ebIX® Rules for use of OCL constraints to tailor BIE’s to Business Requirements

Status: Approved by ETC
Version: 1.0
Revision: C
Date: October, 2015
## Contents

A. Introduction .................................................................................................................. 4

A.1. Versions, updates ......................................................................................................... 4

A.2. References .................................................................................................................. 4

A.2.1. International open standards used: ........................................................................ 4

A.2.2. ebIX® Document references ................................................................................ 5

1. Why constraints for tailoring BIE’s and BDT’s ............................................................ 5

2. Method used .................................................................................................................. 5

2.1. How to tailor BIE’s and BDT’s ................................................................................ 5

2.2. Using constraints for national customization ............................................................ 6

2.3. Stereotypes for constraints ....................................................................................... 7

2.3.1. <<payload>> ........................................................................................................... 7

2.3.2. <<document>> ....................................................................................................... 7

2.3.3. <<abie>> ................................................................................................................ 7

2.3.4. <<bdt>> .................................................................................................................. 7

2.3.5. <<dependency>> ................................................................................................... 7

3. Transformation ............................................................................................................. 9

3.1. Derivation of XML schemas. ................................................................................... 9

4. OCL-statements leading to the XML-element Choice ...................................................... 10

4.1. Example of business requirements for option A ....................................................... 10

4.1.1. Example of a choice between 2 associated classes (red arrows) ......................... 11

4.1.1.1. Constraint belonging to the example ................................................................. 11

4.1.2. Example of a choice between 3 attributes (green arrows) .................................. 12

4.1.2.1. Constraint belonging to the example ................................................................. 12

4.1.3. XML-schema resulting from both constraints (red and green): ......................... 13
4.2. Example for option B: choice between <<MA>>’s (yellow arrow) ........................................ 14
4.2.1. Constraint belonging to the example ............................................................................ 14
4.3. On-the-fly creation of code lists ...................................................................................... 14
5. Overview of OCL-statements for constraints and their mapping to XSD-lines (in an XML schema) .................................................................................................................................................................................. 15
6. Example: Collected Data (Business Information, Message Assembly) .............................. 16
6.1. OCL constraints for CollectedData <<MA>> ........................................................................ 16
Annex A: Example of specifications used for the testing of the ebIX® transformation tool (version per May 2015) .................................................................................................................................................................................. 20
A. Introduction
This document describes the way in which in ebIX® models Core Components (BIE’s and BDT’s) are to be tailored to the business requirements by means of UML constraints built from OCL statements.

A.1. Versions, updates

<table>
<thead>
<tr>
<th></th>
<th>Old</th>
<th>New</th>
<th>Clarification</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start of updates for Version 1r0-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td></td>
<td></td>
<td>Since this is the first version of the document that is published, there is no specification of changes.</td>
<td></td>
</tr>
<tr>
<td>End of updates for Version 1r0-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A.2. References

A.2.1. International open standards used¹:


¹ The versions and websites in the references reflect the status on the moment of creation of this document.

A.2.2. ebIX® Document references
[8] Introduction to ebIX® Models (www.ebix.org)
[9] The Harmonized Role Model (for the Electricity Market) by ebIX®, ENTSO-E, and EFET (www.ebix.org)

1. Why constraints for tailoring BIE’s and BDT’s
The concept of Core Components is aimed at re-use. BIE’s and BDT’s are created to be used more than once. This implies, that these BIE’s and BDT’s are too widely specified for each individual instance in which these are used and therefore will have to be tailored to the actual business requirements for that occasion.

The ebIX® policy is to use where possible open international standards as a basis for its work. Therefore and also in line with the UN/CEFACT Core Component Technical Specification (CCTS) ebIX® has decided to use for the tailoring of BIE’s and BDT’s the concept of standard UML constraints with statements in a formalized language. The language chosen is the Object Constraint Language version 2.0. OCL is an OMG standard (as is UML).

2. Method used
The method used has to take into account three requirements:
- The method shall obey the open international standards chosen ([3] and [7])
- The method shall be fit for tailoring BIE’s and BDT’s
- The method shall allow the derivation of technology dependent models (such as XML schemas or web service definitions or any other syntax for that matter) from the technology independent model for the European Energy Market as developed by ebIX®.

