID">
      <xs:complexType/>
   </xs:element>
   <xs:element name="EnumVal">
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:string">
               <xs:attribute name="Ord" type="xs:unsignedInt"
use="required"/>
            </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:element name="EnumType">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="EnumVal" minOccurs="2" maxOc-
curs="unbounded" />
         </xs:sequence>
         <xs:attribute name="Ref" type="xs:string" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="ExtRef">
      <xs:complexType>
         <xs:attribute name="IEDName" type="xs:string"
use="required"/>
         <xs:attribute name="LDInst" type="xs:string" use="required"/>
         <xs:attribute name="Prefix" type="xs:string"/>
         <xs:attribute name="LNClass" type="xs:NMTOKEN"
use="required"/>
         <xs:attribute name="LNInst" type="xs:string" use="required"/>
         <xs:attribute name="DOName" type="xs:string" use="required"/>
         <xs:attribute name="DAName" type="xs:string"/>
         <xs:attribute name="IntAdr" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="FCDA">
      <xs:complexType>
         <xs:attribute name="LDInst" type="xs:string"/>
         <xs:attribute name="Prefix" type="xs:string"/>
         <xs:attribute name="LNClass" type="xs:NMTOKEN"/>
         <xs:attribute name="LNInst" type="xs:string"/>
         <xs:attribute name="DOName" type="xs:string"/>
         <xs:attribute name="DAName" type="xs:string"/>
         <xs:attribute name="FC" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="FileHandling">
      <xs:complexType/>
   </xs:element>
   <xs:element name="GOOSE">
      <xs:complexType>
         <xs:attribute name="Max" type="xs:unsignedInt"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="GSE">
```

```
      <xs:complexType>
         <xs:attribute name="Addr" type="xs:string"/>
         <xs:attribute name="LDInst" type="xs:string"/>
         <xs:attribute name="CBName" type="xs:string"/>
         <xs:attribute name="Mintime" type="xs:unsignedInt"/>
         <xs:attribute name="Maxtime" type="xs:unsignedInt"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="GSEControl">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="IEDName" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="GCName" type="xs:string" use="required"/>
         <xs:attribute name="ConfRev" type="xs:string"/>
         <xs:attribute name="Type" default="GOOSE">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="GSSE"/>
                  <xs:enumeration value="GOOSE"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="AppID" type="xs:string" use="required"/>
         <xs:attribute name="DatSet" type="xs:string" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="GSEDir">
      <xs:complexType/>
   </xs:element>
   <xs:element name="GSSE">
      <xs:complexType>
         <xs:attribute name="Max" type="xs:unsignedInt"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="GetControlValue">
      <xs:complexType/>
   </xs:element>
   <xs:element name="GetDataObjectDefinition">
      <xs:complexType/>
   </xs:element>
   <xs:element name="GetDataSetValue">
      <xs:complexType/>
   </xs:element>
   <xs:element name="GetDirectory">
      <xs:complexType/>
   </xs:element>
   <xs:element name="Header">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="History" minOccurs="0"/>
            <xs:element ref="Text" minOccurs="0"/>
         </xs:sequence>
         <xs:attribute name="Ref" type="xs:string" use="required"/>
         <xs:attribute name="Version" type="xs:string"/>
         <xs:attribute name="Revision" type="xs:string"/>
         <xs:attribute name="ToolId" type="xs:string"/>
         <xs:attribute name="NameStructure" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="FuncName"/>
                  <xs:enumeration value="IEDName"/>
```

```
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
</xs:element>
<xs:element name="History">
    <xs:complexType>
        <xs:sequence maxOccurs="unbounded">
            <xs:element ref="Hitem"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="Hitem">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="Version" type="xs:string"
use="required"/>
                <xs:attribute name="Revision" type="xs:string"
use="required"/>
                <xs:attribute name="When" type="xs:string"
use="required"/>
                <xs:attribute name="Who" type="xs:string"/>
                <xs:attribute name="What" type="xs:string"/>
                <xs:attribute name="Why" type="xs:string"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>
<xs:element name="IED">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="Services" minOccurs="0"/>
            <xs:element ref="AccessPoint" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string"/>
        <xs:attribute name="Desc" type="xs:string"/>
        <xs:attribute name="Type" type="xs:string"/>
        <xs:attribute name="Manufacturer" type="xs:string"/>
        <xs:attribute name="ConfigVersion" type="xs:string"/>
    </xs:complexType>
</xs:element>
<xs:element name="IEDName" type="xs:string"/>
<xs:element name="Inputs">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="ExtRef" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="LDevice">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LN0"/>
            <xs:element ref="LN" minOccurs="1" maxOc-
curs="unbounded"/>
            <xs:element ref="AccessControl" minOccurs="0"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
```

```
         </xs:sequence>
         <xs:attribute name="Inst" type="xs:string"/>
         <xs:attribute name="Desc" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="LN">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="DataSet" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="ReportControl" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="LogControl" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="DOI" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Inputs" minOccurs="0"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Inst" type="xs:unsignedInt"/>
         <xs:attribute name="LNClass" type="xs:NMTOKEN"
use="required"/>
         <xs:attribute name="Prefix" type="xs:string"/>
         <xs:attribute name="LNType" type="xs:string" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="LN0">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="DataSet" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="ReportControl" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="LogControl" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="GSEControl" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="SampledValueControl" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element ref="SettingControl" minOccurs="0"/>
            <xs:element ref="DOI" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Inputs" minOccurs="0"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="LNType" type="xs:string" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="LNode">
      <xs:complexType>
         <xs:sequence minOccurs="0">
            <xs:element ref="Text"/>
         </xs:sequence>
         <xs:attribute name="Inst" type="xs:string"/>
         <xs:attribute name="LNClass" type="xs:NMTOKEN"
use="required"/>
         <xs:attribute name="Prefix" type="xs:string"/>
         <xs:attribute name="LDInst" type="xs:string"/>
         <xs:attribute name="IEDName" type="xs:string"/>
      </xs:complexType>
```

```
    </xs:element>
    <xs:element name="LNodeType">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Text" minOccurs="0"/>
                <xs:element ref="DO" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="Ref" type="xs:string" use="required"/>
            <xs:attribute name="IEDType" type="xs:string"/>
            <xs:attribute name="Desc" type="xs:string"/>
            <xs:attribute name="LNClass" type="xs:string"
use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Log" type="xs:string"/>
    <xs:element name="LogControl">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Text" minOccurs="0"/>
                <xs:element ref="TrgOpEna" minOccurs="0"/>
            </xs:sequence>
            <xs:attribute name="LCBName" type="xs:string"
use="required"/>
            <xs:attribute name="LogEna" type="xs:string" default="TRUE"/>
            <xs:attribute name="IntgPd" type="xs:string" default="0"/>
            <xs:attribute name="DatSet" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="OptFlds">
        <xs:complexType>
            <xs:choice maxOccurs="8">
                <xs:element ref="SeqNum"/>
                <xs:element ref="TimeStamp"/>
                <xs:element ref="DatSet"/>
                <xs:element ref="ReasonCode"/>
                <xs:element ref="DataRef"/>
                <xs:element ref="BufOvfl"/>
                <xs:element ref="EntryID"/>
            </xs:choice>
        </xs:complexType>
    </xs:element>
    <xs:element name="P">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="Type" type="xs:string"
use="required"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="PhysConn">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="P" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="Type" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Private">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="Type" type="xs:string"/>
            </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:element name="ReadWrite">
      <xs:complexType/>
   </xs:element>
   <xs:element name="ReasonCode">
      <xs:complexType/>
   </xs:element>
   <xs:element name="ReportControl">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="TrgOpEna" minOccurs="0"/>
            <xs:element ref="OptFlds" minOccurs="0"/>
            <xs:element ref="RptEnabled" minOccurs="0"/>
         </xs:sequence>
         <xs:attribute name="RCName" type="xs:string" use="required"/>
         <xs:attribute name="RptID" type="xs:string" use="required"/>
         <xs:attribute name="DatSet" type="xs:string" use="required"/>
         <xs:attribute name="ConfRev" type="xs:unsignedInt"
use="required"/>
         <xs:attribute name="Buffered" type="xs:boolean" de-
fault="FALSE"/>
         <xs:attribute name="BufTim" type="xs:unsignedInt" de-
fault="0"/>
         <xs:attribute name="IntgPd" type="xs:unsignedInt" de-
fault="0"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Router">
      <xs:complexType/>
   </xs:element>
   <xs:element name="RptEnabled">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="ClientLN" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Max" type="xs:unsignedInt"
use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="SBO">
      <xs:complexType/>
   </xs:element>
   <xs:element name="SCL">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Header"/>
            <xs:choice maxOccurs="unbounded">
               <xs:element ref="Substation"/>
               <xs:element ref="IED"/>
               <xs:element ref="LNodeType"/>
               <xs:element ref="Communication"/>
               <xs:element ref="EnumType"/>
            </xs:choice>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="SGEdit">
```

