INTERNATIONAL ELECTROTECHNICAL COMMISSION

Technical Committee No. 3: Information structures, documentation and graphical symbols

Liaison to IEC TC65 - Industrial-process measurement and control: Representation of process control engineering requests in P&I Diagrams

This document is intended for consideration at the meeting of TC3 in Cape Town, under the item on liaison to IEC TC65.

TC3 is presently lacking a formal liaison officer to IEC TC65. The NC of Finland has drawn the attention to document 65/356/NP, which calls for co-ordination with TC3.

Document 65/356/NP Specification for Representation of process control engineering requests in P&I Diagrams and for data exchange between P&ID tools and PCE-CAE tools, was circulated in the beginning of 2005. This document has later been approved for publication as a PAS.

This document contains some comments from Finland to 3/356/NP and the document 3/356/NP.

-------------------------------
TC 65 document 65/356/NP Specification for Representation of process control engineering requests in P&I Diagrams and for data exchange between P&ID tools and PCE-CAE tools, has been approved on voting and will be published as PAS document at first stage.

At next stage PAS will be converted to IEC-standard and targets for CDV and FDIS to be decided at next plenary meeting of TC65 to be held in fall 2006.

Some comments from FI experts for document 65/356/NP

1 Documentation rules
In addition that now in the PAS/IEC 62424, ISO 10628 has been used as base of documentation rules (see figure 1) IEC 61082 rules should be possible to use in the information exchange as much as possible without any additional conversions. At present IEC 61082 rules are not in line with ISO 10628 rules.

2 Reference designation system / numbering system
Reference designation system / numbering system and its examples in the future IEC 62424 should be in line with IEC 61346 and IEC/PAS 62400.

3 Document management
The principles of IEC 82045 should be applied.

4 Data elements
The principles for the data elements used in data exchange shall be in line with TC3/SC3D requirements.

We are of the opinion that in the IEC standards at least IEC TC 3 documentation rules should be applied in TC 65 project (PAS \(\rightarrow\) IEC standard).
A proposal for a new work item within the scope of an existing technical committee or subcommittee shall be submitted to the Central Office. The proposal will be distributed to the P-members of the technical committee or subcommittee for voting, and to the O-members for information. The proposer may be a National Committee of the IEC, the secretariat itself, another technical committee or subcommittee, an organization in liaison, the Standardization Management Board or one of the advisory committees, or the General Secretary. Guidelines for proposing and justifying a new work item are given in ISO/IEC Directives, Part 1, Annex C (see extract overleaf). This form is not to be used for amendments or revisions to existing publications.

The proposal (to be completed by the proposer)

**Title of proposal**
Specification for Representation of process control engineering requests in P&I Diagrams and for data exchange between P&ID tools and PCE-CAE tools

- Standard
- Technical Specification
- Publicly Available Specification

**Scope** (as defined in ISO/IEC Directives, Part 2, 6.2.1)

This document specifies how process control engineering requests are represented in a P&I diagram. It also defines the exchange of process control engineering request relevant data between a process control engineering tool and a P&I tool by means of a data transfer language (called CAEX). These provisions apply to the export/import applications of such tools.

**Purpose and justification**, including the market relevance and relationship to Safety (Guide 104), EMC (Guide 107), Environmental aspects (Guide 109) and Quality assurance (Guide 102). (attach a separate page as annex, if necessary)

Efficient process engineering requires highly sophisticated tools for the different needs of the involved departments. These engineering tools are normally specialized in process design (PD), in process control engineering (PCE), etc. Therefore a working interoperability is essential to optimize the engineering process in total. Thus, the definition of a harmonised interface and data management is a core task to ensure a smooth workflow during the whole project, and to guarantee data consistency in the different tools.

**Target date**
for first CD .......................... for IS ..........................

Estimated number of meetings
Frequency of meetings: per year ................................

Proposed working methods
- E-mail
- ftp

**Relevant documents to be considered**
ISO 3511, ISO 14617-6, ISA S5.1, DIN 19227-1, DIN 19227-2

**Relationship of project to activities of other international bodies**
ISO SC 10 WG 11, ISO 10828 series

**Liaison organizations**
Need for coordination within ISO or IEC
- IEC TC 3, ISO TC 10

**Preparatory work**
Ensure that all copyright issues are identified. Check one of the two following boxes

- A draft is attached for vote and comment
- An outline is attached

We nominate a project leader as follows in accordance with ISO/IEC Directives, Part 1, 2.3.4 (name, address, fax and e-mail): Gerald Mayr, Linde AG, Dr.-Karl-von-Linde-Str. 6 - 14, 82049 Höllriegelskreuth (Munich)

**Concerns known patented items** (see ISO/IEC Directives, Part 2)
- Yes If yes, provide full information as an annex
- No

**Name and/or signature of the proposer**

---

Copyright © 2005 International Electrotechnical Commission, IEC. All rights reserved. It is permitted to download this electronic file, to make a copy and to print out the content for the sole purpose of preparing National Committee positions. You may not copy or “mirror” the file or printed version of the document, or any part of it, for any other purpose without permission in writing from IEC.
Comments and recommendations from the TC/SC officers

<table>
<thead>
<tr>
<th>Comments</th>
<th>Recommendations from the TC/SC officers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Work allocation</td>
<td>Project team ☑ New working group ☐ Existing working group no:</td>
</tr>
<tr>
<td>2) Draft suitable for direct submission as</td>
<td>CD ☠ CDV ☐ Publication as a PAS</td>
</tr>
<tr>
<td>3) General quality of the draft (conformity to ISO/IEC Directives, Part 2)</td>
<td>Little redrafting needed ☑ Substantial redrafting needed ☐ no draft (outline only)</td>
</tr>
<tr>
<td>4) Relationship with other activities</td>
<td>In IEC</td>
</tr>
</tbody>
</table>

In other organizations

Remarks from the TC/SC officers

Chairman and Secretary of SC65 recommend getting positive market feedback within 2 years after PAS publishing before proceeding the work towards an appropriate publication type (e.g. TR or IS).

Elements to be clarified when proposing a new work item

Title
Indicate the subject matter of the proposed new standard.
Indicate whether it is intended to prepare a standard, a technical report or an amendment to an existing standard.

Scope
Give a clear indication of the coverage of the proposed new work item and, if necessary for clarity, exclusions.
Indicate whether the subject proposed relates to one or more of the fields of safety, EMC, the environment or quality assurance.

Purpose and justification
Give details based on a critical study of the following elements wherever practicable.

a) The specific aims and reason for the standardization activity, with particular emphasis on the aspects of standardization to be covered, the problems it is expected to solve or the difficulties it is intended to overcome.
b) The main interests that might benefit from or be affected by the activity, such as industry, consumers, trade, governments, distributors.
c) Feasibility of the activity: Are there factors that could hinder the successful establishment or general application of the standard?
d) Timeliness of the standard to be produced: Is the technology reasonably stabilized? If not, how much time is likely to be available before advances in technology may render the proposed standard outdated? Is the proposed standard required as a basis for the future development of the technology in question?
e) Urgency of the activity, considering the needs of the market (industry, consumers, trade, governments etc.) as well as other fields or organizations. Indicate target date and, when a series of standards is proposed, suggest priorities.
f) The benefits to be gained by the implementation of the proposed standard; alternatively, the loss or disadvantage(s) if no standard is established within a reasonable time. Data such as product volume of value of trade should be included and quantified.
g) If the standardization activity is, or is likely to be, the subject of regulations or to require the harmonization of existing regulations, this should be indicated.

If a series of new work items is proposed, the purpose and justification of which is common, a common proposal may be drafted including all elements to be clarified and enumerating the titles and scopes of each individual item.

Relevant documents
List any known relevant documents (such as standards and regulations), regardless of their source. When the proposer considers that an existing well-established document may be acceptable as a standard (with or without amendments), indicate this with appropriate justification and attach a copy to the proposal.

Cooperation and liaison
List relevant organizations or bodies with which cooperation and liaison should exist.

Preparatory work
Indicate the name of the project leader nominated by the proposer.
Proposal of the German National Committee for a Publicly Available Specification

Specification for Representation of process control engineering requests in P&I Diagrams and for data exchange between P&ID tools and PCE-CAE tools

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Side</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>7</td>
</tr>
<tr>
<td>1 Scope</td>
<td>8</td>
</tr>
<tr>
<td>2 Normative references</td>
<td>9</td>
</tr>
<tr>
<td>3 Terms and definitions</td>
<td>9</td>
</tr>
<tr>
<td>4 Abbreviations</td>
<td>11</td>
</tr>
<tr>
<td>5 Conformity</td>
<td>11</td>
</tr>
<tr>
<td>6 Representation of PCE requests in P&amp;ID</td>
<td>12</td>
</tr>
<tr>
<td>6.1 PCE request and PCE loop</td>
<td>12</td>
</tr>
<tr>
<td>6.2 Objectives and principles</td>
<td>13</td>
</tr>
<tr>
<td>6.3 Requirements for the identification and representation of PCE requests</td>
<td>13</td>
</tr>
<tr>
<td>7 Neutral data exchange of PCE relevant P&amp;ID information</td>
<td>21</td>
</tr>
<tr>
<td>7.1 Objectives</td>
<td>21</td>
</tr>
<tr>
<td>7.2 Meaning of P&amp;I drawing elements</td>
<td>21</td>
</tr>
<tr>
<td>7.3 PCE relevant information of P&amp;ID tools</td>
<td>22</td>
</tr>
<tr>
<td>7.4 Formal description of PCE relevant information of P&amp;ID tools</td>
<td>22</td>
</tr>
<tr>
<td>8 Recommended attributes</td>
<td>28</td>
</tr>
<tr>
<td>Annex A (normative) CAEX – Data model for machine information exchange</td>
<td>30</td>
</tr>
<tr>
<td>A.1 CAEX and its Diagram conventions</td>
<td>30</td>
</tr>
<tr>
<td>A.2 Definition of Terms</td>
<td>31</td>
</tr>
<tr>
<td>A.3 Definition of Elements</td>
<td>33</td>
</tr>
<tr>
<td>A.3.1 General</td>
<td>33</td>
</tr>
<tr>
<td>A.3.2 Element CAEXFile</td>
<td>34</td>
</tr>
<tr>
<td>A.3.3 Element CAEXFile/SystemHierarchy</td>
<td>35</td>
</tr>
<tr>
<td>A.3.4 Element CAEXFile/InterfaceClassLib</td>
<td>36</td>
</tr>
<tr>
<td>A.3.5 Element CAEXFile/RoleClassLib</td>
<td>37</td>
</tr>
<tr>
<td>A.3.6 Element CAEXFile/SystemUnitClassLib</td>
<td>38</td>
</tr>
<tr>
<td>A.4 Definition of Types</td>
<td>39</td>
</tr>
<tr>
<td>A.4.1 General</td>
<td>39</td>
</tr>
<tr>
<td>A.4.2 ComplexType AttributeType</td>
<td>39</td>
</tr>
<tr>
<td>A.4.3 Element AttributeType/DefaultValue</td>
<td>40</td>
</tr>
<tr>
<td>A.4.4 Element AttributeType/refSemantic</td>
<td>40</td>
</tr>
<tr>
<td>A.4.5 Element AttributeType/Constraint</td>
<td>40</td>
</tr>
<tr>
<td>A.4.6 Element AttributeType/Quantity</td>
<td>41</td>
</tr>
</tbody>
</table>
Introduction

Efficient process engineering requires highly sophisticated tools for the different needs of the involved departments. These engineering tools are normally specialised in process design (PD), in process control engineering (PCE), etc. Therefore a working interoperability is essential to optimise the engineering process in total. Thus, the definition of a harmonised interface and data management is a core task to ensure a smooth workflow during the whole project, and to guarantee data consistency in the different tools.

This document defines procedures and specifications for the exchange of PCE relevant data provided by the P&ID module. The requirements for a change management procedure are described. A generally accepted technology for machine information exchange, the Extensible Markup Language (XML) is employed. Hereby, a common basis is given for information integration.

However, still a definition is necessary for a uniform semantics. CAEX (Computer Aided Engineering eXchange) as it is defined in this document is an appropriate data format for this purpose. This concept of data exchange is open for different applications.

The main task of a data exchange is transporting/synchronizing information from the P&ID database to the PCE databases and vice versa. The owners naming system is the key for a unique identification. For detailed information about representation of PCE loops in P&I diagrams see clause 6.

The data exchange system may be a stand-alone, vendor independent application or a module in an engineering environment. The data between a P&ID tool and a PCE tool and vice versa is exchanged via CAEX.

After the data exchange, there are three places where information about the plant is stored. Both the proprietary databases of the considered tools include private and common information. Both are stored at different places and different divisions that are working on them. Hereby, the intermediate database CAEX only stores common information. In a wider approach, the intermediate database can store both common and private information. This becomes important if a third application is connected to the neutral database. If the intermediate database is used as a temporary data stream only (without storing the information in a file), the information will be lost after processing the data conciliation.

Figure 1 illustrates the information flow for the P&ID and the PCE database reconciliation. The data exchange is done via a neutral intermediate CAEX database, not directly from database to database. The intermediate CAEX database can be a file (for file based data exchange) or a stream (for network based data exchange). The term „CAEX database“ within this standard has to be understood in this way, it does not denominate a database product as e.g. SQL.
1 Scope

This document specifies how process control engineering requests are represented in a P&I diagram.
It also defines the exchange of process control engineering request relevant data between a process control engineering tool and a P&I tool by means of a data transfer language (called CAEX). These provisions apply to the export/import applications of such tools.