2.1. How to tailor BIE’s and BDT’s
The constraints for tailoring BIE’s or BDT’s are placed under the class that is stereotyped <<MA>> (Message Assembly). In UML terms: the constraint is owned by the <<MA>>-class. The constraint is of type “invariant” and will therefore have no name. The language used for the statements is OCL 2.0.

This <<MA>> is always specified as the “Constrained Element”. The visibility of the constraint is “public”.

In the constraints in the ebIX® model only invariant statements are used (starting with “inv: ”)².

An example of a customized version of a message assembly (<<MA>>) is found in paragraph 6.

² In MagicDraw for the first line the text “inv: “ is to be skipped, because for this first line the invariant character of the statement is assumed to be default because the specification of the constraint as invariant.
For the transformation of just the payload other constraints are to be used than for the derivation of a complete document (including header information). Therefore ebIX® has specified 5 stereotypes for the constraints. These are again to be subdivided into 2 categories. On the one hand <<abie>> which is used to classify constraints that intend to specify the multiplicity, <<bdt>> which is used to specify the structure and default or potential values for a data type and <<dependency>> which specifies dependencies between classes or properties. On the other hand the stereotypes <<payload>> and <<document>> provide the transformation tool the information which constraint to use for the creation of just a payload or a complete document. A constraint with stereotype <<Payload>> is used to create an XML schema with just the payload information, use of the constraint with stereotype <<Document>> leads on the contrary to a XML schema including the header etc. Consequentially the constraints for the root class with stereotype <<MA>> may get several stereotypes: for example both <<abie>> and <<payload>>.

2.2. Using constraints for national customization
The ebIX® model is open for national customization of business documents (stereotyped <<MA>>). ebIX® supports these national customizations in so far as the customization starts from the same <<MA>> with its <<ABIE>>-cluster and therefore the customization regards changes in the OCL-constraints only. For this purpose the stereotypes for constraints have 2 properties that make it possible to specify the responsible agency for the customization. The property ResponsibleAgency has a default value 260 (for ebIX®). For national customization the property CodeListIdentifier shall get the value for the national organization responsible for the national customization.
2.3. Stereotypes for constraints

2.3.1. <<payload>>
Web services will start from the payload. Payload information is composed of one or more <<ABIE>>’s. This payload has to be tailored to business requirements: some included <<ASBIE>>’s are and others are not used; cardinalities for <<BBIE>>’s have to be redefined; data types have to be redefinable; for some data types a certain code value has to be specified; and finally dependencies may have to be defined.

2.3.2. <<document>>
In case of information exchange by means of a self-contained message (like EDIFACT messages) the payload will have to be “packed” within header and trailer information. This header information also will have to be constructed from <<ABIE>>’s so in principle the same remarks for the tailoring apply here and for the payload (see above).

2.3.3. <<abie>>
Constraints that are stereotyped <<abie>> will be used for re-specifying the structure of the <<ABIE>>. This comes down to changing the cardinality of attributes.

2.3.4. <<bdt>>
Constraints that are stereotyped <<bdt>> will be used for re-specifying the structure of the <<BDT>> or for specifying allowed values for attributes.

2.3.5. <<dependency>>
Constraints that are stereotyped <<dependency>> will be used for specifying dependencies between UML elements. For instance specifying that the appearance of one attribute excludes the...
appearance of another attribute (leading to the use of choice in an XML schema). The stereotype <<dependency>> will be used mostly for <<ABIE>>’s directly (and not for the tailoring of documents). However in case of the content of an <<InfEnvelope>> consisting of either a positive or a negative response, the <<dependency>> is used for the specification of an information flow in the Choreography View.
3. Transformation

As already stated in paragraph 1 the objective of using OCL statements is twofold:
   1. Make a model, that is built from generic BIE’s, specific for particular business requirements;
   2. Do this:
      a. while staying in line with UN/CEFACT recommendations;
      b. and by using an open international standard (OCL)

3.1. Derivation of XML schemas.

To support the XML schemas for the payload and the document the following XML schemas are generated from the UML model. These supporting XML schemas are subsequently included in the payload- and document-schemas.