```
      <xs:complexType/>
   </xs:element>
   <xs:element name="SMV">
      <xs:complexType>
         <xs:attribute name="Addr" type="xs:string"/>
         <xs:attribute name="LDInst" type="xs:string"/>
         <xs:attribute name="CBName" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="SampledValueControl">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="SmvOpts" minOccurs="0"/>
            <xs:element ref="IEDName" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="SVCName" type="xs:string"
use="required"/>
         <xs:attribute name="ConfRev" type="xs:string"/>
         <xs:attribute name="SmvID" type="xs:string" use="required"/>
         <xs:attribute name="Multicast" type="xs:boolean" de-
fault="TRUE"/>
         <xs:attribute name="DatSet" type="xs:string" use="required"/>
         <xs:attribute name="SmpRate" type="xs:unsignedInt"
use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="SeqNum">
      <xs:complexType/>
   </xs:element>
   <xs:element name="Server">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="Authentication" minOccurs="0"/>
            <xs:element ref="LDevice" maxOccurs="unbounded"/>
            <xs:element ref="Association" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Timout" type="xs:unsignedInt" de-
fault="30"/>
         <xs:attribute name="SecAPRef" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Services">
      <xs:complexType>
         <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element ref="DynAssociation"/>
            <xs:element ref="GetDirectory"/>
            <xs:element ref="GetDataObjectDefinition"/>
            <xs:element ref="SGEdit"/>
            <xs:element ref="TimerActivatedControl"/>
            <xs:element ref="DataObjectDirectory"/>
            <xs:element ref="GetDataSetValue"/>
            <xs:element ref="SetDataSetValue"/>
            <xs:element ref="DataSetDirectory"/>
            <xs:element ref="ConfDataSet"/>
            <xs:element ref="DynDataSet"/>
            <xs:element ref="ReadWrite"/>
            <xs:element ref="DynReportControl"/>
            <xs:element ref="SetControlValue"/>
            <xs:element ref="GetControlValue"/>
            <xs:element ref="GSEDir"/>
            <xs:element ref="GOOSE"/>
            <xs:element ref="GSSE"/>
            <xs:element ref="FileHandling"/>
         </xs:choice>
      </xs:complexType>
   </xs:element>
   <xs:element name="SetControlValue">
      <xs:complexType/>
   </xs:element>
   <xs:element name="SetDataSetValue">
      <xs:complexType/>
   </xs:element>
   <xs:element name="TimerActivatedControl">
      <xs:complexType/>
   </xs:element>
   <xs:element name="SettingControl">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
         </xs:sequence>
         <xs:attribute name="NumOfSGs" type="xs:unsignedInt"
use="required"/>
         <xs:attribute name="ActSG" type="xs:unsignedInt" de-
fault="1"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="SmvOpts">
      <xs:complexType>
         <xs:choice maxOccurs="unbounded">
            <xs:element ref="refreshtime"/>
            <xs:element ref="samplesynchronised"/>
            <xs:element ref="samplerate"/>
         </xs:choice>
      </xs:complexType>
   </xs:element>
   <xs:element name="Subdevice">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="Desc" type="xs:string"/>
         <xs:attribute name="Phase" default="all">
            <xs:simpleType>
               <xs:restriction base="xs:NMTOKEN">
                  <xs:enumeration value="all"/>
                  <xs:enumeration value="A"/>
                  <xs:enumeration value="B"/>
                  <xs:enumeration value="C"/>
                  <xs:enumeration value="N"/>
                  <xs:enumeration value="none"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
      </xs:complexType>
   </xs:element>
   <xs:element name="Subnetwork">
      <xs:complexType>
```

```
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="ConnectedAP" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string" use="required"/>
         <xs:attribute name="Bitrate" type="xs:decimal"/>
         <xs:attribute name="Type" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Substation">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="VoltageLevel" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string"/>
         <xs:attribute name="Desc" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="Text" type="xs:string"/>
   <xs:element name="TimeStamp">
      <xs:complexType/>
   </xs:element>
   <xs:element name="Timed">
      <xs:complexType/>
   </xs:element>
   <xs:element name="Transient">
      <xs:complexType/>
   </xs:element>
   <xs:element name="TrgOpEna">
      <xs:complexType>
         <xs:choice maxOccurs="4">
            <xs:element ref="dchg"/>
            <xs:element ref="qchg"/>
            <xs:element ref="dupd"/>
            <xs:element ref="period"/>
         </xs:choice>
      </xs:complexType>
   </xs:element>
   <xs:element name="Val">
      <xs:complexType>
         <xs:simpleContent>
            <xs:extension base="xs:string">
               <xs:attribute name="SGroup" type="xs:integer"/>
            </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>
   <xs:element name="VoltageLevel">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Text" minOccurs="0"/>
            <xs:element ref="LNode" minOccurs="0" maxOc-
curs="unbounded"/>
            <xs:element ref="Bay" maxOccurs="unbounded"/>
            <xs:element ref="Private" minOccurs="0" maxOc-
curs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="Name" type="xs:string"/>
         <xs:attribute name="Desc" type="xs:string"/>
         <xs:attribute name="Voltage" type="xs:string"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="certificate">
      <xs:complexType/>
   </xs:element>
   <xs:element name="dchg">
      <xs:complexType/>
   </xs:element>
   <xs:element name="dupd">
      <xs:complexType/>
   </xs:element>
   <xs:element name="none">
      <xs:complexType/>
   </xs:element>
   <xs:element name="password">
      <xs:complexType/>
   </xs:element>
   <xs:element name="period">
      <xs:complexType/>
   </xs:element>
   <xs:element name="qchg">
      <xs:complexType/>
   </xs:element>
   <xs:element name="refreshtime">
      <xs:complexType/>
   </xs:element>
   <xs:element name="samplerate">
      <xs:complexType/>
   </xs:element>
   <xs:element name="samplesynchronised">
      <xs:complexType/>
   </xs:element>
   <xs:element name="strong">
      <xs:complexType/>
   </xs:element>
   <xs:element name="weak">
      <xs:complexType/>
   </xs:element>
</xs:schema>
```

# Annex B
(normative)

## Extension syntax for drawing layout coordinates

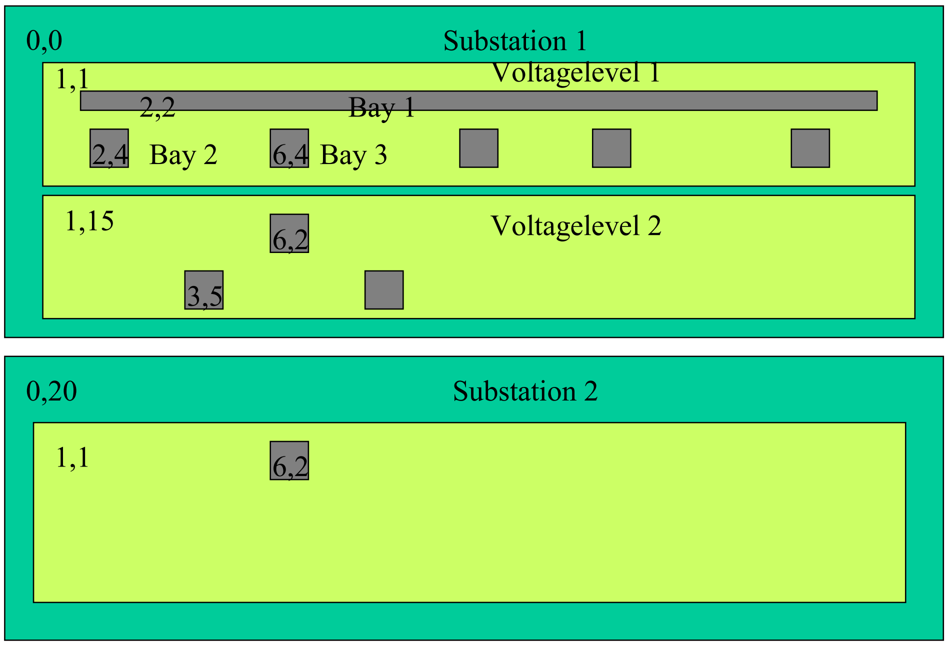
This appendix defines a simple SCL extension to add coordinates to objects, so that they can be easily shown on a drawing. This is sufficient for a lot of drawing tasks, and serves here as an example of extension of the core SCL by another name space.

The handling respective drawing of connection relations as well as the packaging of objects into drawing pages is private, respective a parameter of the interpreting application. Meaningful drawings could be a substation in a substation single line, a bay in a bay single line, the communication section for a communication configuration drawing.

The coordinate system is a relative x, y - system with coordinates using positive integer numbers. The point (0,0) is the upper left point of a drawing plane which is principally open to downwards and right direction. The unit 1 principally refers to the size of an object. It is possible to use different object sizes, then 1 is the size of the smallest object. However, then transport of coordinates between different drawing applications might lead to strange representations.

If coordinates are defined at different SCL tag hierarchy levels, then each level contains coordinates relative to the higher level. The absolute coordinate of a lower level is thus calculated by summing up all higher level coordinates, and the object coordinates themselves. In case that at a higher level there are no coordinates defined, then (x,y) = (0,0) is assumed.

This is illustrated in the following figure. Here e.g. the bay 3 of substation 1 voltage level 1 has the absolute coordinates (0+1+6, 0+1+4) = (7,5) within a picture showing the substation 1, or even both substations.



**Aditional XML elements**:

Only the additional XML attributes *x* and *y* for the coordinates in x and y direction are needed as addition to the SCL elements, which represent drawable objects. Additionally the optional attribute *dir* with value *horizontal* or *vertical* can optionally give the preferred connection direction of the object. If this attribute is defined for a bay, this means that all contained primary devices are oriented vertically, except *dir* is explicitly stated at them. The coordinate name space shall be http://www.iec.ch/61850/sclcoordinates001.

An appropriate XML schema definition is:

```
<xsd:schema xmlns:xsd=”http://www.w3.org/2001/XMLSchema”
xmlns:sxy=”http://www.iec.ch/61850/sclcoordinates001”
targetNamespace=”http://www.iec.ch/61850/sclcoordinates001”
attributeFormDefault=”qualified”>
<xsd:element name=”object” type=”anyType”>
        <xsd:attribute name=”x” type=”xsd:integer” />
        <xsd:attribute name=”y” type””xsd:integer” />
        <xsd:attribute name=”dir” type=”Conndir” default=”” />
</xsd:element>
<xsd:simpleType name=”Conndir”>
        <xsd:restriction base=xsd:string”>
                <xsd:enumeration value=”horizontal”/>
                <xsd:enumeration value=”vertical” />
        </xsd:restriction<
</xsd:simpleType>
</xsd:schema>
```

The following gives an SCL example using the coordinates. The device E1Q1QB1 in this example will have the coordinates (4, 7) relative to the substation.

```
<?xml version="1.0"?>
<SCL xmlns:sxy="http://www.iec.ch/61850/sclcoordinates001",
       xmlns="http://www.iec.ch/61850/scl001">
   <Header Ref="SCL Example" Revision="C" NameStructure="FuncName"/>
   <Substation Ref="" sxy:x="0" sxy:y="0">
      <VoltageLevel Ref="E1" sxy:x="1" sxy:y="1">
         <Bay Ref="Q1" sxy:x="1" sxy:y="1">
            <Device Ref="QA1" Type="CBR" sxy:x="2" sxy:y="2">
               <LNode Ref="1" LNClass="CSWI"/>
               <Connection CNodeRef="L1"/>
               <Private Type="MyLayout">X=5 Y=25</Private>
            </Device>
            <Device Ref="QB1" Type="DIS" sxy:x="2" sxy:y="5"
sxy:dir="vertical">
               <LNode Ref="2" LNClass="CSWI"/>
               <Connection CNodeRef="L1"/>
               <Connection CNodeRef="BB1" BayRef="W1"/>
           </Device>
         </Bay>
      </VoltageLevel>
   </Substation>
</SCL>
```

# Annex C
(normative)

## Relations between SCL and IEC61970-301 (CIM)

This appendix defines the relations between an SCL based model, and the CIM model defined in IEC61970-301 for energy system applications.

The substation structure of the SCL model (Figure 3) with the hierarchy Substation / Voltage level / Bay / Device corresponds to the substation structure in the CIM core model. The SCL tranformer modelling by means of transformer windings as device also corresponds to CIM. The description of topology connections with connectivity nodes and with terminals to connect a device to a connectivity node, also corresponds to the topology description in CIM. So this part of the models can be structurally mapped to each other.

The CIM model contains also a 'has Logical Node' relation from power system resources, i.e. all structural objects listed above, to logical nodes. This corresponds to the SCL attachment of logical nodes with the LNode element to the SCL Substation section elements.

The additional CIM relation of measurand values to logical nodes, and the CIM adoption of the measurand / measurand value naming conventions according to parts 7-x of this standard series, allow to relate SCL data objects and attributes to CIM measurand values. This gives the link to find from the CIM model within the SCL description the IEDs supplying the measurand values, and the measurand value addressing.

As explained above, the structures of CIM and SCL are prepared to be combined. However, for a concrete project also the identifying names of the instances must fit together. Naming in CIM is done with the cim:Name attribute of each power system resource, which corresponds to the Name attribute in SCL. However, within SCL the Name shall be a technical key e.g. according to IEC61346. Within CIM however often some textual, operator related names are used as cim:Name values. Therefore for combination of the CIM model part for energy management applications with the SCL model part of substation automation systems the following shall apply:

- It shall be specified, if the cim:Name contains the same values / value category then the scl:Name.
- If the cim:Name coresponds to the scl:Name, then these two attribute values shall be related to each other.
- If the cim:Name contains an operator related text, then it shall be related to the scl:Desc attribute, and this attribute shall contain the values related to the cim:Name. It is recommended that the CIM alias name is in this case used to hold the technical key corresponding to the scl:Name.

A CIM file containing at least the model parts defined for the Substation section in this standard, and keeps resp. defines the instance naming rules specified above, is a valid SCL extension. The appropriate part of the CIM model with definition of the cim namespace at the SCL element shall be a valid SCL file extension, additional to the SCL substation section.

**NOTE:** in case that the CIM class of a logical node also contains a reference to the IED and the logical device which host the LN instance, with values corresponding to the scl:IEDName and the scl:LDInst, this CIM extension can optionally replace the SCL substation section in an SCL file.

# Annex D (normative)

## Extension syntax for maintenance

This appendix defines a simple SCL extension to indicate for LNodeType attributes, if they are defined as mandatory, optional or private. As this is only needed for planning of system extensions, or by using the SCL syntax as general specification of CDCs, it is considered to be an extension package.

The name space for this extension package shall be http://www.iec.ch/61850/sclmaintenance001.

An appropriate XML schema definition is:

```
<xsd:schema xmlns:xsd=”http://www.w3.org/2001/XMLSchema”
xmlns:sma=”http://www.iec.ch/61850/sclcmaintenance001”
targetNamespace=”http://www.iec.ch/61850/sclmaintenance001” attribute-
                        FormDefault=”qualified”>
<xsd:element name=”object” type=”anyType”>
     <xsd:attribute name=”MOP” type=”mop” />
</xsd:element>
<xsd:simpleType name=”mop”>
     <xsd:restriction base=xsd:string”>
          <xsd:enumeration value=”M”/>
          <xsd:enumeration value=”O” />
          <xsd:enumeration value=”P” />
     </xsd:restriction<
</xsd:simpleType>
</xsd:schema>
```

This syntax defines an attribute mop with possible values M. O and P. The mop value M means mandatory, O means optional, and P means private, i.e. specific to the manufacturer of the concerned IED type.

## Annex E (informative)

## SCL Data Type Definition

```
<!--scl-001.dtd
   Document Type Definition for the Substation Configuration Language
   Version 0.9 02-09-18
-->

<!ELEMENT SCL (Header, (Substation | IED | LNodeType | EnumType | Com-
munication)+, Private*)>

<!ELEMENT Header (History?, Text?) >    <!--Text allows to add arbi-
trary describing text-->
<!ATTLIST Header
   Ref CDATA #REQUIRED
   Version CDATA #IMPLIED
   Revision CDATA ""
   ToolId CDATA ""
   NameStructure (FuncName | IEDName) #REQUIRED>
 <!--FuncName = name from function structure, IEDName = name from IED
structure-->