The representation of the PCE functionality in P&I diagrams will be defined by a minimum number of rules to clearly indicate their category and processing function, independent from the technique of realization (see clause 6). The definition of graphical symbols for process equipment (e.g., vessels, valves, columns, etc.), their implementation and rules for the numbering system are not in the scope of this standard. These rules are independent from this standard and can be found for example in DIN EN ISO 10628 or DIN ISO 14617.

Clause 7 specifies the data flow between the different tools and the data model CAEX.

2 Normative references

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

IEC 60050-826, International Electrotechnical Vocabulary – Part 826: Chapter 826: Electrical installations of buildings

IEC 61346-1, Structuring principles and reference designation – Part 1: Basic rules


IEC 61512-1, Batch control – Part 1: Models and terminology

EN 13480-1, Metallic industrial piping – Part 1: General

EN 1594, Gas supply systems – Pipelines for maximum operating pressure over 16 bar – Functional requirements

EN 982, Safety of machinery – Safety requirements for fluid power systems and their components – Hydraulics


3 Terms and definitions

3.1 Process control equipment
equipment, having a process control function

NOTE According to IEV 826-07-01

3.2 Process control function
function to work on process variables, which is composed of basic functions of process control, are specific to units of the plant

NOTE In addition to process control functions associated with single sensors and actuators, there can also be process control functions that link input and output variables across several sensors and actuators. For instance, a process control function in the feedback path with the controlled variable as input variable and the manipulated variable as output variable, describes the action path from the sensor via the controller to the final controlling element.
3.3 PCE request
a PCE request describes requirements for process control equipment. Each PCE request is graphically represented by a bubble which collects all information on the functional requirements.

3.4 Sensor
functional unit that senses the effect of a measured variable at its input and places a corresponding measurement signal at its output.

NOTE Examples of sensors are:

a) Thermocouple c) Foil strain gauge d) pH electrode

3.5 Actuator
functional unit that generates from the controller output variable the manipulated variable required to drive the final controlling element.

NOTE If the final controlling element is mechanically actuated, it is controlled via an actuating drive. The actuator drives the actuating drive in this case.

EXAMPLE A practical example of an actuator acting directly on the final controlling element is a DC drive. The control unit takes the function of an actuator. The final controlling element is formed by the thyristor assembly that delivers a variable DC voltage as an output variable. The control unit and the thyristor assembly together form the final controlling equipment.

3.6 CE loop
a collection of PCE requests and PCE control functions depicting their functional coherence.

3.7 Bubble
the symbol used to denote and identify a process function. It contains an identification.

3.8 Process function
function in a process.

NOTE According to IEC 61512-1

3.9 PCE control function
function in a PCE control.

NOTE According to IEC 61512-1

3.10 Process category
letter, that designates the kind of process control request.
4 Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CAE</td>
<td>Computer Aided Engineering</td>
</tr>
<tr>
<td>CAEX</td>
<td>Computer Aided Engineering eXchange</td>
</tr>
<tr>
<td>CCR</td>
<td>Central Control Room</td>
</tr>
<tr>
<td>E&amp;I</td>
<td>Electrical and Instrumentation</td>
</tr>
<tr>
<td>ERP</td>
<td>Enterprise Resource Planning</td>
</tr>
<tr>
<td>GMP</td>
<td>Good Manufacturing Practice</td>
</tr>
<tr>
<td>N.A.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>PCE</td>
<td>Process Control Engineering</td>
</tr>
<tr>
<td>PCS</td>
<td>Process Control System</td>
</tr>
<tr>
<td>P&amp;ID</td>
<td>Piping and Instrumentation Diagram</td>
</tr>
<tr>
<td>PD</td>
<td>Process design</td>
</tr>
<tr>
<td>PU</td>
<td>Package Unit</td>
</tr>
<tr>
<td>SIL</td>
<td>Safety Integrity Level</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>

5 Conformity

To claim conformity to this standard with respect to the graphical representation of PCE requests in P&ID the requirements of clause 6 shall be fulfilled.

To claim conformity to this standard with respect to the PCE relevant data exchange, the requirements of clause 7 and the following requirements shall be fulfilled.

The data exchange shall be performed by a separate or integrated import/export application that provides for the data exchange between the related tool and CAEX.

**NOTE** The goal of the import/export application is to provide for a data reconciliation for the intersection of the source and target databases. It is able to read the proprietary database of the considered tool and to reconcile the data with the neutral CAEX database.

The export/import application shall check, report and provide the intersection data of both databases. The neutral database shall be open for additional applications.

The data import function shall enforce a configurable checking step (e.g., rule based) during the import process; it shall not allow unguided automatic changes. The configurable checking step shall include functionality for automatic or manual acceptance of data changes, allowing single decisions up to bulk data management.

All changes in the proprietary database and all discovered data inconsistencies shall be reported by the import application. The generation of the report shall be configurable. The import/export application has to assure that the intersection of the different databases holds the same information, and that additional division specific data is handled in a consistent way. Data manipulation by a project division is an ongoing process during the whole project and beyond it. Thus the creation, changing and deletion of data shall be possible during the life-cycle of the plant.

CAEX databases have to be consistent. This requires a consistency check before exporting the data. This procedure has to be followed after a successful data manipulation in a P&ID-tool or PCE-tool in order to bring
the new information into the neutral database or vice versa. Before any data changing action is carried out, the user shall be informed and asked for confirmation. The consistency check shall encompass at least the following steps and fulfill the following requirements:

Data export from source database to neutral database:

1) Check P&ID and PCE database for at least:
   i) duplicate PCE requests or loop designations
   ii) mandatory fields being filled in
   iii) correct use of numbering system of the PCE requests
   Inconsistent data shall not be exported.

2) Generate PCE relevant information

3) Check for changed information in comparison with previously stored data in the neutral database

4) Renaming of PCE request shall be supported by the export functionality.

5) Perform data export from proprietary into neutral database
   i) e.g. if the PCE request has been changed, the old PCE request within the neutral database has to be deleted and the new one can be exported from the proprietary database into the neutral database. The old PCE request information may be stored in a backup.
   ii) Other changes shall be performed with the existing object.

6) Generate reports after each data exchange
   i) e.g. new PCE requests list, missing PCE requests list, changed PCE requests list, deleted PCE requests list, problems and error list

Data import from neutral database into the target database:

7) Generate PCE relevant information from neutral database.

8) Check for changed information by comparing the neutral database with the target database.

9) Perform data import from the neutral into the proprietary database.

10) Renaming of PCE request shall be supported by the import functionality.

11) Generate reports after each data exchange.
   i) e.g. error lists,
   ii) inconsistencies due to imported data can be detected by the target application during the import process and are not considered within this standard.

6 Representation of PCE requests in P&ID

6.1 PCE request and PCE loop

In a P&ID the functional design of a plant is determined. Details of technical equipment are given only, if functions are correlated with the design of specific equipment. Consequentially the P&ID describes requirements for the design of the process control equipment. Each PCE request shall be illustrated in the P&ID with an individual identification. In order to meet the requirements of data handling the same identification shall not be used for different PCE requests. Functional coherence may be depicted by collecting the individual PCE requests in a PCE loop. A PCE loop does not have a graphical representation but is embedded into a numbering system. Depending on the engineering strategy, a PCE loop thereby consists of at least one, but also can combine several PCE requests. If PCE loops are used, these shall be represented in the identification of all concerned PCE requests. An example of this concept is given in Figure 2.
6.2 Objectives and principles

This subclause defines how to represent the process control engineering functionality in P&ID. Technical details of the used equipment shall not be depicted in general. This is due to the goal to ensure a smooth engineering workflow by separating process and instrumentation design.

Therefore, it is necessary to clarify:
1) the PCE category and function
2) the numbering system for PCE requests in P&ID
3) the graphical representation of PCE requests in P&ID
4) the type of functional connection between the PCE requests: the control functions
5) the graphical representation of signals in P&ID

Detailed information on complex control functions shall not be part of the P&ID. Therefore, additional documentation has to be prepared to define the required functionality. A control function shall also be individually identified and may be represented on the P&ID.

6.3 Requirements for the identification and representation of PCE requests

6.3.1 General

Each PCE request shall be graphically represented by a bubble, which collects all information about the functional requirements. Three data fields inside and ten data fields outside the bubble are defined to hold all the information of a PCE request (see Figure 3). For detailed information see 6.3.3 - 6.3.9.
As stated before, only the PCE functionality shall be depicted on P&ID, not the PCE implementation. In exceptional cases however, there might be constellations where the presentation of detailed realization information is inevitable. For example, in the case of a multi sensor element which means an instrument that produces measurements for different categories, every category shall be represented by it's an own bubble. The bubbles are stacked up, as shown in Figure 4.

In all cases where the PCE request is connected to the equipment or pipe this shall be shown by a full line, connecting the bubble with the equipment or pipe.

### 6.3.2 Types of lines

Signal lines are used to illustrate the functional relationship between PCE requests. A signal line shall be depicted as dashed line with an arrow to indicate the information flow. Source of information flow shall be a bubble of a PCE control function or request or switching action consisting of the six fields right outside the bubble. The sink of information flow shall be a bubble of a PCE request or a control function.

Process connections shall be depicted by a solid line without a direction. Multisensor instruments with only one process connection shall have an extra bubble for each category and only one process connection.

### 6.3.3 Displaying the location of the operator interface

Each PCE request is graphically represented by a bubble. This standard distinguishes the location of the operator interface between local, the local control panel and a central control room. The location does not reflect any realisation in systems.

A local interface shall be represented by a plain bubble as shown in Figure 5. It could be e.g. a pressure gauge.
6.3.4 PCE categories and processing functions

6.3.4.1 Indication of PCE categories and processing functions

The upper part of the bubble shall show the information of the PCE category and its PCE processing function. Each bubble shall have at least one PCE category and one PCE processing function. As exception for the processing function see subclause 6.3.4.3.

6.3.4.2 PCE categories

The first character shall be selected according to the measured or manipulated variable. Table 2 shows the list of the used PCE categories.
### Table 2 – PCE categories

<table>
<thead>
<tr>
<th>Letter</th>
<th>PCE category</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Analysis</td>
</tr>
<tr>
<td>B</td>
<td>(1)</td>
</tr>
<tr>
<td>C</td>
<td>(1)</td>
</tr>
<tr>
<td>D</td>
<td>(1)</td>
</tr>
<tr>
<td>E</td>
<td>Electrical value (detail definition by user)</td>
</tr>
<tr>
<td>F</td>
<td>Flow</td>
</tr>
<tr>
<td>G</td>
<td>Distance, length, position</td>
</tr>
<tr>
<td>H</td>
<td>Manual and manually initiated operation</td>
</tr>
<tr>
<td>I</td>
<td>(1)</td>
</tr>
<tr>
<td>J</td>
<td>(1)</td>
</tr>
<tr>
<td>K</td>
<td>Time based function</td>
</tr>
<tr>
<td>L</td>
<td>Level</td>
</tr>
<tr>
<td>M</td>
<td>(1)</td>
</tr>
<tr>
<td>N</td>
<td>Actuation setting (motor)</td>
</tr>
<tr>
<td>O</td>
<td>(1)</td>
</tr>
<tr>
<td>P</td>
<td>Pressure</td>
</tr>
<tr>
<td>Q</td>
<td>Quantity or Counter</td>
</tr>
<tr>
<td>R</td>
<td>Radiation</td>
</tr>
<tr>
<td>S</td>
<td>Speed or frequency</td>
</tr>
<tr>
<td>T</td>
<td>Temperature</td>
</tr>
<tr>
<td>U</td>
<td>N.A. (see 6.3.10)</td>
</tr>
<tr>
<td>V</td>
<td>Vibration</td>
</tr>
<tr>
<td>W</td>
<td>Weight, mass, force</td>
</tr>
<tr>
<td>X</td>
<td>(2)</td>
</tr>
<tr>
<td>Y</td>
<td>Actuation Setting (valve)</td>
</tr>
<tr>
<td>Z</td>
<td>(1)</td>
</tr>
</tbody>
</table>

(1) These letters can be defined by users.

(2) The unclassified letter X is intended to cover unlisted meanings that will be used only once or used to a limited extent. If used, the letter may have any number of meanings as a PCE category and any number of meanings as a PCE function.

#### 6.3.4.3 PCE processing functions

Starting with the second character, the successive letters in the upper part of the bubble shall represent the processing function of the PCE request. The letters given in Table 3 shall be used to indicate the processing function of a PCE request.
### Table 3 – PCE processing function

<table>
<thead>
<tr>
<th>Letter</th>
<th>Processing Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Alarm, Message</td>
</tr>
<tr>
<td>B</td>
<td>Restriction</td>
</tr>
<tr>
<td>C</td>
<td>Control</td>
</tr>
<tr>
<td>D</td>
<td>Difference</td>
</tr>
<tr>
<td>E</td>
<td>N.A.</td>
</tr>
<tr>
<td>F</td>
<td>Ratio</td>
</tr>
<tr>
<td>G</td>
<td>N.A.</td>
</tr>
<tr>
<td>H</td>
<td>High limit, on, opened</td>
</tr>
<tr>
<td>I</td>
<td>Indication of analog values</td>
</tr>
<tr>
<td>J</td>
<td>N.A.</td>
</tr>
<tr>
<td>K</td>
<td>N.A.</td>
</tr>
<tr>
<td>L</td>
<td>Low limit, off, closed</td>
</tr>
<tr>
<td>M</td>
<td>N.A.</td>
</tr>
<tr>
<td>N</td>
<td>N.A.</td>
</tr>
<tr>
<td>O</td>
<td>Local or DCS status indication of binary signals</td>
</tr>
<tr>
<td>P</td>
<td>N.A.</td>
</tr>
<tr>
<td>Q</td>
<td>Integrating or counting</td>
</tr>
<tr>
<td>R</td>
<td>Recorded value</td>
</tr>
<tr>
<td>S</td>
<td>Binary control function or switching function (not safety relevant)</td>
</tr>
<tr>
<td>T</td>
<td>N.A.</td>
</tr>
<tr>
<td>U</td>
<td>N.A.</td>
</tr>
<tr>
<td>V</td>
<td>N.A.</td>
</tr>
<tr>
<td>W</td>
<td>N.A.</td>
</tr>
<tr>
<td>X</td>
<td>(2)</td>
</tr>
<tr>
<td>Y</td>
<td>Computing function</td>
</tr>
<tr>
<td>Z</td>
<td>Binary control function (safety relevant)</td>
</tr>
</tbody>
</table>

(1) The triangle may also be used to indicate in a redundant way that the processing function is safety relevant (see Figure 3).

