General Modifications of Proteus Schema

|  |  |  |  |
| --- | --- | --- | --- |
| 0.1 | 2015-12-09 | Manfred Theißen, AixCAPE | initial version |
| 0.2 | 2016-01-18 | Manfred Theißen, AixCAPE | * note about optional ID for Node element added * proposal for new Project element added * minor typos fixed |
| 0.3 | 2016-01-28 | Manfred Theißen, AixCAPE | * fixed a headline * implementation notes and future tasks added where applicable |
| 0.4 | 2016-04-22 | Manfred Theißen, AixCAPE | * replace Project introduced in 0.2 with Metadata * PlantStructureItem added |
| 0.5 | 2017-05-04 | ManfredTheißen, AixCAPE | * Extent made optional everywhere |

This document lists proposals to solve some issues related to Proteus P&ID Profile Schema 3.6.0 encountered in the DEXPI project. These proposals have been implemented in Proteus P&ID Profile Schema 4.0.1.

# Remove defaults for attributes FlowIn and FlowOut of ConnectionPoints

**Problem**

The attributes FlowIn and FlowOut of the ConnectionPoints element have default values “1” and “2”, respectively. The existence of default values makes it impossible to cover cases when there is no flow in node or no flow out node in a clean way.

**Example**

An item that has a flow out node, but no flow in node (e.g., a CrossPageConnection) would have ConnectionPoints as follows:

<ConnectionPoints NumPoints="2" FlowOut="1">  
 <Node>  
 <!-- This Node (index 0) represents the item itself. -->  
 </Node>  
 <Node>  
 <!-- This Node (index 1) is the flow out. -->  
 </Node>  
 </ConnectionPoints>

Due to the default value given in the XML schema, the Node with index 1 is also a flow in.

**Proposal**

The default values for FlowIn and FlowOut should be removed.

*Current schema:*

<xsd:element name="ConnectionPoints">  
...  
 <xsd:attribute default="1" name="FlowIn" type="xsd:positiveInteger" use="optional"/>  
 <xsd:attribute default="2" name="FlowOut" type="xsd:positiveInteger" use="optional"/>  
...  
</xsd:element>

*Modified schema:*

<xsd:element name="ConnectionPoints">  
...  
 <xsd:attribute  
 name="FlowIn" type="xsd:positiveInteger" use="optional"/>  
 <xsd:attribute  
 name="FlowOut" type="xsd:positiveInteger" use="optional"/>  
...  
</xsd:element>

**Implementation Note**

Implemented as described above.

# Add ID attribute to Node element

**Problem**

The Node element has no ID attribute. In consequence, references to a certain Node are difficult, e.g., in case a Text element refers to an attribute of the Node (“DependantAttribute”). Alternatives currently allowed (e.g., PersistentIDs) add additional complexity and are in general ambiguous.

**Proposal**

The Node element should have an ID attribute.

*Current schema:*

<xsd:element name="Node">  
...  
</xsd:element>

*Modified schema:*

<xsd:element name="Node">  
...  
 <xsd:attribute name="ID" type="xsd:ID" use="required"/>  
...  
</xsd:element>

The attribute is declared as “required” in analogy to the ID attribute of PlantItem, and thus in favor of *consistency* of the overall schema.

However, an “optional” attribute would also solve the problem. An optional ID would also be *compatible* with the prior Proteus version in the sense that Proteus 3.6.0 files would be formally valid.

**Implementation Note**

Implemented as “required” attribute.

# Add MetaData element

**Problem**

No element for arbitrary metadata available.

**Proposal**

Add Project element as subclass of PlantItem.

**Implementation Note**

Added as an optional element after <PlantInformation> in <PlantModel>.

**Future Tasks**

Align <MetaData> with the existing after <PlantInformation> in Proteus. Check if MetaData as subclass of PlantItem is really a good choice, but keep in mind that MetaData must have an “ID” and GenericAttributes (both currently inherited from PlantItem).

# Add PlantStructureItem element

**Problem**

No element for plant structure available.

**Proposal**

Add PlantStructureItem element as subclass of PlantItem.

**Implementation Note**

Added as an optional element (0..\*) after <MetaData> in <PlantModel>.

**Example**

Plant PP\_ABC\_PLANT contains a section PS\_UTILITIES:

<PlantStructureItem ID=”PS\_UTILITIES” ...>

...

<Association Type=”is a part of” ItemID=”PP\_ABC\_PLANT”/>

</PlantStructureItem>

<PlantStructureItem ID=”PP\_ABC PLANT” ...>

...

<Association Type=”is a collection including”

ItemID=”PS\_UTILITIES”/>

</PlantStructureItem>

Section PS\_UTILITIES contains an equipment P4712:

<PlantStructureItem ID=”PS\_UTILITIES” ...>

...

<Association Type=”is a collection including” ItemID=”P4712”/>

</PlantStructureItem>

<Equipment ID=”P4712” ...>

...

<Association Type=”is a part of” ItemID=”PS\_UTILITIES”/>

</Equipment>

**Future Tasks**

Check if PlantStructureItem as subclass of PlantItem is really a good choice, but keep in mind that PlantStructureItem must have an “ID” and GenericAttributes (both currently inherited from PlantItem).

# Make Extent optional

**Problem**

<Extent> element

* is used inconsistently without clear rationale (sometimes required (e.g., in CenterLine) sometimes optional (e.g., AnnotationItem)),
* has unclear semantics (idealized graphical extent vs. extent including line width?, …),
* is not required for data exchange.

**Proposal**

Make <Extent> element optional everywhere.

**Implementation Note**

<Extent> element has been made optional in those elements in which it was required (CenterLine, Circle, etc.).

**Future Tasks**

Check if <Extent> can be removed completety. If not, fix problems given above. Note: Extent may be useful to give the overall dimensions of a drawing.