<!ELEMENT Text (#PCDATA)*> <!--an additional text used in following
elements-->
<!ELEMENT Private (#PCDATA)*> <!--unspecified additions, preserved on
standard import and export of the concerned element-->
<!ATTLIST Private
Type CDATA "">

<!ELEMENT History (Hitem)+>  <!--The History contains several history
item entries-->
<!ELEMENT Hitem (#PCDATA)*>  <!--The text can be used to explain fur-
ther details to this version-->
<!ATTLIST Hitem
Version CDATA #REQUIRED
Revision CDATA #REQUIRED
When CDATA #REQUIRED
Who CDATA ""
What CDATA ""
Why CDATA "">

<!ELEMENT Substation (Text?, LNode*, VoltageLevel+ , Private*) >
<!ATTLIST Substation
Name CDATA ""
Desc CDATA "">


<!ELEMENT VoltageLevel (Text?, LNode*, Bay+, Private* ) >
<!ATTLIST VoltageLevel
Name CDATA ""
Desc CDATA ""
Voltage CDATA #IMPLIED>

<!ELEMENT Bay (Text?, LNode*, ConNode*, Device*, Private?) >
<!ATTLIST Bay
Name CDATA ""
Desc CDATA "" >
```

```
<!ELEMENT ConNode (Text?, LNode*, Private*) >
<!ATTLIST ConNode
Name CDATA #REQUIRED
Desc CDATA "">

<!ELEMENT Device (Text?, LNode*, Connection*, Subdevice*, Private* )>
<!ATTLIST Device
Name CDATA #REQUIRED
Desc CDATA ""
Type NMTOKEN #REQUIRED>  <!--the primary apparatus type-->

<!ELEMENT Connection EMPTY>
<!ATTLIST Connection
CNodeName CDATA #REQUIRED
Terminal CDATA ""
BayName CDATA #IMPLIED
VLName CDATA #IMPLIED
SSName CDATA #IMPLIED >  <!--the substation name(line to other substa-
tion-->

<!ELEMENT Subdevice (Text?, LNode*, Private*) >
<!ATTLIST Subdevice
Name CDATA #REQUIRED
Desc CDATA ""
Phase (all | A | B | C | N | none) "all">

<!ELEMENT LNode (Text)?> <!--element identifying the SA function by
specifying a logical node (LN) as defined in parts 5 and 7 of this
standard.-->
<!ATTLIST LNode
Inst CDATA ""
LNClass NMTOKEN #REQUIRED
Prefix CDATA ""
LDInst CDATA ""
IEDName CDATA #IMPLIED>  <!--the reference to the IED which contains
the LN-->

<!ELEMENT IED (Text?, Services?, AccessPoint+, Private*) >

<!ATTLIST IED
Name CDATA ""
Desc CDATA ""
Type CDATA ""
Manufacturer CDATA ""
ConfigVersion CDATA "">

<!ELEMENT Services (DynAssociation | SettingGroups | GetDirectory |
GetDataObjectDefinition | SGEdit
    | DataObjectDirectory | GetDataSetValue | SetDataSetValue | Da-
taSetDirectory | ConfDataSet | DynDataSet | ReadWrite
    | TimerActivatedControl| DynReportControl | SetControlValue | Get-
ControlValue
    | GSEDir | GOOSE | GSSE | FileHandling)* >

<!ELEMENT DynAssociation EMPTY> <!--all services for dynamic building
of associations-->
<!ELEMENT SettingGroups EMPTY><!--all setting group related services)--
>
<!ELEMENT GetDirectory EMPTY> <!--read the server directory (all LDs,
LNs and DATA)-->
<!ELEMENT GetDataObjectDefinition EMPTY>
<!ELEMENT DataObjectDirectory EMPTY>
<!ELEMENT GetDataSetValue EMPTY>
<!ELEMENT SetDataSetValue EMPTY>
```

```
<!ELEMENT SGEdit EMPTY>
<!ELEMENT TimerActivatedControl EMPTY>
<!ELEMENT DataSetDirectory EMPTY>
<!ELEMENT ConfDataSet EMPTY>
<!ATTLIST ConfDataSet
 Max CDATA #REQUIRED
 Modify CDATA "true" >
<!ELEMENT DynDataSet EMPTY>  <!--Services to dynamically create and de-
lete data sets-->
<!ATTLIST DynDataSet
 Max CDATA #REQUIRED>
<!ELEMENT DynReportControl EMPTY>  <!--dynamically create / delete re-
port control blocks-->
<!ATTLIST DynReportControl
 Max CDATA #REQUIRED>
<!ELEMENT SetControlValue EMPTY>   <!--change control block values-->
<!ELEMENT GetControlValue EMPTY>
<!ELEMENT ReadWrite EMPTY>
<!ELEMENT GSEDir EMPTY>
<!ELEMENT GOOSE EMPTY>
<!ATTLIST GOOSE
 Max CDATA "">
<!ELEMENT GSSE EMPTY>
<!ATTLIST GSSE
 Max CDATA "">
<!ELEMENT FileHandling EMPTY> <!--all file handling services-->

<!ELEMENT AccessPoint (Text?, Router?, Clock?, (Server | LN+)?) >
   <!--a communication access point of the IED -->
<!ATTLIST AccessPoint
Name CDATA #REQUIRED>

<!ELEMENT Router EMPTY >
<!ELEMENT Clock EMPTY >

<!ELEMENT Server (Text?, Authentication?, LDevice+, Association*, Pri-
vate*)>
<!ATTLIST Server
Timout CDATA "30"
SecAPRef CDATA "">

<!ELEMENT Authentication (none | password | weak | strong | certifi-
cate)+ >
<!ELEMENT none EMPTY>
<!ELEMENT password EMPTY>
<!ELEMENT weak EMPTY>
<!ELEMENT strong EMPTY>
<!ELEMENT certificate EMPTY>

<!ELEMENT LDevice (Text?, LN0, LN*, AccessControl?, Private* ) > <!--a
logical device of the IED It shall contain at least one LN -->

<!ATTLIST LDevice
Inst CDATA ""
Desc CDATA "">

<!ELEMENT LN0 (Text?,Log?,DataSet*, ReportControl*, LogControl*, GSE-
Control*, SampledValueControl*, SettingControl?, DOI*, Inputs?, Pri-
vate* )>
<!ATTLIST LN0
LNType CDATA #REQUIRED>

<!ELEMENT LN (Text?, DataSet*, ReportControl*, LogControl*, DOI*, In-
puts?, Private* ) >
```

```
<!ATTLIST LN
Inst CDATA ""
LNClass NMTOKEN #REQUIRED
Prefix CDATA ""
LNType CDATA #REQUIRED>

<!ELEMENT Log (#PCDATA) >

<!ELEMENT DataSet (Text?, FCDA+) >      <!-- a data set definition of
the LN -->
<!ATTLIST DataSet
Name CDATA #REQUIRED
Desc CDATA "">

<!ELEMENT FCDA EMPTY>
<!ATTLIST FCDA
LDInst CDATA ""
Prefix CDATA ""
LNClass NMTOKEN ""
LNInst CDATA ""
DOName CDATA ""
DAName CDATA ""
FC CDATA "" >

<!ELEMENT ReportControl  (Text?, TrgOpEna?, OptFlds?, RptEnabled? ) >
<!ATTLIST ReportControl
RCName CDATA #REQUIRED
RptID CDATA #REQUIRED
DatSet CDATA #REQUIRED
ConfRev CDATA #REQUIRED
Buffered CDATA "FALSE"
BufTim CDATA "0"
IntgPd CDATA "0">

<!ELEMENT RptEnabled (ClientLN* ) >
<!ATTLIST RptEnabled
Max CDATA "1">

<!ELEMENT ClientLN EMPTY>
<!ATTLIST ClientLN
IEDName CDATA #REQUIRED
LDInst CDATA #REQUIRED
Prefix CDATA ""
LNInst CDATA ""
LNClass NMTOKEN #REQUIRED>

<!ELEMENT TrgOpEna (dchg | qchg | dupd | period)+>
<!ELEMENT dchg EMPTY>
<!ELEMENT qchg EMPTY>
<!ELEMENT dupd EMPTY>
<!ELEMENT period EMPTY>