(2) The unclassified letter X is intended to cover unlisted meanings that will be used only once or used to a limited extent. If used, the letter may have any number of meanings as a PCE category and any number of meanings as a PCE function.

The letters I and R refer to the result of the preceding processing function, e.g. FIQI means the indication of a flow and its quantity.

To avoid redundant information, the PCE processing functions A, H, L, O, S and Z shall only be used outside the bubble. In this case, it is possible that the category may be depicted without a process control function.

The processing control functions shall be used in the sequence given in Table 4.

### Table 4 – Sequence combinations

<table>
<thead>
<tr>
<th>Category</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>See Table 3</td>
<td>F</td>
<td>D</td>
<td>Y</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>Q</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

NOTE: The table hierarchy shall be from left to right and per column top down.
6.3.4.4 PCE processing functions for actuators

The PCE processing functions can be used for actuators in the same way as for sensors. Some examples are shown in Table 5.

Table 5 – PCE processing functions for actuators

<table>
<thead>
<tr>
<th>Letter</th>
<th>Processing Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>YS</td>
<td>On/off valve</td>
</tr>
<tr>
<td>YC</td>
<td>Control valve</td>
</tr>
<tr>
<td>YCS</td>
<td>Control valve with on/off function</td>
</tr>
<tr>
<td>YZ</td>
<td>On/off valve (safety relevant)</td>
</tr>
<tr>
<td>YIC</td>
<td>Control valve continuous position indication</td>
</tr>
<tr>
<td>NS</td>
<td>On/off motor</td>
</tr>
<tr>
<td>NC</td>
<td>Control motor</td>
</tr>
</tbody>
</table>

The letters C, S, Z may be bound by paranthesis in order to indicate that they describe a property of the actuator and not an additional processing function.

The valve operating mechanism, e.g. electric, pneumatic or hydraulic, is not specified by its presentation in the P&ID.

6.3.5 PCE request numbering system

A general numbering system (e.g. IEC 61346) shall be used in order to identify the PCE request unambiguously. This identification shall be independent of the PCE processing function of the PCE request and depicted in the lower part of the bubble. Preceding identification levels (e.g. site, plant, unit, area) can be omitted in the bubble if the uniqueness of the request within the context of the P&ID is guaranteed (see Figure 8). If PCE requests are combined in a PCE loop, their identification shall have separated levels for the loop and the request.

![Diagram](image)

NOTE In the bubble of the depicted request only the last level of the identification system is shown. The plant, area, unit information can be taken from the lower left corner. Thus the complete identification of the request is: pppp-aaaa-uuuu-xxxx

Figure 8 – Example for a PCE request identification
6.3.6 PU-vendor and typical identification

Above the horizontal line, but outside the bubble on its left side, PU-Vendor information may be given as shown in Figure 9. This field may also be used to show other, project specific, information.

![Figure 9](image)

**Figure 9 – Example: Flow measurement delivered by vendor A specified by typical A20**

To support automatic loop, request and tag generation with PCE CAE tool requests, especially motor requests should be indicated by a “typical” number on the centre left side, outside the bubble. These “typicals” are fixed by the project-team and are used to determine the composition of the request, e.g. how the motor drive should be switched (with start-stop only, with start-stop and running indication, with current measurement, etc.).

6.3.7 Device information

Device information can be indicated in the lower zone outside the bubble on the left side. (see Figure 10).

![Figure 10](image)

**Figure 10 – Example: pH-measurement**

6.3.8 Alarming, switching and indicating

The characters H and L as PCE processing functions, indicating the high or low limit, shall be used in combination with A, O, S or Z only if an automatic action (S or Z), an operator action (A) or an indication (O) is activated when the limits are reached. In each level (e.g. H, HH, HHH) it shall be possible to combine alarming and switching function, e.g. AS or AZ. These functions shall always be indicated outside the bubble, as shown in Figure 11. Up to three levels for high and also for low alarm/switching/indication shall be possible. If necessary the safety integrity level (SIL) according to IEC 61511 may also be indicated in parenthesis.

![Figure 11](image)

**Figure 11 – Example: Flow measurement with safety-relevant high alarm and a low alarm**

The representation shall be: <processing function><alarm level>, whereas the order of the processing function shall be O, A, S, Z.

The (Z) may be used to mark an alarm being part of a safety function. If there is not only an alarming (A) but also a switching function (S), the corresponding limits may easily be documented by a controlling function U (see 6.3.10). On the other hand, the limits of the switching function may be indicated on the P&ID. If so, it shall be in an unambiguous way and shall be connected with the controlling function or the actuator starting with the SH, SHH, SL or SLL symbols as shown in Figure 12.

![Figure 12](image)

**Figure 12 – Flow measurement with high alarm and a high-high switching function**

The combination of the figure 11 and 12 is possible as shown in Figure 13.
6.3.9 Safety-relevant, GMP and quality-relevant PCE requests

Outside the bubble, a circle symbol should be used as an indication for GMP relevant sensors or actuators and a square for the indication of a quality-relevant PCE request. A triangle should be used for a safety function (see Figure 14).

These symbols shall be placed as closely as possible to the bubbles.

6.3.10 PCE control functions

PCE control functions essentially contain the functional relationship between sensors and actuators. These control functions are the “building stones”, the elements of the entire process-functionality. Mostly they are technically achieved via control system configuration. In simple configurations, e. g. one sensor and one actuator, where the relation is unambiguously represented in the P&ID, the PCE control function may be omitted.

The symbol for the PCE control function is the hexagon. This hexagon, see Figure 15, symbolises the control functionality, which has one or more sensors as inputs, and one or more actuators as outputs. The operators’ inaccessible PCE control functions may be depicted by using the same symbol but with a dashed horizontal line.

The hexagon symbol shall be connected with dashed lines and arrows to the various bubbles that represent the relevant PCE requests (see examples). The arrows indicate the direction of the information (Sensor to U and U to actuator). As for PCE requests, information about the PU vendor and a typical identification may be given on the left side (see 6.3.6).

The PCE control functions shall be identified separately. A general identification system shall be used in order to identify the PCE control function unambiguously. This identification shall be independent of the PCE processing function of the PCE control function and depicted in the lower part of the hexagon. Preceding identification levels (e. g. site, plant, unit, area) may be omitted in the hexagon if the uniqueness of the control function within the context of the P&ID is guaranteed (see 6.3.5). If control functions are integrated in a PCE loop, their identification shall have different levels for the loop and control function.

The detailed and complete function of the U shall be documented in a separate document, entitled with the U identification.

The upper part of the hexagon symbol shall contain Uaaa, where a is one or more of the processing functions A, C, D, F, Q, S, Y or Z (see Table 3).
It is possible, for example, that a US has a partial UZ-character. In that case the U shall become the designation USZ. Every USZ shall have at least one sensor and one actuator which is safety relevant, this means that at least one sensor and one actuator connected to a USZ has the Z as a processing function.

7 Neutral data exchange of PCE relevant P&ID information

7.1 Objectives

P&I drawings include a variety of information relevant for process control engineering purposes. In clause 6 is defined as to how basic information concerning PCE requests and their process-relevant functionality shall be represented in a P&I drawing. The given specification concerns primarily the graphical notation, but of course, this establishes structural and semantic fixings too. In this clause these structural and semantic fixings shall be mapped to a semiformal form. To do this, the CAEX-system-description-language (see Annex A) is used. For this language, an XML-representation is given in Annex C, which allows an open exchange of the modeled data between the P&I drawing system and the PCE systems.

7.2 Meaning of P&I drawing elements

P&I drawings show a plant (or a part of it) in its function as a physical framework. Aspects are the material flow through vessels and pipes, physical actuations (pumps, stirrers, electrical heating), the coupling between the physical and the control world (PCE requests), and the main dependencies between the control functions.

P&I diagrams show functional requirements (roles) and not the assembly of equipment. A shown pump symbolises not the equipment "pump" but the requirement: At this place a "pumping functionality" is needed. Additional attribute-requirements concerning this pumping functionality like "flow rate", "inlet pressure" and so on can be added.

![Figure 16 – P&I elements and associations (PCE relevant items are shown in dark lines)](image)

P&I diagrams show graphically the functional associations between the elements. In the example given in Figure 16 four main classes of associations are shown:

- **Signal association:**
  Symbolising the requirement that the functionality of one PCE request is influenced by the functionality of another PCE request. The signal association is notated as declared in clause 6 by a dashed line, the so-called "SignalLine". The SignalLine only symbolises the functional influence and not electrical wiring.

- **Process connection association:**
  Symbolising the coupling point at which information from the control world is transformed to a physical effect and vice versa. A process connection association always links one PCE request with one process plant element. The process connection association is notated as declared in clause 6 by a plain line, the so-called "ProcessConnectionLine". The ProcessConnectionLine only symbolises the functional coupling and not an actuation element. The length of the line is without any meaning.

- **Product association:**
  Symbolises the coupling of two pieces of equipment with the possibility of material transfer between them (pipe-pipe, pipe-vessel). The properties of this kind of association are not subject of this document.
– Mechanical association:
Symbolizes the mechanical coupling within actuation elements (drive-valve, motor-pump). The properties of
this kind of association are not subject of this document.

### 7.3 PCE relevant information of P&ID tools

Besides general structural and functional information, P&ID tools handle a variety of information, which are of
direct interest to the basic process control engineering:

1. **Control relevant information**

PCE requests, process connections, signal lines with all their attributes and interfaces described in clause 6
comprises the process-relevant information needed for the process control engineering.

2. **Additional information**

In many cases, the P&ID tools support additional process-relevant or technology-relevant functional
requirements concerning the process connections. Examples are maximum pressure, pipe diameters,
information concerning the medium and so on. Some of this information can be important for the process
control engineering system too. Clause 8 gives a list of relevant additional parameters.

### 7.4 Formal description of PCE relevant information of P&ID tools

#### 7.4.1 General

**NOTE** The P&I diagram is the most important interface between process engineering and process control
engineering. It is of fundamental interest to standardise not only the graphical notation of the PCE relevant information but
an exchange format too, which supports an open information flow from the P&ID tools to the PCE tools and vice versa.

The PCE data model for PCE relevant information as described in clause 6 is shown in Figure 17.

![Figure 17 – Process data model (PCE relevant items are shown in dark lines)](image)

The specifications in clause 6 provides that:

- PlantHierarchyItems are defined as composite objects that may contain further Plant-HierarchyItems (this
  allows creating a hierarchical plant design).
- Each PlantHierarchyItem can contain PCE requests and/or SignalLines. Furthermore, physical functions
  like valves and pipes may be presented here if desired including their interfaces, but they are not in the
  scope of this norm.
- Each PCE request contains 0...n ProcessConnectionInterfaces.
– Each PlantHierarchyItem, PCE request, Physical Function, SignalLine, ProcessConnectionInterface and SignalInterface can have a set of attributes.
– Control functions can be handled in the same way as PCE requests but does not include ProcessConnectionInterfaces.

7.4.2 Using the CAEX System Description Language

The CAEX System Description Language supports an exchange of CAE data by an XML file. It is schema-based. The semantic of the CAEX System Description Language allows for the exchange of instance data (plant data), type data (class data) and complete libraries as well. It comprises a means to support the change management process.

The schema model of the CAEX System Description Language is given in Annex A.

7.4.3 CAEX System Description Language mapping

Each PCE request is part of one and only one PlantHierarchyItem (which may of course be itself part of one and only one higher PlantHierarchyItem).

7.4.4 Basic mappings

The PlantHierarchyItem is represented by a CAEX-SystemUnitClass.

Each PCE request, which is part of this PlantHierarchyItem, is represented in CAEX by a nested role CAEX RoleObject within the SystemUnitClass.

Each PCE request possesses either a SignalInterface (Source) "main" or a ProcessConnectionInterface "ActuatorSource" with respect to the signal output of its processing function. A PCE request without any interface makes no sense.

Each defined additional switching function (SH, SHH,...,SL,...,ZH,...) implements an additional SignalInterface (source) within the PCE request.

Each ending SignalLine implements an additional SignalInterface (sink) within the target PCE request.

Each SignalLine between two PCE requests of the same PlantHierarchyItem implements an additional Link-Object, linking the corresponding interfaces of the two PCE requests within the SystemUnitClass.

Each defined alarming function (AH, A, ALL...) implements an additional AlarmInterface (source) within the PCE request.

Each PCE request shall have the following attributes (mandatory):

– Location (Local, Local Panel, Central)

Each PCE request may have one or more of the following attributes (optional):

– PU vendor (string)
– Typical identification (string)
– Device information (string)
– C processing function Boolean
– F processing function Boolean
– D processing function Boolean
– Y processing function Boolean
– B processing function Boolean
– Q processing function Boolean
– X processing function Boolean
– GMP relevant Boolean
– Safety relevant (string)
– Quality relevant Boolean
Each attribute specified for a certain PCE request has to be included in the attribute list of the corresponding CAEX-Role-Object in the given order.

The symbol for a PCE request – bubble or hexagon – carries no additional information and is not mapped to the CAEX-Model.

The local name of the PCE request within the PlantHierarchyItem shall correspond to the RoleName within the SystemUnitClass.

Process connections represent associations between PCE requests and ProcessElements. They are outside the scope of PCE and are not mapped to the CAEX model within this standard.

Each end of a process connection at a PCE request implements an additional ProcessConnectionInterface within this PCE request.

All additional information given by the P&ID tool with respect to this process connection shall be mapped to attributes of the corresponding ProcessConnectionInterface.

If defined, the attributes listed in clause 8 shall be the first in the given order.

Each defined indication function (I, O, OH, ....) implements an additional IndicationInterface.

NOTE The function OSH creates an IndicationInterface and a SignalInterface as well.

Figure 18 – PCE request data model

7.4.5 Advanced mapping

This subclause gives information only.

7.4.5.1 Connections between elements belonging to different PlantHierarchyItems

Figure 19 gives an example in which a SignalLine couples a PCE request of PlantSection A with a PCE request of PlantSection B. In this case, the plant sections themselves get external signal interfaces. PlantSection B has an external SignalSource and PlantSection A an external SignalSink.
The SignalLine in this case is mapped to three objects:
1) link which is part of PlantSection B, connecting B/027/SH with B/SIOut000
2) external link which is part of the external plant, connecting B/SIOut000 with A/SIIIn000
3) link which is part of PlantSection A, connecting A/SIIIn000 with A/080.1/In001

7.4.5.2 PCE loops

PCE loops are identified by naming conventions. PCE loops will not be mapped to structural elements.

The consuming application has to know the special meaning of the naming convention to be able to identify PCE loops.

7.4.6 Example

Figure 19 is an example for the CAEX mappings given in Annex B (see B.7). For example, the PCE request 080.5 is mapped as internal element under the system unit class. This means the PCE request is an element of a plant. It contains the required attributes and the external interfaces, which are linked with the external interfaces of the other PCE requests or technical units. The PCE request may be extended with optional attributes.
Figure 20 – Basic mappings

In the following, the full XML-text is shown for this example.
<CAEXFile FileNm="" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="D:\CAEX-Projekt\Dke 941\CAEX_ClassModel_V_1.11.xsd">
  <SystemUnitClassLib GlobalSystemUnitLibName="AnnexLib">
    <SystemUnitClass SystemUnitClassName="B.7">
      <InternalElement localElementName="080.1">
        <RoleRequirements RoleClassLibName="IEC PASS6xxxx RoleLib" refRoleClass="PCE request">
          <AdditionalAttribute LocalAttributeName="C processing function">
            <DefaultValue>true</DefaultValue>
          </AdditionalAttribute>
          <AdditionalAttributeValue refAttribute="PCE category">
            <Value>Y</Value>
          </AdditionalAttributeValue>
          <AdditionalAttributeValue refAttribute="Location">
            <Value>Central</Value>
          </AdditionalAttributeValue>
          <AdditionalExternalInterface LocalInterfaceName="In000" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="SignalSink"/>
          <AdditionalExternalInterface LocalInterfaceName="Y" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="ActuatorSource"/>
        </RoleRequirements>
      </InternalElement>
      <InternalElement localElementName="080.2">
        <RoleRequirements RoleClassLibName="IEC PASS6xxxx RoleLib" refRoleClass="PCE request">
          <AdditionalAttribute LocalAttributeName="I processing function">
            <DefaultValue>true</DefaultValue>
          </AdditionalAttribute>
          <AdditionalAttribute LocalAttributeName="C processing function">
            <DefaultValue>true</DefaultValue>
          </AdditionalAttribute>
          <AdditionalAttributeValue refAttribute="PCE category">
            <Value>T</Value>
          </AdditionalAttributeValue>
          <AdditionalAttributeValue refAttribute="Location">
            <Value>Central</Value>
          </AdditionalAttributeValue>
          <AdditionalExternalInterface LocalInterfaceName="TIC" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="SignalSource"/>
          <AdditionalExternalInterface LocalInterfaceName="AH" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="AlarmSource"/>
          <AdditionalExternalInterface LocalInterfaceName="AL" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="AlarmSource"/>
          <AdditionalExternalInterface LocalInterfaceName="In000" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="SensorSink"/>
          <AdditionalExternalInterface LocalInterfaceName="I" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="IndicationSource"/>
        </RoleRequirements>
      </InternalElement>
      <InternalElement localElementName="080.3">
        <RoleRequirements RoleClassLibName="IEC PASS6xxxx RoleLib" refRoleClass="PCE request">
          <AdditionalAttribute LocalAttributeName="Device Information">
            <DefaultValue>on/off</DefaultValue>
          </AdditionalAttribute>
          <AdditionalAttributeValue refAttribute="PCE category">
            <Value>H</Value>
          </AdditionalAttributeValue>
          <AdditionalAttributeValue refAttribute="Location">
            <Value>Local panel</Value>
          </AdditionalAttributeValue>
          <AdditionalExternalInterface LocalInterfaceName="SH" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="SignalSource"/>
          <AdditionalExternalInterface LocalInterfaceName="SL" GlobalInterfaceClassLibName="IEC PASS6xxxx InterfaceLib" refInterfaceClass="SignalSource"/>
        </RoleRequirements>
      </InternalElement>
      <InternalElement localElementName="080.4">
        <RoleRequirements RoleClassLibName="IEC PASS6xxxx RoleLib" refRoleClass="PCE request">
          <AdditionalAttribute LocalAttributeName="Y processing function">
            <DefaultValue>true</DefaultValue>
          </AdditionalAttribute>
          <AdditionalAttributeValue refAttribute="PCE category">
            <Value>U</Value>
          </AdditionalAttributeValue>
          <AdditionalAttributeValue refAttribute="Location">
            <Value>Central</Value>
          </AdditionalAttributeValue>
        </RoleRequirements>
      </InternalElement>
    </SystemUnitClass>
  </SystemUnitClassLib>
</CAEXFile>
8 Recommended attributes

The object of this subclause is to give a set of attributes, which are typically stored in P&ID systems and relevant in the PCE environments. If used, these attributes shall be exchanged in the manner as shown in Table 6 via the CAEX data exchange format.

The attributes given in Table 6 describe information with respect to the special process connections. These attributes shall be mapped to additional attributes of corresponding process connection interfaces.
Table 6 – P&ID attributes relevant in PCE environment

<table>
<thead>
<tr>
<th>Attributes</th>
<th>CAEX mapping</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium Code</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Medium code description</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Material balance point</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Pressure rating</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td>according to EN 982</td>
</tr>
<tr>
<td>Design temperature</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td>according to EN 1594</td>
</tr>
<tr>
<td>Pipe class</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td>according to EN 13480-1</td>
</tr>
<tr>
<td>Main nominal pipe size</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Adjusted nominal pipe size</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Connection size</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Heat tracing</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Heat tracing type</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Heat tracing temperature set point</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Equipment/pipeline flag</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Equipment id</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Pipe id</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Insulation type</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Insulation width</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
</tbody>
</table>

The attributes given in Table 7 concern information with respect to the internal object management. They shall be mapped to additional attributes of the corresponding object.

Table 7 – Data handling attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>CAEX mapping</th>
<th>normative reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>InternalUniqueID</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Short description</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
<tr>
<td>Long description</td>
<td>AdditionalAttribute (see A.17.6)</td>
<td></td>
</tr>
</tbody>
</table>
Annex A
(normative)

CAEX – Data model for machine information exchange

A.1 CAEX and its Diagram conventions

The neutral data format CAEX defines structures for definition of plant elements with their characteristics and its relationships. CAEX is a basis of a general exchange format for CAE planning data and is specified as XML schema.

It allows mixtures from standardised partial solutions and single solutions on all hierarchy levels, it supports the change process, it promotes but does not enforce the structure of libraries and permits the free exchange of data.

The schema diagrams uses the following conventions to illustrate the structure of the schema elements, the types of the elements, the attributes, the rules for optional elements and the repetitions (see Table 8).
### Table 8 – Convention

<table>
<thead>
<tr>
<th>Diagram Element</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rectangle with solid border</td>
<td>Indicates an Element that shall be implemented (mandatory).</td>
<td><img src="image1.png" alt="Example" /></td>
</tr>
<tr>
<td>Rectangle with dashed border</td>
<td>Indicates an Element that may be implemented (optional).</td>
<td><img src="image2.png" alt="Example" /></td>
</tr>
<tr>
<td>Datatype</td>
<td>Indicates the datatype of an Element – after the keyword “type” in the second line of an Element</td>
<td><img src="image3.png" alt="Example" /></td>
</tr>
<tr>
<td>Namespace</td>
<td>Indicates the namespace of the used datatype. (Keyword “xs:”) The described CAEX schema refers only to the namespace of W3C (xs:schema xmlns:xs=&quot;<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>&quot;)</td>
<td><img src="image4.png" alt="Example" /></td>
</tr>
<tr>
<td>Sequence</td>
<td>Indicates that the following elements shall be in the defined order.</td>
<td><img src="image5.png" alt="Example" /></td>
</tr>
<tr>
<td>Range</td>
<td>Indicates the range in which the element can occur. For example 1 to infinite.</td>
<td><img src="image6.png" alt="Example" /></td>
</tr>
<tr>
<td>Plus Sign</td>
<td>Indicates that xml-element contains another elements. The containing elements are not shown.</td>
<td><img src="image7.png" alt="Example" /></td>
</tr>
<tr>
<td>Minus Sign</td>
<td>Indicates that all containing XML-elements are shown.</td>
<td><img src="image8.png" alt="Example" /></td>
</tr>
<tr>
<td>Greying background with dashed rectangle</td>
<td>Indicates that the shown elements are composed in a defined datatype. The name of the datatype is shown at the top of the dashed rectangle with dashed lines.</td>
<td><img src="image9.png" alt="Example" /></td>
</tr>
</tbody>
</table>

### A.2 Definition of Terms

In the following, all elements and datatypes of CAEX are described.

### Table 9 – CAEX datatypes and elements

<table>
<thead>
<tr>
<th>AdditionalInformation</th>
<th>Auxiliary field that may contain any additional information. It may be used in the substructure of the Header.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdditionalAttribute</td>
<td>characterises an additional attribute</td>
</tr>
<tr>
<td>AdditionalAttributeValue</td>
<td>characterises an additional value of a defined attribute and its constraints</td>
</tr>
<tr>
<td>AdditionalExternalInterface</td>
<td>characterises an additional external interface of a defined interface</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Attribute</td>
<td>characterises a property of an defined element</td>
</tr>
<tr>
<td>AttributeNameMapping</td>
<td>mapping of an internal used attribute name to an ‘external’ defined attribute name</td>
</tr>
<tr>
<td>AttributeValue</td>
<td>values of predefined attributes may be set</td>
</tr>
<tr>
<td>CAEXFile</td>
<td>root-element of the exchanged data</td>
</tr>
<tr>
<td>Constraint</td>
<td>element to restrict the range of validity of a defined attribute</td>
</tr>
<tr>
<td>Copyright</td>
<td>organisational information about the copyright protection</td>
</tr>
<tr>
<td>DefaultValue</td>
<td>a predefined value for an attribute</td>
</tr>
<tr>
<td>Description</td>
<td>textual description</td>
</tr>
<tr>
<td>ExternalInterface</td>
<td>description of an external interface of an element</td>
</tr>
<tr>
<td>InterfaceClass</td>
<td>definition of a class of an interface type</td>
</tr>
<tr>
<td>InterfaceClassLib</td>
<td>definition of a library of interface classes. It may contain any class definitions</td>
</tr>
<tr>
<td>InterfaceNameMapping</td>
<td>mapping of an internally used interface name to an ‘externally’ defined interface name</td>
</tr>
<tr>
<td>InternalElement</td>
<td>description of the internal structure of an element</td>
</tr>
<tr>
<td>InternalLink</td>
<td>defines the relationship between internal elements</td>
</tr>
<tr>
<td>MappingObject</td>
<td></td>
</tr>
<tr>
<td>NominalScaledType</td>
<td>element of to define constraints of nominal scaled attribute values</td>
</tr>
<tr>
<td>OrdinalScaledType</td>
<td>element of to define constraints of ordinal scaled attribute values</td>
</tr>
<tr>
<td>SystemHierarchy</td>
<td>describes a concrete hierarchy of a plant</td>
</tr>
<tr>
<td>SystemHierarchyElement</td>
<td>describes a sub plant structure</td>
</tr>
<tr>
<td>PredefinedRealisation</td>
<td>element to reference a predefined realisation of a used element</td>
</tr>
<tr>
<td>Quantity</td>
<td></td>
</tr>
<tr>
<td>refSemantic</td>
<td>a reference to a definition of a defined attribute, e.g. to an attribute in a standardised library</td>
</tr>
<tr>
<td>RequiredMaxValue</td>
<td>element to define a maximum value of an attribute</td>
</tr>
<tr>
<td>RequiredMinValue</td>
<td>element to define a minimum value of an attribute</td>
</tr>
<tr>
<td>RequiredValue</td>
<td>element to define a required value of an attribute</td>
</tr>
<tr>
<td>RequiredValues</td>
<td>Element to define a required value of an attribute. It may be instantiated multiple times as a substructure of the element NominalScaledType.</td>
</tr>
<tr>
<td>Requirements</td>
<td>defines requirements as a constraint for an attribute value</td>
</tr>
<tr>
<td>Revision</td>
<td>organisational information about the state of the revision</td>
</tr>
<tr>
<td>RoleClass</td>
<td>definition of a class of a role type</td>
</tr>
<tr>
<td>RoleClassLib</td>
<td>definition of a library of role classes. It may contain any class definitions</td>
</tr>
<tr>
<td>RoleRequirements</td>
<td>Defines requirements for a role. It contains substructures for the definition of a required additional attribute, a required additional attribute value, and a required additional external interface</td>
</tr>
<tr>
<td>SingletonClassDescription</td>
<td>Definition of a class of a type SystemUnitClass as a unique description. This means it cannot be referenced by another element and it cannot be derived from another class definition.</td>
</tr>
</tbody>
</table>
A.3 Definition of Elements

A.3.1 General

The CAEX Model consists of abstract XML Elements and attributes for the specification of any plant items. Elements may have subelements and attributes.
Each element has the attributes “PropertyString” and “changeMode”. They are designed to store detailed information about the exchanged data. Usually they would contain structured information. In the case of the attribute “changeMode”, it contains a list of data: {state, create, delete, change} (see A.21).