<!ELEMENT OptFlds ((SeqNum | TimeStamp | DatSet | ReasonCode | DataRef|
BufOvfl | EntryID)+ )>
<!ELEMENT SeqNum EMPTY>
<!ELEMENT TimeStamp EMPTY>
<!ELEMENT DataRef EMPTY>
<!ELEMENT DatSet EMPTY>
<!ELEMENT ReasonCode EMPTY>
<!ELEMENT BufOvfl EMPTY>
<!ELEMENT EntryID EMPTY>

<!ELEMENT LogControl (Text?, TrgOpEna?) >    <!-- a log definition of
the LN -->
```

```
<!ATTLIST LogControl
LCBName CDATA #REQUIRED
LogEna CDATA "FALSE"
IntgPd CDATA "0"
DatSet CDATA #REQUIRED>

<!ELEMENT GSEControl  (Text?, IEDName*) >    <!--a GOOSE control defi-
nition-->
<!ATTLIST GSEControl
GCName CDATA #REQUIRED
ConfRev CDATA ""
Type (GSSE | GOOSE) "GOOSE"
AppID CDATA #REQUIRED
DatSet CDATA #REQUIRED>

<!ELEMENT IEDName (#PCDATA) > <!--the name of an IED in the system, to
which GOOSE or SAMPLE data shall be sent-->

<!ELEMENT SampledValueControl (Text?, SmvOpts?, IEDName* ) >  <!--a
sampled value control definition of the LN -->
<!ATTLIST SampledValueControl
SVCName CDATA #REQUIRED
ConfRev CDATA ""
SmvID CDATA #REQUIRED
Multicast CDATA "TRUE"
DatSet CDATA #REQUIRED
SmpRate NMTOKEN #REQUIRED>

<!ELEMENT SmvOpts ( (refreshtime | samplesynchronised | samplerate)+ )>
<!ELEMENT refreshtime EMPTY>
<!ELEMENT samplesynchronised EMPTY>
<!ELEMENT samplerate EMPTY>

<!ELEMENT SettingControl (Text?) >
<!ATTLIST SettingControl
NumOfSGs CDATA #REQUIRED
ActSG CDATA "1">

<!ELEMENT Inputs (Text?, ExtRef+) >
<!ELEMENT ExtRef EMPTY>
<!ATTLIST ExtRef
IEDName CDATA #REQUIRED
LDInst CDATA #REQUIRED
Prefix CDATA ""
LNClass NMTOKEN #REQUIRED
LNInst CDATA #REQUIRED
DOName CDATA #REQUIRED
DAName CDATA ""
IntAdr CDATA "">

<!ELEMENT AccessControl ANY >

<!ELEMENT Association EMPTY>
<!ATTLIST Association
Kind (pre-established | predefined) #REQUIRED
AssociationId CDATA ""
IEDName CDATA #REQUIRED
LDInst CDATA #REQUIRED
Prefix CDATA ""
LNInst CDATA #REQUIRED
LNClass NMTOKEN #REQUIRED>

<!ELEMENT Communication (Text?, Subnetwork+, Private*) >
<!ELEMENT Subnetwork (Text?, ConnectedAP+, Private* ) >
```

```
<!ATTLIST Subnetwork
Name CDATA #REQUIRED
Bitrate CDATA ""
Type CDATA "">     <!--the bus protocol type-->

<!ELEMENT GSE EMPTY>  <!--GOOSE communication parameters -->
<!ATTLIST GSE
Addr CDATA ""
LDInst CDATA ""
CBName CDATA ""
Mintime CDATA ""
Maxtime CDATA "">

<!ELEMENT SMV EMPTY >
<!ATTLIST SMV
Addr CDATA ""
LDInst CDATA ""
CBName CDATA "">

<!ELEMENT ConnectedAP (Address, GSE*, SMV*, PhysConn*, Private*) >  <!-
-the IED access point connected to the bus -->

<!ATTLIST ConnectedAP
IEDName CDATA #REQUIRED
APName CDATA #REQUIRED>

<!ELEMENT Address (P* )>
<!ELEMENT P (#PCDATA)>
<!ATTLIST P
Type CDATA #REQUIRED>

<!ELEMENT PhysConn (P* ) >
<!ATTLIST PhysConn
Type CDATA #REQUIRED>

<!ELEMENT LNodeType (Text?, DO+ ) >
<!ATTLIST LNodeType
Ref CDATA #REQUIRED
IEDType CDATA ""
Desc CDATA ""
LNClass CDATA #REQUIRED>

<!ELEMENT DOI (DAI*, Private* ) > <!--a data object which is contained
in the LN instance -->
<!ATTLIST DOI
Name CDATA #REQUIRED
Desc CDATA ""
AccessControl CDATA "">

<!ELEMENT DAI (Val*) >
<!ATTLIST DAI
SAdr CDATA ""
Valkind (Spec | Conf | RO | Set) "Set"
Name CDATA #REQUIRED> <!--the name of the configuration attribute-->

<!ELEMENT DO (Transient?, DA*, Private* ) >  <!--a data object which is
contained in the LN type-->
<!ATTLIST DO
Name CDATA #REQUIRED
Desc CDATA ""
CDC CDATA #REQUIRED
AccessControl CDATA "">

<!ELEMENT Transient EMPTY>
```

```
<!ELEMENT DA (Val*) >

<!ATTLIST DA
SAdr CDATA ""
FC (ST | MX | CO | SP | SG | SE | SV | AX | CF | DC | EX) #REQUIRED
TrgOpt (dchg | qchg | dupd | none) "none"
BType (BOOL | INT8 | INT16 | INT32 | INT8U | INT16U | INT32U | FLOAT32
| FLOAT64 | Enum | CodedEnum | VisString32 | VisString64 | VisString255
| Octet64) #REQUIRED
Valkind (Spec | Conf | RO | Set) "Set"
EnumType CDATA ""
Name CDATA #REQUIRED> <!--the name of the configuration attribute-->

<!ELEMENT EnumType (EnumVal+) >
<!ATTLIST EnumType
Ref CDATA #REQUIRED>

<!ELEMENT EnumVal (#PCDATA) >
<!ATTLIST EnumVal
Ord CDATA #REQUIRED>

<!ELEMENT Val (#PCDATA) > <!--PCDATA is the value of the configuration
attribute-->
<!ATTLIST Val
SGroup CDATA "">
```

# Annex F
(informative)

# Examples

## D.1 SCL file contents

Below you will find a small incomplete example with one substation bay and one IED just to give an idea how an SCL file may look like. It is set up in such a way that verification with the XML schema file of Appendix A can be done.

```
<?xml version="1.0"?>
<SCL xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
          xs:noNamespaceSchemaLocation="Mypath/scl-001.xsd">
   <Header Ref="SCL Example" Revision="C" NameStructure="IEDName"/>
   <Substation Name="">
      <VoltageLevel Name="E1">
         <Bay Ref="Q1">
            <Device Name="QA1" Type="CBR">
                 <LNode Inst="1" LNClass="CSWI" IEDName="AA2SB1" LDInst="C1"/>
                 <Connection CNodeName="L1"/>
              </Device>
            <Device Name="QB1" Type="DIS">
                 <LNode Name="2" LNClass="CSWI" IEDName="AA2SB1" LDInst="C1"/>
                 <Connection CNodeName="L1"/>
                 <Connection CNodeName="BB1" BayName="W1"/>
             </Device>
         </Bay>
      </VoltageLevel>
   </Substation>
   <Communication>
      <Subnetwork Name="W1" Type="8-MMS/TCP">
         <Text>Station bus</Text>
         <ConnectedAP IEDName="AA2SB1" APName="S1"><Address/>
          </ConnectedAP>
         <ConnectedAP IEDName="AA2SB2" APName="S1"><Address/>
         </ConnectedAP>
         <ConnectedAP IEDName="AA1KA5" APName="S1"><Address/>
         </ConnectedAP>
      </Subnetwork>
      <Subnetwork Name="W2" Type="8-MMS/TCP">
         <Text>remote station unit via router</Text>
         <ConnectedAP IEDName="AA1KA1" APName="S1"><Address/>
          </ConnectedAP>
         <ConnectedAP IEDName="AA1KA5" APName="S2"><Address/>
         </ConnectedAP>
      </Subnetwork>
   </Communication>
   <IED Name="AA2SB1" Type="MyType">
      <AccessPoint Ref="S1">
         <Server>
            <LDevice Inst="C1">
               <LN0 LNType="LN0"/>
                 <LN Name="1" LNClass="CSWI" LNType="CSWIa"/>
               <LN Name="2" LNClass="CSWI" LNType="CSWIa">
                  <DataSet Name="Positions">
                     <DOName LNClass="CSWI" LNInst="1" DOName="Pos" FC="ST"/>
                     <DOName LNClass="CSWI" LNInst="2" DOName="Pos" FC="ST"/>
                  </DataSet>
                  <ReportControl CBName="Report1" DataSet="Positions">
                     <RptEnabled Max="4">
              <LNName IEDName="AA1KA1" LDInst="" LNInst="1" LNClass="IHMI"/>
                     </RptEnabled>
                  </ReportControl>
                  <LogControl Name="PositionLog" LogRef="C1"
DataSet="Positions">
                      <TrgOpts><dchg/></TrgOpts>
                   </LogControl>
```