A.3.2 Element CAEXFile

The element “CAEXFile” describes the root element of the data exchange format. The attribute “FileName” shall be used and stores the name of the transferred file.

```
<xs:complexType>
  <xs:sequence>
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="SystemHierarchy" type="SystemHierarchyLibType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="InterfaceClassLib" type="InterfaceClassLibType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="RoleClassLib" type="RoleClassLibType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="SystemUnitClassLib" type="SystemUnitClassLibType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="FileName" type="xs:string" use="required"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
```
A.3.3 Element CAEXFile/SystemHierarchy

The element “SystemHierarchy” specifies hierarchy structures of plants. Therefore, the attribute “GlobalSystemHierarchyName” shall be used, it describes the identifier of the hierarchy. A detailed description of the hierarchy may be stored in the attribute “description”. The element “SystemHierarchy” is of the type “SystemHierarchyLibType” (see A.16).

<table>
<thead>
<tr>
<th>Type</th>
<th>SystemHierarchyLibType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children</td>
<td>Description Version Revision Copyright AdditionalInformation SystemHierarchyElement</td>
</tr>
<tr>
<td>Attributes</td>
<td>Name</td>
</tr>
<tr>
<td></td>
<td>GlobalSystemHierarchyName</td>
</tr>
<tr>
<td></td>
<td>description</td>
</tr>
<tr>
<td></td>
<td>PropertyString</td>
</tr>
<tr>
<td></td>
<td>changeMode</td>
</tr>
<tr>
<td>Source</td>
<td>&lt;xs:element name=&quot;SystemHierarchy&quot; type=&quot;SystemHierarchyLibType&quot; minOccurs=&quot;0&quot; maxOccurs=&quot;unbounded&quot;/&gt;</td>
</tr>
</tbody>
</table>
A.3.4 Element CAEXFile/InterfaceClassLib

The element "InterfaceClassLib" is the root element for the definition of interface classes. It may contain multiple interface class definitions. Each interface class library shall have a unique library name "GlobalInterfaceLibName" in the exchanged file. The element “InterfaceClassLib” is of the type “InterfaceClassLibType” (see A.7).

<table>
<thead>
<tr>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>InterfaceClassLibType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children</td>
<td>Description Version Revision Copyright AdditionalInformation InterfaceClass</td>
</tr>
<tr>
<td>Attributes</td>
<td>Name</td>
</tr>
<tr>
<td>GlobalInterfaceLibName</td>
<td>xs:string</td>
</tr>
<tr>
<td>PropertyString</td>
<td>propertystring</td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
</tr>
</tbody>
</table>

Source: `<xs:element name="InterfaceClassLib" type="InterfaceClassLibType" minOccurs="0" maxOccurs="unbounded"/>`
A.3.5 Element CAEXFile/RoleClassLib

The element “RoleClassLib” is the root element for the definition of role classes. It may contain multiple role class definitions. Each role class library shall have an unique library name “GlobalRoleLibName” in the exchanged file. The element “RoleClassLib” is of the type “RoleClassLibType” (see A.12).

<table>
<thead>
<tr>
<th>Type</th>
<th>RoleClassLibType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children</td>
<td>Description Version Revision Copyright AdditionalInformation RoleClass</td>
</tr>
<tr>
<td>Attributes</td>
<td>GlobalRoleLibName xs:string required Default Default Fixed Fixed Annotation</td>
</tr>
<tr>
<td></td>
<td>PropertyString   propertystring optional state</td>
</tr>
<tr>
<td></td>
<td>changeMode       changeMode optional state</td>
</tr>
</tbody>
</table>

Source: `<xs:element name="RoleClassLib" type="RoleClassLibType" minOccurs="0" maxOccurs="unbounded"/>`
A.3.6 Element CAEXFile/SystemUnitClassLib

The element “SystemUnitClassLib” is the root element for the definition of system unit classes. It may contain multiple system unit class definitions. Each system unit class library shall have a unique library name “GlobalSystemUnitLibName” in the exchanged file. The element “SystemUnitClassLib” is of the type “SystemUnitClassLibType” (see A.17).

<table>
<thead>
<tr>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>SystemUnitClassLibType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children</td>
<td>Description Version Revision Copyright AdditionalInformation SystemUnitClass</td>
</tr>
<tr>
<td>Attributes</td>
<td>Name</td>
</tr>
<tr>
<td></td>
<td>GlobalSystemUnitLibName</td>
</tr>
<tr>
<td></td>
<td>PropertyString</td>
</tr>
<tr>
<td></td>
<td>changeMode</td>
</tr>
<tr>
<td>Source</td>
<td>&lt;xs:element name=&quot;SystemUnitClassLib&quot; type=&quot;SystemUnitClassLibType&quot; minOccurs=&quot;0&quot; maxOccurs=&quot;unbounded&quot;/&gt;</td>
</tr>
</tbody>
</table>
A.4 Definition of Types

A.4.1 General

In the following, datatypes of the CAEX metamodel are described. Basically XML allows the definition of two different kinds of datatypes: simple and complex datatypes. Simple datatypes may enhance or restrict predefined XML datatypes. Complex datatypes may have also subelements. Once defined, datatype may be used multiple times in the CAEX metamodel.

A.4.2 ComplexType AttributeType

Attributes may specify characteristics of plant items. Each attribute requires a unique attribute name stored in “LocalAttributeName”. A textual description may be stored in “description”. “attributeDatatype” may contain the type of the specified attribute, for example integer, single, string, etc. The “dimension” may contain the dimension like cm, m, or °C.

```
<xs:complexType name="AttributeType">
  <xs:sequence minOccurs="0">
    <xs:element name="DefaultValue" type="xs:anyType" minOccurs="0"/>
    <xs:element name="refSemantic" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="correspondingAttributeName" type="reference" use="required"/>
        <xs:attribute name="GlobalSemanticLibName" type="xs:string" use="required"/>
        <xs:attribute name="description" type="xs:string" use="optional"/>
        <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="Constraint" type="AttributeValueRequirementType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Quantity" minOccurs="0">
      <xs:complexType>
        <xs:attribute name="quantityName" type="reference" use="required"/>
        <xs:attribute name="GlobalQuantityLibName" type="xs:string" use="required"/>
        <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="PropertyString" type="propertystring" use="optional"/>
    <xs:element name="changeMode" type="changeMode" use="optional" default="state"/>
  </xs:sequence>
</xs:complexType>
```
A.4.3 Element AttributeType/DefaultValue

The element “DefaultValue” may store predefined values of the specified attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>xs:element name=&quot;DefaultValue&quot; type=&quot;xs:anyType&quot; minOccurs=&quot;0&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A.4.4 Element AttributeType/refSemantic

The meaning of the specified attribute may be defined in the element “refSemantic”. The meaning may be specified by a reference to a corresponding attribute in a semantic library. Therefore, the element contains the required field “correspondingAttributeName” and “GlobalSemanticLibName”. GlobalSemantic libraries may contain a list of standardised attribute definitions. The field “description” may contain more descriptions of the meaning of the referenced attribute.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>correspondingAttributeName</td>
<td>reference</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GlobalSemanticLibName</td>
<td>xs:string</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>xs:string</td>
<td>optional</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
<td>optional</td>
<td></td>
<td>state</td>
<td></td>
</tr>
<tr>
<td>source</td>
<td>xs:element name=&quot;refSemantic&quot; minOccurs=&quot;0&quot; maxOccurs=&quot;unbounded&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A.4.5 Element AttributeType/Constraint

The element “constraint” may specify restrictions of the specified attributes. For each constraint a field “constraintName” shall be used. The datatype of this element is described in A.5.12.
A.4.6 Element AttributeType/Quantity

The element “Quantity” may contain basic dimension description like length, or time. The element contains similar to the description of “refSemantic” element a field for the quantity name “quantityName” and a reference to the quantity library “GlobalQuantityLibName” where it is defined.
A.4.7 Element AttributeType/Attribute

The element “attribute” as a subelement of the attributetype may specify userdefined categories of attributes. This means a definition of structured attributes are possible. This definition describes a recursive structure of attribute definitions. It has the same attribute datatype definition.

```
<xs:element name="Attribute" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>
```
A.5 Group Header

A.5.1 General

The group “Header” specifies a general description of organisational information.

---

**Diagram:**

```
<xs:group name="Header">
<xs:sequence>
  <xs:element name="Description" minOccurs="0">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
          <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="Version" minOccurs="0">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
          <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="Revision" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="RevisionDate" type="xs:dateTime"/>
        <xs:element name="OldVersion" type="xs:string" minOccurs="0"/>
        <xs:element name="NewVersion" type="xs:string" minOccurs="0"/>
        <xs:element name="AuthorName" type="xs:string"/>
        <xs:element name="Comment" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
      <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="Copyright" minOccurs="0">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
          <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="AdditionalInformation" type="xs:string" minOccurs="0"/>
</xs:sequence>
</xs:group>
```
A.5.2 Element Header/Description

The element “Description” may store organisational information of a unit.

diagram

```
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
      <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:element name="Description" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:attribute name="title" type="xs:string" use="required"/>
    <xs:attribute name="content" type="xs:string" use="optional"/>
    <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
    <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
  </xs:complexType>
</xs:element>
```

A.5.3 Element Header/Version

The element “Version” may store information about the version.

diagram

```
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
      <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:element name="Version" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:attribute name="title" type="xs:string" use="required"/>
    <xs:attribute name="content" type="xs:string" use="optional"/>
    <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
    <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
  </xs:complexType>
</xs:element>
```
### A.5.4 Element Header/Revision

The element “Revision” may store general information about the revision.

```
source <xs:element name="Revision" minOccurs="0" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="RevisionDate" type="xs:dateTime"/>
<xs:element name="OldVersion" type="xs:string" minOccurs="0"/>
<xs:element name="NewVersion" type="xs:string" minOccurs="0"/>
<xs:element name="AuthorName" type="xs:string"/>
<xs:element name="Comment" type="xs:string" minOccurs="0"/>
</xs:sequence>
<xs:attribute name="PropertyString" type="propertystring" use="optional"/>
<xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
</xs:element>
```

### A.5.5 Element Header/Revision/RevisionDate

The element “RevisionDate” may store the date of the revision.

```
source <xs:element name="RevisionDate" type="xs:dateTime"/>
```

### A.5.6 Element Header/Revision/OldVersion

The element “OldVersion” may store the old version of the revision information.

```
source <xs:element name="OldVersion" type="xs:string" minOccurs="0"/>
```
A.5.7 Element Header/Revision/NewVersion

The element "NewVersion" may store the new version of the revision information.

```
<xs:element name="NewVersion" type="xs:string" minOccurs="0"/>
```

A.5.8 Element Header/Revision/AuthorName

The element "AuthorName" may store the name of the author who has filled out the version information.

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

A.5.9 Element Header/Revision/Comment

The element "Comment" may store comments of the revision information.

```
<xs:element name="Comment" type="xs:string" minOccurs="0"/>
```

A.5.10 Element Header/Copyright

The element "Copyright" may store copyright information.

```
<xs:element name="Copyright" minOccurs="0">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
                <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>
```
A.5.11 Element Header/AdditionalInformation

The element "AdditionalInformation" may store additional information of the organizational information. It shall contain the field "title" and may contain the field "content".

![Diagram of AdditionalInformation]

**attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>xs:string</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>content</td>
<td>xs:string</td>
<td>optional</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PropertyString</td>
<td>propertystring</td>
<td>optional</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
<td>optional</td>
<td>state</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**source**

```xml
<source>
<xs:element name="AdditionalInformation" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:attribute name="title" type="xs:string" use="required"/>
    <xs:attribute name="content" type="xs:string" use="optional"/>
    <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
    <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
  </xs:complexType>
</source>
```

A.5.12 complexType AttributeValueRequirementType

The datatype "AttributeValueRequirementType" may specify the requirements for values of attributes. It shall contain the field "constraintName".

![Diagram of AttributeValueRequirementType]

**children**

OrdinalScaledType NominalScaledType UnknownType

**used by**

AttributeType/Constraint AttributeValueType/Constraint

**attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>constraintName</td>
<td>xs:string</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PropertyString</td>
<td>propertystring</td>
<td>optional</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
<td>optional</td>
<td>state</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**source**

```xml
<source>
<xs:complexType name="AttributeValueRequirementType">
  <xs:choice>
    <xs:element name="OrdinalScaledType">
      <xs:complexType>
        <xs:sequence minOccurs="0">
          <xs:element name="RequiredMaxValue" type="xs:anyType" minOccurs="0"/>
          <xs:element name="RequiredValue" type="xs:anyType" minOccurs="0"/>
          <xs:element name="RequiredMinValue" type="xs:anyType" minOccurs="0"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="NominalScaledType">
      <xs:complexType>
        <xs:sequence minOccurs="0">
          <xs:element name="RequiredValues" type="xs:anyType" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="UnknownType">
      <xs:complexType>
        <xs:sequence minOccurs="0">
          <xs:element name="Requirements" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:choice>
</source>
```
A.5.13 Element AttributeValueRequirementType/OrdinalScaledType

The element “OrdinalScaledType” may store requirements of values of attributes if the datatype of the specified attribute is of an ordinal scaled datatype.

```
<xs:attribute name="constraintName" type="xs:string" use="required"/>
<xs:attribute name="PropertyString" type="propertystring" use="optional"/>
<xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
<xs:complexType>
  <xs:sequence minOccurs="0">
    <xs:element name="RequiredMaxValue" type="xs:anyType" minOccurs="0"/>
    <xs:element name="RequiredValue" type="xs:anyType" minOccurs="0"/>
    <xs:element name="RequiredMinValue" type="xs:anyType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

```
A.5.14 Element AttributeValueRequirementType/OrdinalScaledType/RequiredMaxValue

The element “RequiredMaxValue” may store a maximum value of a specified attribute.

```
<xs:element name="RequiredMaxValue" type="xs:anyType" minOccurs="0"/>
```

```
A.5.15 Element AttributeValueRequirementType/OrdinalScaledType/RequiredValue

The element “RequiredValue” may store a required value of a specified attribute.

```
<xs:element name="RequiredValue" type="xs:anyType" minOccurs="0"/>
```
A.5.16 Element AttributeValueRequirementType/OrdinalScaledType/RequiredMinValue

The element “RequiredMinValue” may store a minimum value of a specified attribute.

<table>
<thead>
<tr>
<th>type</th>
<th>xs:anyType</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes Name</td>
<td>Type</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;RequiredMinValue&quot; type=&quot;xs:anyType&quot; minOccurs=&quot;0&quot;/&gt;</td>
</tr>
</tbody>
</table>

A.5.17 Element AttributeValueRequirementType/NominalScaledType

The element “NominalScaledType” may store requirements of values of attributes if the datatype of the specified attribute is of a nominal scaled datatype.

<table>
<thead>
<tr>
<th>children</th>
<th>RequiredValues</th>
</tr>
</thead>
</table>
| source   | <xs:element name="NominalScaledType">  
  <xs:complexType>  
  <xs:sequence minOccurs="0">  
  <xs:element name="RequiredValues" type="xs:anyType" minOccurs="0" maxOccurs="unbounded"/>  
  </xs:sequence>  
  </xs:complexType>  
  </xs:element> |

A.5.18 Element AttributeValueRequirementType/NominalScaledType/RequiredValues

The element “RequiredValues” may store values of specified attribute.

<table>
<thead>
<tr>
<th>type</th>
<th>xs:anyType</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes Name</td>
<td>Type</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;RequiredValues&quot; type=&quot;xs:anyType&quot; minOccurs=&quot;0&quot; maxOccurs=&quot;unbounded&quot;/&gt;</td>
</tr>
</tbody>
</table>
A.5.19 Element AttributeValueRequirementType/UnknownType

The element "UnknownType" may store requirements of values of attributes if the datatype of the specified attribute is unknown.

```
<xs:element name="UnknownType">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element name="Requirements" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

A.5.20 Element AttributeValueRequirementType/UnknownType/Requirements

The element "Requirements" may store any requirements of a specified attribute.

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

A.6 ComplexType AttributeValueType

A.6.1 General

The datatype "AttributeValueType" may store values and constraints of defined attributes. Predefined attribute definitions can be enhanced by value definitions or restricted by constraint definitions. It contains the required field "refAttribute" which refers to the definition of the attribute. The description and dimension of the attribute can be stored in the fields "description" and "dimension".

```
<xs:complexType name="AttributeValueType">
  <xs:sequence>
    <xs:element name="Value" type="xs:anyType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Constraint" type="AttributeValueRequirementType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="refAttribute" type="reference" use="required"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="dimension" type="xs:string" use="optional"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
```

A.5.19 Element AttributeValueRequirementType/UnknownType

The element "UnknownType" may store requirements of values of attributes if the datatype of the specified attribute is unknown.

```
<xs:element name="UnknownType">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element name="Requirements" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

A.5.20 Element AttributeValueRequirementType/UnknownType/Requirements

The element "Requirements" may store any requirements of a specified attribute.

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

A.6 ComplexType AttributeValueType

A.6.1 General

The datatype "AttributeValueType" may store values and constraints of defined attributes. Predefined attribute definitions can be enhanced by value definitions or restricted by constraint definitions. It contains the required field "refAttribute" which refers to the definition of the attribute. The description and dimension of the attribute can be stored in the fields "description" and "dimension".

```
<xs:complexType name="AttributeValueType">
  <xs:sequence>
    <xs:element name="Value" type="xs:anyType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Constraint" type="AttributeValueRequirementType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="refAttribute" type="reference" use="required"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="dimension" type="xs:string" use="optional"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
```
A.6.2 Element AttributeValueType/Value

The element “Value” may store the value of a specified attribute.

\[
\text{type} \quad \text{xs:anyType}
\]

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\text{Name} & \text{Type} & \text{Use} & \text{Default} & \text{Fixed} & \text{Annotation} \\
\hline
\text{source} & \text{<xs:element name="Value" type="xs:anyType" minOccurs="0"/>} \\
\hline
\end{array}
\]

A.6.3 Element AttributeValueType/Constraint

The element “Constraint” may store restrictions of specified attribute. It is based in the defined datatype “AttributeValueRequirementType” (see A.5.12).

\[
\text{type} \quad \text{AttributeValueRequirementType}
\]

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\text{Name} & \text{Type} & \text{Use} & \text{Default} & \text{Fixed} & \text{Annotation} \\
\hline
\text{constraintName} & \text{xs:string} & \text{required} & \text{} & \text{} & \text{} \\
\text{PropertyString} & \text{propertystring} & \text{optional} & \text{} & \text{} & \text{} \\
\text{changeMode} & \text{changeMode} & \text{optional} & \text{state} & \text{} & \text{} \\
\hline
\text{source} & \text{<xs:element name="Constraint" type="AttributeValueRequirementType" minOccurs="0" maxOccurs="unbounded"/>} \\
\hline
\end{array}
\]

A.7 ComplexType InterfaceClassLibType

A.7.1 General

The datatype “InterfaceClassLibType” specifies the structure of interface libraries. It may contain organizational information, stored in the element “Header” and class definitions of interfaces. The datatype shall contain the field “GlobalInterfaceLibName”.
A.7.2 Element InterfaceClassLibType/InterfaceClass

Interfaces are connection points of plant items. The definition of an interface may be stored in the element “InterfaceClass”. Each interface class shall contain a field “InterfaceClassName” which shall be unique in the structure of the interface library.
The datatype "InterfaceClassType" defines structures for the specification of interface classes. Each interface shall have an unique identifier stored in the field "InterfaceClassName".
A.8.2 Element InterfaceClassType/Attribute

The element “Attribute” is of the datatype “AttributeType”. For description of this datatype see A.4.2.
A.8.3 Element InterfaceClassType/AttributeValue

The element “AttributeValue” is of the datatype “AttributeValueType”. For description of this datatype see A.6.

A.9 ComplexType InterfaceFamilyType

A.9.1 General

The datatype “InterfaceFamilyType” is an extension of the datatype “InterfaceClassType”. For the description of this datatype see A.8. The datatype “InterfaceFamilyType” allows the creation of child elements of interface classes. Child classes contain all characteristics (attribute elements) of the base class but may be extended
by new definitions of characteristics (attribute elements). Derived classes may restrict the value of predefined characteristics (attribute elements) by means of constraints.

A.9.2 Element InterfaceFamilyType/InterfaceClass

The element “InterfaceClass” defines a class derived from base interface class. The element is of the datatype “InterfaceFamilyType”. For a description of this datatype see A.9.
A.10 ComplexType InterfaceRequirementType

A.10.1 General

The datatype “InterfaceRequirementType” may specify the requirements of an interface. Existing interface class definitions may be enhanced by additional attributes or values of attributes to be set. If used, this datatype shall contain a field “LocalInterfaceName”. With the field “refInterfaceClass” an existing interface class definition may be referenced.
A.10.2 Element InterfaceRequirementType/AdditionalAttribute

The element “AdditionalAttribute” is of the datatype “AttributeType”. For description of this datatype see A.4.2.
A.10.3 Element InterfaceRequirementType/AttributeValue

The element "AttributeValue" is of the datatype "AttributeValueType". For description of this datatype see A.6.
A.11 ComplexType MappingType

A.11.1 General

The datatype “MappingType” defines structures for the mapping of a general identifier to an user defined identifier.

```
<xs:complexType name="MappingType">
  <xs:sequence minOccurs="0">
    <xs:element name="AttributeNameMapping" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="SystemUnitAttributeName" type="xs:string" use="required"/>
        <xs:attribute name="RoleAttributeName" type="xs:string" use="required"/>
        <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
        <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="InterfaceNameMapping" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="SystemUnitInterfaceName" type="xs:string" use="required"/>
        <xs:attribute name="RoleInterfaceName" type="xs:string" use="required"/>
        <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
        <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
```

A.11.2 Element MappingType/AttributeNameMapping

The element “AttributeNameMapping” maps the role attribute name to a system unit attribute name. Therefore, it contains the fields “SystemUnitAttributeName” and “RoleAttributeName”.

```
<xs:element name="AttributeNameMapping" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:attribute name="SystemUnitAttributeName" type="xs:string" use="required"/>
    <xs:attribute name="RoleAttributeName" type="xs:string" use="required"/>
    <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
    <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
  </xs:complexType>
</xs:element>
```
A.11.3 Element MappingType/InterfaceNameMapping

The element “InterfaceNameMapping” maps the role interface name to a system unit interface name. Therefore, it contains the fields “SystemUnitInterfaceName” and “RoleInterfaceName”.

<table>
<thead>
<tr>
<th>attributes</th>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SystemUnitInterfaceName</td>
<td>xs:string</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RoleInterfaceName</td>
<td>xs:string</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PropertyString</td>
<td>propertystring</td>
<td>optional</td>
<td></td>
<td></td>
<td>state</td>
<td></td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
<td>optional</td>
<td>default</td>
<td>state</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

source
```xml
<xs:complexType>
  <xs:attribute name="SystemUnitInterfaceName" type="xs:string" use="required"/>
  <xs:attribute name="RoleInterfaceName" type="xs:string" use="required"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
```

A.12 ComplexType RoleClassLibType

A.12.1 General

The datatype “RoleClassLibType” specifies the structure of role libraries. Role libraries may contain multiple role class definitions. Each Role library shall have an unique library name stored in the field “GlobalRoleLibName”.

<table>
<thead>
<tr>
<th>children</th>
<th>Description Version Revision Copyright AdditionalInformation RoleClass</th>
</tr>
</thead>
</table>

used by
```xml
element    CAEXFile/RoleClassLib
```

<table>
<thead>
<tr>
<th>attributes</th>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlobalRoleLibName</td>
<td>xs:string</td>
<td>required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PropertyString</td>
<td>propertystring</td>
<td>optional</td>
<td></td>
<td></td>
<td>state</td>
<td></td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
<td>optional</td>
<td></td>
<td>state</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

source
```xml
<xs:complexType name="RoleClassLibType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="RoleClass" type="RoleFamilyType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```
A.12.2 Element RoleClassLibType/RoleClass

The specification of roles may be stored in the element “RoleClass”. Each role class shall contain a field “RoleClassName” which shall be unique in the structure of the role library.
A.13 ComplexType RoleClassType

A.13.1 General

The datatype “RoleClassType” defines structures for the specification of role classes. Each role shall have an
unique identifier stored in the field “RoleClassName”.

```xml
<source>
<xs:complexType name="RoleClassType">
    <xs:sequence minOccurs="0">
        <xs:group ref="Header" minOccurs="0"/>
        <xs:element name="Attribute" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="AttributeValue" type="AttributeValue" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="ExternalInterface" type="InterfaceRequirementType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="PredefinedRealisation" minOccurs="0">
            <xs:complexType>
                <xs:complexContent>
                    <xs:extension base="SystemUnitClassType">
                        <xs:attribute name="GlobalSystemUnitClassLibName" type="xs:string" use="optional"/>
                        <xs:attribute name="refBaseSystemUnitClass" type="xs:string" use="optional"/>
                        <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
                        <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</source>
```
A.13.2 Element RoleClassType/Attribute

The element "Attribute" is of the datatype "AttributeType". For description of this datatype see A.4.2.

A.13.3 Element RoleClassType/AttributeValue

The element "AttributeValue" is of the datatype "AttributeValue Type". For description of this datatype see A.6.
A.13.4 Element RoleClassType/ExternalInterface

The element “ExternalInterface” may specify external interfaces of roles. The element is of the datatype “InterfaceRequirementType”. For description of this datatype see A.10. The element shall have a field “LocalInterfaceName”. Each interface shall have a unique interface identifier. Existing interfaces may be referenced by the fields “refInterfaceClass” and “GlobalInterfaceClassLibName” when these interface classes are defined.