```
                </LN>
            </LDevice>
            <Association Kind="pre-established" IEDName="AA1KA1" LDInst=""
LNClass="IHMI" LNInst="1"/>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="A1KA5" Type="Ibb-PB Router">
        <!-- IED type definition for Router -->
      <AccessPoint Name="S1">
         <Router/>
      </AccessPoint>
      <AccessPoint Nam="S2"/>
   </IED>
   <LNodeType Ref="CSWIa" LNClass="CSWI">
      <DO Nam="Pos" Type="DPC">
        <CtlService> <SBO/> </CtlService>
        <DA Name=”stVal” FC=”ST” TrgOpt=”dchg” BType=”Enum” EnumType=”Pos”/>
        <DA Name=”ctlVal” FC=”CO” TrgOpt=”none” BType=”BOOL” />
      </DO>
      <DO Name="GrpAl" Type="SPS">
         <DA Name=”stVal” FC=”ST” TrgOpt=”dchg” BType=”BOOL”>
      </DO>
   </LNodeType>
   <LNodeType Ref="LN0" LNClass="LLN0">
      <DO Name="Mode" Type="INC"/>
          <DA Name=”stVal” FC=”ST” TrgOpt=”dchg” BType=”INT32”>
      </DO>
      <DO Name="Health" Type="ISI"/>
         <DA Name=”stVal” FC=”ST” TrgOpt=”dchg” BType=”INT32”>
      </DO>
   </LNodeType>
   <EnumType Ref=”Pos”>
      <EnumVal Ord=”0”>intermediate-state</EnumVal>
      <EnumVal Ord=”1”>off</EnumVal>
      <EnumValk Ord=”2”>on</EnumVal>
      <EnumVal Ord=”3”>bad-state</EnumVal>
   </EnumType>
</SCL>
```

## D2 Complete example

Here a bigger example based on the specification in part 5 section I.1.3.2 of this standard series is given. The naming of devices is changed however to conform to IEC 61346. Even this example is not 100% complete, but it illustrates most of the SCL possibilities for system description, i.e. it is a .SCD file.

## Substation configuration

### Example T 1-1

2 Voltage Levels

D1 – 220 kV
E1 – 132 kV

5 Bays

1 - D1Q1 Feeder with Transformer, CT
2 – E1Q2 Feeder with DIS, CBR, CT, VT
3 – E1Q4 Static Busbar
4 – E1Q1 Feeder with DIS, CBR , CT, VT
5 – E1Q3 Feeder with DIS, CBR , CT, VT

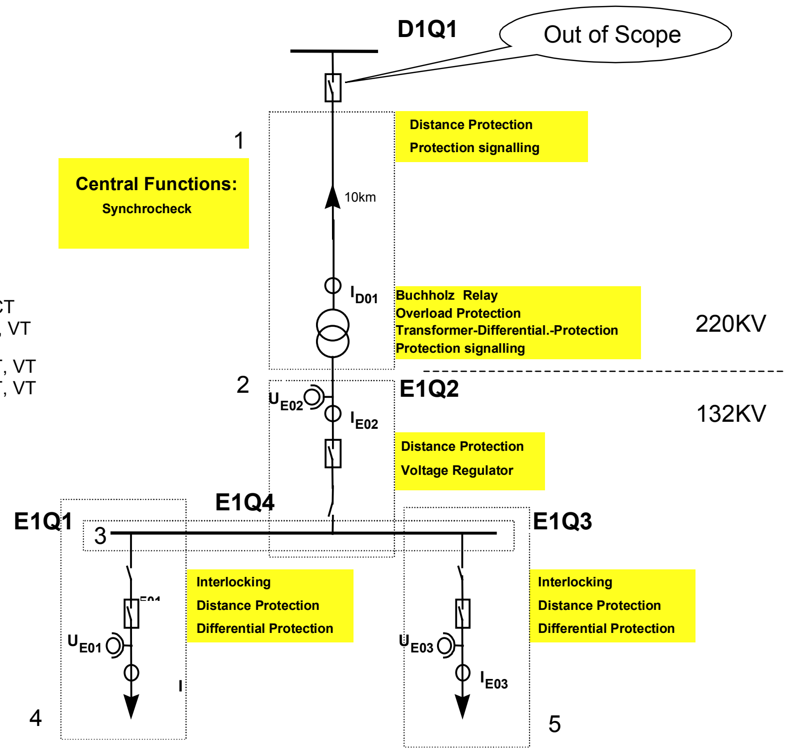


**Figure 10 - T1-1 Substation Configuration**

## Communication system configuration

### Example T 1-1

Single communication bus

IEDs for:
Transformer.
Combined Bay Unit (Circuit Breaker, Disconnector, CT and VT).
Each Protection.
Central Functions.

| No. | Name | ID |
|---|---|---|
| 1 | Dist | E1Q1BP3 (PDIS) |
| 2 | Difn | E1Q1BP2 (PDIF) |
| 3 | Dist | E1Q3BP3 (PDIS) |
| 4 | Difn | E1Q3BP2 (PDIF) |
| 5 | Dist | D1Q1BP3 (PDIS) |
| 6 | TDifn | D1Q1BP2 (PDIF) |
| 7 | Trafo | D1T1SB1 |
| 8 | LV Bay1 | E1Q1SB1 |
| 9 | LV Bay2 | E1Q2SB1 |
| 10 | LV Bay2 | E1Q3SB1 |
| 11 | Central | D1Q1SB4 (CILO, RSYN) |

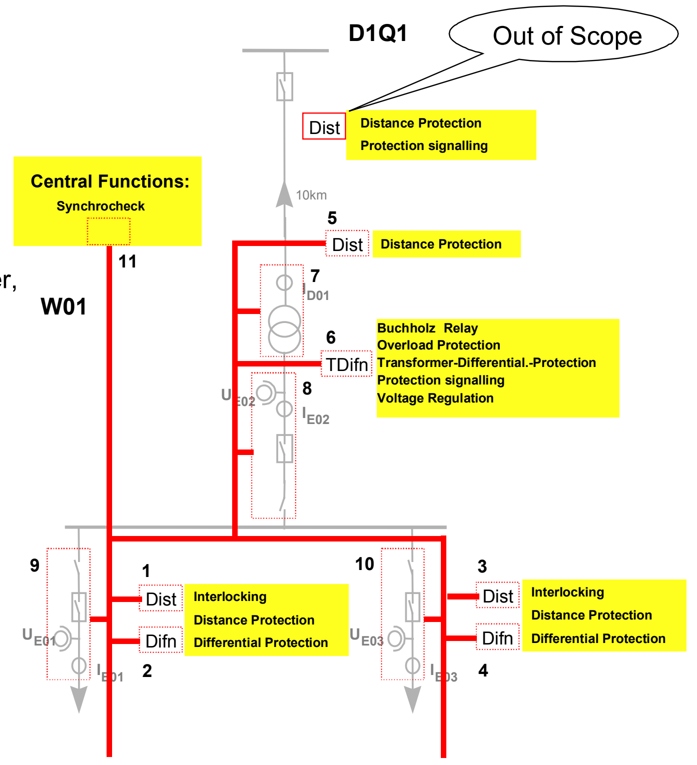


**Figure 11 - T1-1 Communication Configuration**

**Transformer IED**

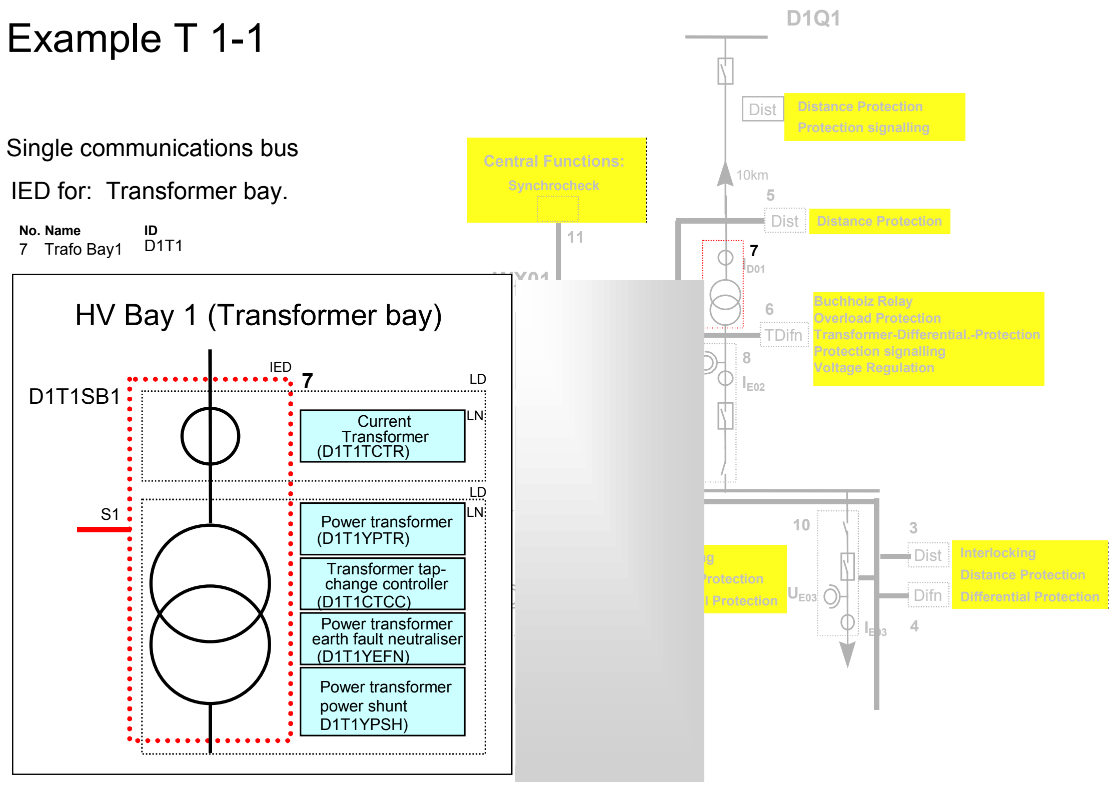


**Figure 12 - T1-1 Transformer bay**

**SCL file contents**

Below is a (not fully completed) SCL file for above example specification. Some IEDs are missing, connections of IEDs to subnetworks are missing, and the full allocation of logical nodes to the Substation is missing. A nearly complete part is bay E1Q1 with IEDs E1Q1SB1, E1Q1BP2 and E1Q1BP3.

Further control blocks and data set engineering are incomplete. A small example is given in E1Q1SB1.