A.13.5 Element RoleClassType/PredefinedRealisation

The element “PredefinedRealisation” may assign a certain class definition of a system unit to a role specification. Therefore, it contains the field “GlobalSystemUnitLibName” to reference the library where the class is defined. The field “refBaseSystemUnitClass” may reference to an existing system unit class. The referenced system unit class definition may be enhanced by a detailed specification of the datatype “SystemUnitClassType”. The element “PredefinedRealisation” is of the datatype “SystemUnitClassType”. For a description of this datatype see A.18.
### Type Extension of SystemUnitClassType

**children**
- Description
- Version
- Revision
- Copyright
- AdditionalInformation
- Attribute
- AttributeValue
- ExternalInterface
- InternalElement
- SupportedRoleClass
- InternalLink

**attributes**
- **GlobalSystemUnitClassLibName** (`xs:string`, `optional`)
- **refBaseSystemUnitClass** (`xs:string`, `optional`)
- **PropertyString** (`propertystring`, `optional`)
- **changeMode** (`changeMode`, `optional`, default: `state`)

**source**

```xml
<xs:element name="PredefinedRealisation" minOccurs="0"/>
<xs:complexType>
    <xs:complexContent>
        <xs:extension base="SystemUnitClassType">
            <xs:attribute name="GlobalSystemUnitClassLibName" type="xs:string" use="optional"/>
            <xs:attribute name="refBaseSystemUnitClass" type="xs:string" use="optional"/>
            <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
            <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```
A.14 ComplexType RoleFamilyType

A.14.1 General

The datatype “RoleFamilyType” is an extension of the datatype “RoleClassType”. For a description of this datatype see A.13. The datatype “RoleFamilyType” allows the creation of child elements of role classes. Child classes contain all characteristics (attribute elements) of the base class but may be extended by new definitions of characteristics (attribute elements). Derived classes may restrict the value of predefined characteristics (attribute elements) by means of constraints.
A.14.2 Element RoleFamilyType/RoleClass

The element "RoleClass" defines a derived class of a base role class. The element is of the datatype "RoleFamilyType". For a description of this datatype see A.14.
A.15 ComplexType SystemHierarchyElementType

A.15.1 General

The datatype "SystemHierarchyElementType" allows the definition of hierarchical structures. This datatype is an extension of the datatype "SystemUnitInstanceType". For a description of this datatype see A.20. The subelement "SystemHierarchyElement" allows a recursive definition of plant hierarchies. Each "SystemHierarchyElementType" shall have a field "SystemUnitInstanceName" to store the identifier of the hierarchy.

```xml
<xs:complexType name="SystemHierarchyElementType">
  <xs:complexContent>
    <xs:extension base="SystemUnitInstanceType">
      <xs:sequence minOccurs="0">
        <xs:element name="SystemHierarchyElement" type="SystemHierarchyElementType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
A.15.2 Element SystemHierarchyElementType/SystemHierarchyElement

The element “SystemHierarchyElement” is of the datatype “SystemHierarchyElementType”. For a description of this datatype see A.15.

A.16 ComplexType SystemHierarchyLibType

A.16.1 General

The datatype “SystemHierarchyLibType” allows for the definition of plant hierarchies. The name of the base hierarchy shall be stored in the field “GlobalSystemHierarchyName”. Underlaying hierarchies may be defined by the subelement “SystemHierarchyElement”.

```
<xs:element name="SystemHierarchyElement" type="SystemHierarchyElementType" minOccurs="0" maxOccurs="unbounded"/>
```
A.16.2 Element SystemHierarchyLibType/SystemHierarchyElement

The element “SystemHierarchyElement” is of the datatype “SystemHierarchyElementType”. For a description of this datatype see A.15.
A.17 ComplexType SystemUnitClassLibType

A.17.1 General

The datatype “SystemUnitClassLibType” specifies the structure of system unit libraries. System unit libraries may contain multiple system unit class definitions. Each system unit library shall have an unique library name stored in the field “GlobalSystemUnitLibName”.

```xml
<xs:element name="SystemHierarchyElementType" type="SystemHierarchyElementType" minOccurs="0" maxOccurs="unbounded"/>
```

A.17.2 Element SystemUnitClassLibType/SystemUnitClass

The specification of a system unit may be stored in the element "SystemUnitClass". Each system unit class shall contain a field “SystemUnitClassName” which shall be unique in the structure of the system unit library.
**SystemUnitFamilyType**

- **Description**
  - Type: xs:string

- **Version**
  - Type: xs:string

- **Revision**
  - Type: xs:string

- **Copyright**
  - Type: xs:string

- **AdditionalInformation**
  - Type: xs:string

**SystemUnitClass**

- **SystemUnitClassName**
  - Type: xs:string
  - Use: required

- **PropertyString**
  - Type: propertystring
  - Use: optional

- **changeMode**
  - Type: changeMode
  - Use: optional
  - Default: state

**source**

```
<xs:element name="SystemUnitClass" type="SystemUnitFamilyType" minOccurs="0" maxOccurs="unbounded"/>
```
A.18 ComplexType SystemUnitClassType

A.18.1 General

The datatype "SystemUnitClassType" defines structures for the specification of system unit classes. Each system unit requires a unique identifier stored in the field "RoleClassName".

<table>
<thead>
<tr>
<th>children</th>
<th>Description</th>
<th>Version</th>
<th>Revision</th>
<th>Copyright</th>
<th>AdditionalInformation</th>
<th>Attribute</th>
<th>AttributeValue</th>
<th>ExternalInterface</th>
<th>InternalElement</th>
<th>SupportedRoleClass</th>
<th>InternalLink</th>
</tr>
</thead>
<tbody>
<tr>
<td>used by</td>
<td>elements</td>
<td>RoleClassType/PredefinedRealisation</td>
<td>SystemUnitClassType/InternalElement/PredefinedRealisation</td>
<td>SystemUnitInstanceType/SingletonClassDescription</td>
<td>SystemUnitFamilyType</td>
<td>complexType</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

source

```xml
<xs:complexType name="SystemUnitClassType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="Attribute" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="AttributeValue" type="AttributeValueType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="ExternalInterface" type="InterfaceRequirementType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```
<xs:element name="InternalElement" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element name="RoleRequirements" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="AdditionalAttribute" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="AdditionalAttributeValue" type="AttributeValue" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="AdditionalExternalInterface" type="InterfaceRequirementType" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="PredefinedRealisation" minOccurs="0">
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="SystemUnitClassType">
              <xs:attribute name="GlobalSystemUnitClassLibName" type="xs:string" use="optional"/>
              <xs:attribute name="refBaseSystemUnitClass" type="reference" use="optional"/>
              <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
              <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="MappingObject" type="MappingType" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="localElementName" type="xs:string" use="required"/>
            <xs:element name="description" type="xs:string" use="optional"/>
            <xs:element name="PropertyString" type="propertystring" use="optional"/>
            <xs:element name="changeMode" type="changeMode" use="optional" default="state"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SupportedRoleClass" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence minOccurs="0">
            <xs:element name="MappingObject" type="MappingType" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="InternalLink" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="localLinkName" type="xs:string" use="required"/>
          <xs:attribute name="refPartnerSideA" type="reference" use="optional"/>
          <xs:attribute name="refPartnerSideB" type="reference" use="optional"/>
          <xs:attribute name="description" type="xs:string" use="optional"/>
          <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
          <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
A.18.2 Element SystemUnitClassType/Attribute

The element “Attribute” is of the datatype “AttributeType”. For a description of this datatype see A.4.2.

<table>
<thead>
<tr>
<th>type</th>
<th>AttributeType</th>
</tr>
</thead>
<tbody>
<tr>
<td>children</td>
<td>DefaultValue refSemantic Constraint Quantity Attribute</td>
</tr>
<tr>
<td>attributes</td>
<td>Name</td>
</tr>
<tr>
<td></td>
<td>LocalAttributeName</td>
</tr>
<tr>
<td></td>
<td>description</td>
</tr>
<tr>
<td></td>
<td>dimension</td>
</tr>
<tr>
<td></td>
<td>attributeDataType</td>
</tr>
<tr>
<td></td>
<td>PropertyString</td>
</tr>
<tr>
<td></td>
<td>changeMode</td>
</tr>
</tbody>
</table>

source `<xs:element name="Attribute" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>`
A.18.3 Element SystemUnitClassType/AttributeValue

The element “AttributeValue” is of the datatype “AttributeValueType”. For a description of this datatype see A.6.

<table>
<thead>
<tr>
<th>type</th>
<th>AttributeValueType</th>
</tr>
</thead>
<tbody>
<tr>
<td>children</td>
<td>Value Constraint</td>
</tr>
<tr>
<td>attributes</td>
<td>Name</td>
</tr>
<tr>
<td>refAttribute</td>
<td>reference</td>
</tr>
<tr>
<td>description</td>
<td>xs:string</td>
</tr>
<tr>
<td>dimension</td>
<td>xs:string</td>
</tr>
<tr>
<td>PropertyString</td>
<td>propertystring</td>
</tr>
<tr>
<td>changeMode</td>
<td>changeMode</td>
</tr>
</tbody>
</table>

source `<xs:element name="AttributeValue" type="AttributeValueType" minOccurs="0" maxOccurs="unbounded"/>

A.18.4 Element SystemUnitClassType/ExternalInterface

The element “ExternalInterface” may specify external interfaces of system units. The element is of the datatype “InterfaceRequirementType”. For a description of this datatype see A.10. The element shall have a field “LocalInterfaceName”. Each interface shall have an unique interface identifier. Existing interfaces may be referenced by the fields “refInterfaceClass” and “GlobalInterfaceClassLibName” when these interface classes are defined.
A.18.5 Element SystemUnitClassType/InternalElement

The element “InternalElement” allows for the definition of the internal structure of a system unit. It shall have the attribute “LocalElementName” to store the identifier of the element. The description of the element may be stored in the attribute “description”.

The “InternalElement” may have a reference to a predefined, pre-existing unit, which may be realised by the element “PredefinedRealisation”. Requirements to the defined internal elements can be made by the definition of the element “RoleRequirements”. The child element “MappingObject” allows for the mapping of user defined names to predefined names in class definitions.
A.18.6 Element SystemUnitClassType/InternalElement/RoleRequirements

The element “RoleRequirements” allows for the definition of requirements to a predefined role class definition. To reference the defined role class, it shall have the attributes “RoleClassLibName” and “refRoleClass”.

The sub elements “AdditionalAttribute”, “AdditionalAttributeValue” and “AdditionalExternalInterface” allow for the enhancement of the referenced role class definition by new attribute definitions, with new value requirements or new definitions of required external interfaces.
A.18.7 Element SystemUnitClassType/InternalElement/RoleRequirements/AdditionalAttribute

The element “AdditionalAttribute” is of the datatype “AttributeType”. For a description of this datatype see A.4.2.
A.18.8 Element
SystemUnitClassType/InternalElement/RoleRequirements/AdditionalAttributeValue

The element “AttributeValue” is of the datatype “AttributeValueType”. For a description of this datatype see A.6.

A.18.9 Element
SystemUnitClassType/InternalElement/RoleRequirements/AdditionalExternalInterface

The element “AdditionalExternalInterface” is of the datatype “InterfaceRequirementType”. For a description of this datatype see A.10. Besides the attributes of the latter datatype, an element “additionalExternalInterface” shall have the attribute “LocalInterfaceName” to identify the specified new additional external interface.
A.18.10 Element SystemUnitClassType/InternalElement/PredefinedRealisation

The element “PredefinedRealisation” is of the datatype “SystemUnitClassType”. For a description of this datatype see A.18. The Element allows for the definition as well as for the enhancement of complex units and to assign them, for example, to internal elements (see A.18.5).
type extension of SystemUnitClassType

children Description Version Revision Copyright AdditionalInformation Attribute AttributeValue ExternalInterface InternalElement SupportedRoleClass InternalLink

attributes
Name GlobalSystemUnitClassLibName refBaseSystemUnitClass PropertyString changeMode
Type xs:string reference propertystring changeMode
Use optional optional optional optional default "state"

source
<xs:element name="PredefinedRealisation" minOccurs="0">
<xs:complexType>
<xs:complexContent>
<xs:extension base="SystemUnitClassType">
<xs:attribute name="GlobalSystemUnitClassLibName" type="xs:string" use="optional"/>
<xs:attribute name="refBaseSystemUnitClass" type="reference" use="optional"/>
<xs:attribute name="PropertyString" type="propertystring" use="optional"/>
<xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
A.18.11 Element SystemUnitClassType/InternalElement/MappingObject

The element “MappingObject” is of the datatype “MappingType”. For a description of this datatype see A.11.

```
<xs:element name="MappingObject" type="MappingType" minOccurs="0"/>
```

A.18.12 Element SystemUnitClassType/SupportedRoleClass

The element “SupportedRole” allows the assignment of predefined roles to the surrounding system unit class definition. To identify a role definition in a role library the element shall have the attributes “GlobalRoleLibName” and “refRoleClass”.

```
<xs:element name="SupportedRoleClass" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element name="MappingObject" type="MappingType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="GlobalRoleLibName" type="xs:string" use="required"/>
    <xs:attribute name="refRoleClass" type="reference" use="required"/>
    <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
    <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
  </xs:complexType>
</xs:element>
```

A.18.13 Element SystemUnitClassType/SupportedRoleClass/MappingObject

The element “MappingObject” is of the datatype “MappingType”. For a description of this datatype see A.11.