```
<?xml version="1.0"?>
<SCL xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="Mypath/scl001.xsd" >
<Header Ref="SCL Example T1-1" NameStructure="IEDName"/>
   <Substation Name="">
      <VoltageLevel Name="D1" Voltage="220kV">
         <Bay Name="Q1">
            <LNode Inst="1" LNClass="PDIS" LDInst="F1" IED-
Name="D1Q1BP3"/>
            <Device Name="I1" Type="CTR">
               <Connection CNodeName="L1"/>
            </Device>
         </Bay>
      </VoltageLevel>
      <VoltageLevel Name="E1" Voltage="132kV">
         <Bay Name="Q1">
            <LNode Inst="1" LNClass="MMXU" LDInst="" IEDName="E1Q1SB1"/>
            <LNode Inst="1" LNClass="PDIS" LDInst="F1" IED-
Name="E1Q1BP3"/>
```

```
            <LNode Inst="1" LNClass="PDIF" LDInst="F1" IED-
Name="E1Q1BP2"/>
            <Device Name="QA1" Type="CBR">
               <LNode Inst="1" LNClass="CSWI" LDInst="C1" IED-
Name="E1Q1SB1"/>
               <Connection CNodeName="L1"/>
               <Connection CNodeName="L2"/>
            </Device>
            <Device Name="QB1" Type="DIS">
               <Connection CNodeName="B1" BayName="Q4"/>
               <Connection CNodeName="BB1" BayName="W1"/>
               <Connection CNodeName="L1"/>
            </Device>
            <Device Name="U1" Type="VTR">
               <Connection CNodeName="L2"/>
               <Connection CNodeName="L3"/>
            </Device>
            <Device Name="I1" Type="CTR">
               <Connection CNodeName="L3"/>
               <Connection CNodeName="L4"/>
            </Device>
         </Bay>
         <Bay Name="Q2">
            <Device Name="QA1" Type="CBR">
               <LNode Inst="1" LNClass="CILO" LDInst="C1" IED-
Name="D1Q1SB4"/>
               <Connection CNodeName="L0"/>
               <Connection CNodeName="L1"/>
            </Device>
            <Device Name="QB1" Type="DIS">
               <LNode Inst="2" LNClass="CSWI" LDInst="C1" IED-
Name="E1Q2SB1"/>
               <LNode Inst="2" LNClass="CILO" LDInst="C1" IED-
Name="D1Q1SB4"/>
               <Connection CNodeName="B1" BayName="Q4"/>
               <Connection CNodeName="L0"/>
            </Device>
            <Device Name="I1" Type="CTR">
               <Connection CNodeName="L1"/>
               <Connection CNodeName="L2"/>
            </Device>
            <Device Name="U1" Type="VTR">
               <Connection CNodeName="L2"/>
               <Connection CNodeName="L3"/>
            </Device>
         </Bay>
         <Bay Name="Q3">
            <LNode Inst="1" LNClass="MMXU" LDInst="" IEDName="E1Q3KA1"/>
            <LNode Inst="1" LNClass="PDIS" LDInst="" IEDName="E1Q3KA3"/>
            <LNode Inst="1" LNClass="PDIF" LDInst="" IEDName="E1Q3KA2"/>
            <Device Name="QA1" Type="CBR">
               <LNode Inst="1" LNClass="CSWI" LDInst="C1" IED-
Name="E1Q3SB1"/>
               <Connection CNodeName="L1"/>
               <Connection CNodeName="L2"/>
            </Device>
            <Device Name="QB1" Type="DIS">
               <Connection CNodeName="B1" BayName="Q4"/>
               <Connection CNodeName="BB1" BayName="W1"/>
               <Connection CNodeName="L1"/>
            </Device>
            <Device Name="U1" Type="VTR">
               <Connection CNodeName="L2"/>
               <Connection CNodeName="L3"/>
            </Device>
            <Device Name="I1" Type="CTR">
               <Connection CNodeName="L3"/>
               <Connection CNodeName="L4"/>
            </Device>
         </Bay>
      </VoltageLevel>
      <VoltageLevel Name="">
         <Bay Name="T1">
            <LNode Inst="1" LNClass="PDIF" LDInst="F1" IED-
Name="D1Q1BP2"/>
            <LNode Inst="1" LNClass="TCTR" LDInst="C1" IED-
Name="D1Q1SB1"/>
            <Device Name="W1" Type="PTW">
               <Connection CNodeName="L1" BayName="Q1" VLName="D1"/>
            </Device>
            <Device Name="W2" Type="PTW">
               <Connection CNodeName="L3" BayName="Q2" VLName="E1"/>
            </Device>
         </Bay>
      </VoltageLevel>
   </Substation>
   <Communication>
      <Subnetwork Name="W01" Type="8-MMS" Bitrate="10">
         <Text>Station bus</Text>
         <ConnectedAP IEDName="D1Q1SB4" APName="S1">
            <Address>
               <P Type="IP">10.0.0.11</P>
            </Address>
         </ConnectedAP>
         <ConnectedAP IEDName="E1Q1SB1" APName="S1">
            <Address>
               <P Type="IP">10.0.0.1</P>
            </Address>
         </ConnectedAP>
         <ConnectedAP IEDName="E1Q1BP2" APName="S1">
            <Address>
               <P Type="IP">10.0.0.2</P>
            </Address>
         </ConnectedAP>
         <ConnectedAP IEDName="E1Q1BP3" APName="S1">
            <Address>
               <P Type="IP">10.0.0.3</P>
            </Address>
         </ConnectedAP>
      </Subnetwork>
   </Communication>
   <IED Name="E1Q1SB1">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="C1">
               <LN0 LNType="LN0">
                  <DataSet Name="Positions">
                     <FCDA LDInst="" Prefix="" LNInst="1" LNClass="CSWI"
DOName="Pos" FC="Model"/>
                     <FCDA LDInst="" Prefix="" LNInst="2" LNClass="CSWI"
DOName="Pos" FC="PTR"/>
                  </DataSet>
         <!--this data set definition is not sufficient for all needed
communication. Either it must be completed, or the clients have to set up
their needed defintions dynamically -->
                  <ReportControl RCName="Report" RptID="E1Q1Switches"
DatSet="Positions" ConfRev="0">
                     <TrgOpEna>
```