```
<xs:element name="MappingObject" type="MappingType" minOccurs="0" />
```
A.18.14 Element SystemUnitClassType/InternalLink

The element “InternalLink” allows the specification of the relationships of defined internal elements. To identify the relationship, it shall have a attribute “LocalLinkName”. To reference the partner elements, it has the two optional attributes: “refPartnerSideA” and “refPartnerSideB”.

A.19 ComplexType SystemUnitFamilyType

A.19.1 General

The datatype “SystemUnitFamilyType” is an extension of the datatype “SystemUnitClassType”. For a description of this datatype see A.18. The datatype “SystemUnitFamilyType” allows for the creation of child elements of system unit classes. Child classes contain all characteristics (attribute elements) of the base class but may be extended by new definitions of characteristics (attribute elements).
type extension of SystemUnitClassType

children
Description Version Revision Copyright AdditionalInformation Attribute AttributeValue ExternalInterface
InternalElement SupportedRoleClass InternalLink SystemUnitClass

used by
SystemUnitFamilyType/SystemUnitClass SystemUnitClassLibType/SystemUnitClass

attributes
Name                  Type            Use    Default    Fixed    Annotation
SystemUnitClassName  xs:string      required
PropertyString        propertystring   optional
changeMode            changeMode      optional  state

source
<xs:complexType name="SystemUnitFamilyType">
  <xs:complexContent>
    <xs:sequence>
      <xs:element name="SystemUnitClass" type="SystemUnitClassType" maxOccurs="0..1"/>
      <xs:element name="InternalElement" type="InternalElementType" maxOccurs="0..1"/>
      <xs:element name="SupportedRoleClass" type="SupportedRoleClassType" maxOccurs="0..1"/>
      <xs:element name="InternalLink" type="InternalLinkType" maxOccurs="0..1"/>
    </xs:sequence>
  </xs:complexContent>
</xs:complexType>
A.19.2 Element SystemUnitFamilyType/SystemUnitClass

The element is of the datatype “SystemUnitFamilyType”. For a description of this datatype see A.19.
A.20 ComplexType SystemUnitInstanceType

A.20.1 General

This element allows for the specification of an instance of a plant item. To identify this instance, it shall have the required attribute "SystemUnitInstanceName". The detailed description may be stored in the attribute "description".

The sub element "SingletonClassDescription" allows the assignment of particular plant class items to an instance.

```
<xs:complexType name="SystemUnitInstanceType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="SingletonClassDescription" minOccurs="0">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="SystemUnitClassType">
            <xs:attribute name="GlobalSystemUnitClassLibName" type="xs:string" use="optional"/>
            <xs:attribute name="refBaseSystemUnitClass" type="reference" use="optional"/>
            <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
            <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="SystemUnitInstanceName" type="xs:string" use="required"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
```

```
<xs:complexType name="SystemUnitClassType">
  <xs:sequence minOccurs="0">
    <xs:element ref="Header" minOccurs="0"/>
    <xs:element name="Description" type="xs:string"/>
    <xs:element name="Version" type="xs:string"/>
    <xs:element name="Revision" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Copyright" type="xs:string"/>
    <xs:element name="AdditionalInformation" type="SystemUnitClassType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```
A.20.2 Element SystemUnitInstanceType/SingletonClassDescription

The element “SingletonClassDescription” is of the datatype “SystemUnitClassType”. For a description of this datatype see A.18. By means of this element, new class descriptions may be defined on the fly and assigned to a SystemUnitInstance.

```
<xs:element name="SingletonClassDescription" minOccurs="0">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="SystemUnitClassType">
        <xs:sequence>
          <xs:element name="Description" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="Version" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="Revision" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="Copyright" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="AdditionalInformation" type="xs:string" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```
A.21 SimpleType changeMode

A.21.1 General

The datatype “changeMode” is an extension of the XML base type string. It may have the following items: “state”, “create”, “delete”, and “change”. This means “create” serves to identify created new items, “delete” identifies deleted items, “change” to identifies changed items related to earlier data of the exchanged data and “state” shows that no change of exchanged data has taken place.

<table>
<thead>
<tr>
<th>type</th>
<th>restriction of xs:string</th>
</tr>
</thead>
<tbody>
<tr>
<td>used by</td>
<td>attributes</td>
</tr>
<tr>
<td></td>
<td>Header/AdditionalInformation/@changeMode</td>
</tr>
<tr>
<td></td>
<td>Header/Copyright/@changeMode</td>
</tr>
<tr>
<td></td>
<td>Header/Revision/@changeMode</td>
</tr>
<tr>
<td></td>
<td>Header/Version/@changeMode</td>
</tr>
<tr>
<td></td>
<td>Header/Description/@changeMode</td>
</tr>
<tr>
<td></td>
<td>MappingType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>MappingType/InterfaceNameMapping/@changeMode</td>
</tr>
<tr>
<td></td>
<td>AttributeValueRequirementType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>AttributeType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemHierarchyLibType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitInstanceType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitInstanceType/SingletonClassDescription/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitClassLibType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitFamilyType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/SupportedRoleClass/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/InternalElement/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/InternalElement/PredefinedRealisation/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/InternalElement/RoleRequirements/@changeMode</td>
</tr>
<tr>
<td></td>
<td>RoleClassLibType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>RoleClassType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>InterfaceRequirementType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>InterfaceClassLibType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>InterfaceClassType/@changeMode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>facets</th>
<th>enumeration state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enumeration create</td>
</tr>
<tr>
<td></td>
<td>enumeration delete</td>
</tr>
<tr>
<td></td>
<td>enumeration change</td>
</tr>
</tbody>
</table>

source

<xs:simpleType name="changeMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="state"/>
    <xs:enumeration value="create"/>
    <xs:enumeration value="delete"/>
    <xs:enumeration value="change"/>
  </xs:restriction>
</xs:simpleType>

A.22 SimpleType propertystring

A.22.1 General

The datatype “propertystring” is an extension of the XML base type string. It serves for further developments. Similar to the definition of changeMode, it can contain multiple enumerations or structured information.

<table>
<thead>
<tr>
<th>type</th>
<th>xs:string</th>
</tr>
</thead>
<tbody>
<tr>
<td>used by</td>
<td>attributes</td>
</tr>
<tr>
<td></td>
<td>Header/AdditionalInformation/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>Header/Copyright/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>Header/Revision/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>Header/Version/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>Header/Description/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>MappingType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>MappingType/InterfaceNameMapping/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>AttributeValueRequirementType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>AttributeType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemHierarchyLibType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemUnitInstanceType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemUnitInstanceType/SingletonClassDescription/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemUnitClassLibType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemUnitFamilyType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/@PropertyString</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/SupportedRoleClass/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/InternalElement/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/InternalElement/PredefinedRealisation/@changeMode</td>
</tr>
<tr>
<td></td>
<td>SystemUnitType/InternalElement/RoleRequirements/@changeMode</td>
</tr>
<tr>
<td></td>
<td>RoleClassLibType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>RoleClassType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>InterfaceRequirementType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>InterfaceClassLibType/@changeMode</td>
</tr>
<tr>
<td></td>
<td>InterfaceClassType/@changeMode</td>
</tr>
</tbody>
</table>
### A.23 SimpleType reference

#### A.23.1 General

The datatype “reference” is an extension of the XML base type string.

<table>
<thead>
<tr>
<th>type</th>
<th>xs:string</th>
</tr>
</thead>
<tbody>
<tr>
<td>used by</td>
<td>attributes</td>
</tr>
<tr>
<td></td>
<td>AttributeType/refSemantic/@correspondingAttributeName AttributeType/Quantity/@quantityName AttributeType/Value/@refAttribute SystemUnitClassType/InternalElement/PredefinedRealisation/@refBaseSystemUnitClass SystemUnitInstanceType/SingletonClassDescription/@refBaseSystemUnitClass InterfaceRequirementType/@refInterfaceClass SystemUnitClassType/InternalLink/@refPartnerSideA SystemUnitClassType/InternalLink/@refPartnerSideB SystemUnitClassType/InternalElement/RoleRequirements/@refRoleClass SystemUnitClassType/SupportedRoleClass/@refRoleClass</td>
</tr>
</tbody>
</table>

| source | <xs:simpleType name="reference">  
| <xs:restriction base="xs:string"/> | <</xs:simpleType> |
Annex B
(informative)

Examples for PCE requests

Figure 21 – Local level indication, 1 process connection

Figure 22 – Local level indication, 2 process connections

Figure 23 – Local flow indication

Figure 24 – Local pressure indication

Figure 25 – Local temperature indication
Figure 26 – Local control panel, pressure indication, alarm high

Figure 27 – Local temperature indication, CCR temperature high alarm

Figure 28 – Local pressure indication, CCR pressure high alarm and switch

Figure 29 – CCR flow indication, device information: Orifice Plate

Figure 30 – CCR pressure indication, low, low low and high alarm
Figure 31 – CCR temperature indication and registration

Figure 32 – CCR level indication and registration, 1 process connection

Figure 33 – CCR level indication, 2 process connections

Figure 34 – Two flow indications and flow ratio control in CCR
Figure 35 – CCR flow indication and high alarm, flow control, control valve with extra interlock and open/close indication

Figure 36 – Local pressure indication, CCR pressure indication, high alarm and high high safety relevant switch

Figure 37 – Local pressure indication, CCR pressure indication, alarms and switches

Figure 38 – CCR pressure indication, high and low alarm, safety relevant switch action on on/off valve
Figure 39 – Switched valve with on/off indication and switching action, safety relevant switched valve

Figure 40 – Pressure restriction

Figure 41 – Flow restriction
Figure 42 – PT compensated flow control, safety relevant pressure switch (two out of three (2oo3) shutdown), switched control valve with on/off indication and switching action at open position.

Figure 43 – CCR temperature control, additional manual switch actions from CCR with indication and local control panel.
Figure 44 – Motor typical, local on/off control, CCR off control, current, fault with alarm and running indication

Figure 45 – Multivariable controller
Figure 46 – On/off valve with position indication

Figure 47 – On/off valve with safety relevant switch and position indication

Figure 48 – Level control with continuous controller

Figure 49 – Level control with on/off switch
Figure 50 – Cascade control for temperature as control input, flow control as follow-up controller

Figure 51 – Safety directed high control to a subsequent valve, manual control for reset function and manual control for manual/automatic switch of the valve, valve with open/close indication and safety relevant switch to subsequent valve

Figure 52 – Flow control in CCR

Figure 53 – Temperature control with high alarm and high switch
Figure 54 – Manual control from CCR

Figure 55 – Flow measurement with display and alarms in CCR, high high switch on process control function and switch on/off valve

Figure 56 – Local P-/F-/T-/S- control without auxiliary power (stand-alone)
Annex C
(normative)

CAEX Model

C.1 Full XML schema of the CAEX model

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- CAEX - Computer Aided Engineering Data-Exchange-Metamodel -->
<!-- Version 1.0.10; 26.01.2004-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <xs:simpleType name="changeMode">
    <xs:restriction base="xs:string">
      <xs:enumeration value="state"/>
      <xs:enumeration value="create"/>
      <xs:enumeration value="delete"/>
      <xs:enumeration value="change"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="reference">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:simpleType name="propertystring">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:group name="Header">
    <xs:sequence>
      <xs:element name="Description" minOccurs="0">
        <xs:complexType>
          <xs:simpleContent>
            <xs:extension base="xs:string">
              <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
              <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
            </xs:extension>
          </xs:simpleContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="Version" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="RevisionDate" type="xs:dateTime"/>
            <xs:element name="OldVersion" type="xs:string" minOccurs="0"/>
            <xs:element name="NewVersion" type="xs:string" minOccurs="0"/>
            <xs:element name="AuthorName" type="xs:string"/>
            <xs:element name="Comment" type="xs:string" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
          <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="Copyright" minOccurs="0">
        <xs:complexType>
          <xs:simpleContent>
            <xs:extension base="xs:string">
              <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
              <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
            </xs:extension>
          </xs:simpleContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="AdditionalInformation" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:simpleContent>
            <xs:extension base="xs:string">
              <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
              <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
            </xs:extension>
          </xs:simpleContent>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:group>
</xs:schema>
```
<xs:complexType>
  <xs:attribute name="title" type="xs:string" use="required"/>
  <xs:attribute name="content" type="xs:string" use="optional"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:group>
<xs:complexType name="InterfaceClassType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="InterfaceClassLibType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="InterfaceRequirementType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="RoleClassType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:complexContent>
</xs:complexType>
</xs:complexType>
<xs:complexType name="RoleFamilyType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="SystemUnitFamilyType">
  <xs:complexContent>
    <xs:extension base="SystemUnitClassType">
      <xs:sequence minOccurs="0">
        <xs:element name="SystemUnitClass" type="SystemUnitFamilyType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="SystemUnitClassName" type="xs:string" use="required"/>
      <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
      <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="SystemUnitClassLibType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="SystemUnitClass" type="SystemUnitFamilyType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="GlobalSystemUnitLibName" type="xs:string" use="required"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>

<xs:complexType name="SystemUnitInstanceType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="SingletonClassDescription" minOccurs="0">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="SystemUnitClassType">
            <xs:attribute name="GlobalSystemUnitClassLibName" type="xs:string" use="optional"/>
            <xs:attribute name="refBaseSystemUnitClass" type="reference" use="optional"/>
            <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
            <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="SystemUnitInstanceName" type="xs:string" use="required"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>

<xs:complexType name="SystemHierarchyElementType">
  <xs:complexContent>
    <xs:extension base="SystemUnitInstanceType">
      <xs:sequence minOccurs="0">
        <xs:element name="SystemHierarchyElement" type="SystemHierarchyElementType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="SystemHierarchyLibType">
  <xs:sequence minOccurs="0">
    <xs:group ref="Header" minOccurs="0"/>
    <xs:element name="SystemHierarchyElement" type="SystemHierarchyElementType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="GlobalSystemHierarchyName" type="xs:string" use="required"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="PropertyString" type="propertystring" use="optional"/>
  <xs:attribute name="changeMode" type="changeMode" use="optional" default="state"/>
</xs:complexType>

<xs:complexType name="AttributeType">
  <xs:sequence minOccurs="0">
    <xs:element name="DefaultValue" type="xs:anyType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

Annex D
(informative)

Bibliography

ISO 14617-6:2002-09 Graphical symbols for diagrams – Part 6: Measurement and control functions

ISA-5.1-1984 – (R1992), Instrumentation Symbols and Identification (available at www.isa.org)