```
                        <dchg/> <qchg/>
                     </TrgOpEna>
                     <RptEnabled Max="5">
                        <ClientLN IEDName="A1KA1" LDInst="LD1" LNInst="1"
LNClass="IHMI"/>
                     </RptEnabled>
                  </ReportControl>
                  <LogControl LCBName="Log" DatSet="Positions">
                     <TrgOpEna>
                        <dchg/> <qchg/>
                     </TrgOpEna>
                  </LogControl>
               </LN0>
               <LN Inst="1" LNClass="CSWI" LNType="CSWIa"/>
               <LN Inst="2" LNClass="CSWI" LNType="CSWIa"/>
               <LN Inst="1" LNClass="MMXU" LNType="MMXUa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="E1Q1BP2">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="F1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="PDIF" LNType="PLDif"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="E1Q1BP3">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="F1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="PDIS" LNType="PDISa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="E1Q3SB1">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="C1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="CSWI" LNType="CSWIa"/>
               <LN Inst="2" LNClass="CSWI" LNType="CSWIa">
                  <DataSet Name="Positions">
                     <FCDA LDInst="" Prefix="" LNInst="1" LNClass="CSWI"
DOName="Pos" FC="Model"/>
                     <FCDA LDInst="" Prefix="" LNInst="2" LNClass="CSWI"
DOName="Pos" FC="Model"/>
                  </DataSet>
               </LN>
               <LN Inst="1" LNClass="MMXU" LNType="MMXUa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="E1Q3BP2">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="F1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="PDIF" LNType="PLDif"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="E1Q3BP3">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="PDIS" LNType="PDISa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="D1T1SB1">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="C1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="TCTR" LNType="CTRa"/>
            </LDevice>
            <LDevice Inst="C2">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="YPTR" LNType="PTRa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="D1T1BP2">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="F1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="PDIF" LNType="PTrDif"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="D1Q1BP3">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="F1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="PDIS" LNType="PDISa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <IED Name="D1Q1SB4">
      <AccessPoint Name="S1">
         <Server>
            <LDevice Inst="C1">
               <LN0 LNType="LN0"/>
               <LN Inst="1" LNClass="RSYN" LNType="RSYNa"/>
               <LN Inst="1" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="2" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="3" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="4" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="5" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="6" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="7" LNClass="CILO" LNType="CILOa"/>
               <LN Inst="8" LNClass="CILO" LNType="CILOa"/>
            </LDevice>
         </Server>
      </AccessPoint>
   </IED>
   <LNodeType Ref="LN0" LNClass="LLN0">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="NamPlt" CDC="LPL">
         <DA Name="ldNs" FC="EX" TrgOpt="none" BType="VisString255">
            <Val>IEC61850-7-4:2002</Val>
         </DA>
         <DA Name="configRev" FC="DC" TrgOpt="none" BType="VisString255">
            <Val>Rev 3.45</Val>
         </DA>
      </DO>
   </LNodeType>
   <LNodeType Ref="CSWIa" LNClass="CSWI">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Pos" CDC="DPC">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="Enum" Enum-
Type="Pos"/>
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="BOOL"/>
      </DO>
      <DO Name="GrpAl" CDC="SPS">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
   </LNodeType>
   <LNodeType Ref="MMXUa" LNClass="MMXU">
      <DO Name="Mod" CDC="INC">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Amps" CDC="MV">
         <DA Name="mag.f" FC="MX" TrgOpt="dchg" BType="FLOAT32"/>
      </DO>
      <DO Name="Volts" CDC="MV">
         <DA Name="mag.f" FC="MX" TrgOpt="dchg" BType="FLOAT32"/>
      </DO>
   </LNodeType>
   <LNodeType Ref="PDISa" LNClass="PDIS">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
```

```
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Str" CDC="ACD">
         <DA Name="general" FC="ST" TrgOpt="dchg" BType="BOOL"/>
         <DA Name="dirGeneral" FC="ST" TrgOpt="dchg" BType="Enum" Enum-
Type="ACDdir"/>
      </DO>
      <DO Name="Op" CDC="ACT">
         <DA Name="general" FC="ST" TrgOpt="dchg" BType="BOOL"/>
      </DO>
   </LNodeType>
   <LNodeType Ref="CILOa" LNClass="CILO">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Name" CDC="LPL">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="EnaOpen" CDC="SPS">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="EnaClose" CDC="SPS">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
   </LNodeType>
   <LNodeType Ref="RSYNa" LNClass="RSYN">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Name" CDC="LPL">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Rel" CDC="SPS">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="SynPrg" CDC="SPS">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
   </LNodeType>
   <LNodeType Ref="PLDif" LNClass="PDIF">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Name" CDC="LPL">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Op" CDC="ACT">
         <DA Name="general" FC="ST" TrgOpt="dchg" BType="BOOL"/>
      </DO>
      <DO Name="Str" CDC="ACD">
         <DA Name="general" FC="ST" TrgOpt="dchg" BType="BOOL"/>
         <DA Name="dirGeneral" FC="ST" TrgOpt="dchg" BType="Enum" Enum-
Type="ACDdir"/>
      </DO>
      <DO Name="ATimCrv" CDC="CURVE">
         <DA Name="setCharact" FC="SP" TrgOpt="none" BType="INT32">
            <Val>3</Val>
         </DA>
      </DO>
   </LNodeType>
   <LNodeType Ref="PTrDif" LNClass="PDIF">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Name" CDC="LPL">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Op" CDC="ACT">
         <DA Name="general" FC="ST" TrgOpt="dchg" BType="BOOL"/>
      </DO>
      <DO Name="Str" CDC="ACD">
         <DA Name="general" FC="ST" TrgOpt="dchg" BType="BOOL"/>
         <DA Name="dirGeneral" FC="ST" TrgOpt="dchg" BType="Enum" Enum-
Type="ACDdir"/>
      </DO>
      <DO Name="ATimCrv" CDC="CURVE">
         <DA Name="setCharact" FC="SP" TrgOpt="none" BType="INT32">
            <Val>3</Val>
         </DA>
      </DO>
   </LNodeType>
   <LNodeType Ref="CTRa" LNClass="TCTR">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Amps" CDC="MV">
         <DA Name="mag.f" FC="MX" TrgOpt="dchg" BType="FLOAT32"/>
      </DO>
   </LNodeType>
   <LNodeType Ref="PTRa" LNClass="YPTR">
      <DO Name="Mode" CDC="INC">
         <DA Name="ctlVal" FC="CO" TrgOpt="none" BType="INT32"/>
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Beh" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="Health" CDC="ISI">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="INT32"/>
      </DO>
      <DO Name="IMTemp" CDC="MV">
         <DA Name="mag.f" FC="MX" TrgOpt="dchg" BType="FLOAT32"/>
      </DO>
      <DO Name="HPTemp" CDC="MV">
         <DA Name="mag.f" FC="MX" TrgOpt="dchg" BType="FLOAT32"/>
      </DO>
      <DO Name="HPTTr" CDC="SPS">
         <DA Name="stVal" FC="ST" TrgOpt="dchg" BType="BOOL"/>
      </DO>
   </LNodeType>
   <EnumType Ref="Pos">
      <EnumVal Ord="0">intermediate</EnumVal>
      <EnumVal Ord="1">off</EnumVal>
      <EnumVal Ord="2">on</EnumVal>
      <EnumVal Ord="3">failure</EnumVal>
   </EnumType>
   <EnumType Ref="ACDdir">
      <EnumVal Ord="0">unknown</EnumVal>
      <EnumVal Ord="1">forward</EnumVal>
      <EnumVal Ord="2">backward</EnumVal>
      <EnumVal Ord="3">both</EnumVal>
   </EnumType>
</SCL>
```