- XML schemas for code lists (enumerations)
- XML schema specially created by UN/CEFACT for restricted usages of ISO 8601 date/time formats (XBT)
- XML schemas for generic ebIX® CC’s (<<ABIE>> and <<BDT>>)
- XML schemas for customized ebIX® CC’s (<<MBIE>> and <<MDT>>) belonging to one particular <<MA>>

There is a theoretical need for XML schemas for UN/CEFACT CC’s (ACC and CDT) to be imported in the ebIX® schemas for ABIE’s and for BDT’s. Since this need is only theoretical, this import is not used for the ebIX® schemas.

The XML schema for special usages of date/time is not derived from the model, but imported from UN/CEFACT (XBT XML-schema).

ebIX® uses assembled code lists (in UML specified as an empty enumeration which inherits from several other enumerations that do contain literals). According to UN/CEFACT Naming and Design Rules for XML this kind of construction may in XML lead to an enumeration that includes other enumerations. Or to a qualified data type that combines these enumerations (or offers a choice from several enumerations). ebIX® has chosen the option of creating enumerations that include the literals from other enumerations.
4. OCL-statements leading to the XML-element Choice

ebIX® needs in addition to the constraints for tailoring BIE’s and BDT’s also another type of constraint. This type of constraint has to handle dependencies.

In the ebIX® model the dependency mostly appears as:

A. A dependency applied to a class stereotyped as <<MA>> used in the Information View to represent the root class for a business document:
   - As a dependency between ASBIE’s;
   - As a dependency between BBIE’s;

B. A dependency applied to a class stereotyped <<InfEnvelope>> used in the Choreography View to represent an information flow, when this flow can either contain a positive response or a negative response

In the XML schemas the dependency appears as a choice.

4.1. Example of business requirements for option A

Below you find an example of a class diagram specifying business requirements that lead to dependencies which in turn lead to the use of choice in the resulting XML schema.

The red arrow highlights an example of a dependency between associated classes.

The green arrow highlights an example of a dependency between attributes within a class.
4.1.1. Example of a choice between 2 associated classes (red arrows)

4.1.1.1. Constraint belonging to the example

context: Energy_TimeSeries inv:
-- Dependency Energy_TimeSeries
-- Metering_Installation & Generation_Installation

self.Metering_Installation->notEmpty() implies Generation_Installation->size()=0
inv: self.Generation_Installation->notEmpty() implies Metering_Installation->size()=0
4.1.2. Example of a choice between 3 attributes (green arrows)

4.1.2.1. Constraint belonging to the example
context: Energy_Observation inv:³

-- Dependency Energy_Observation
-- Energy_Quality & Generated_Quality & Quantity_Missing
self.Energy_Quantity->notEmpty() implies Generated_Quantity->size()=0 and Quantity_Missing->size()=0
inv: self.Generated_Quantity->notEmpty() implies Energy_Quantity->size()=0 and Quantity_Missing->size()=0
inv: self.Quantity_Missing->size()=1 implies Energy_Quantity->size()=0 and Generated_Quantity->size()=0
inv: self.Quantity_Missing->size()=1 implies Quantity_Missing = true

³ The constraint in this example is somewhat complicated by the fact that there are basically two dependencies to be solved: first the dependency between “Energy_Quality” and “Generated_Quality”, but secondly on top of this also the dependency between “Energy_Quality” and “Quantity_Missing”. And the first and second dependency are also mutually dependent.
4.1.3. **XML-schema resulting from both constraints (red and green):**
4.2. **Example for option B: choice between <<MA>>’s (yellow arrow)**

Below you find an example of a class diagram specifying an information flow in the model. This flow consists either of a positive response to a request or of a negative response to that request. This leads to a dependency which in turn lead to the use of choice in the resulting XML schema.

The yellow arrow highlights this dependency between message assemblies (<<MA>>).

![Class Diagram Example]

4.2.1. **Constraint belonging to the example**

<table>
<thead>
<tr>
<th>#</th>
<th>Specification</th>
<th>Constrained Element</th>
<th>Applied Stereotype</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dependency Response Request Collected Data</td>
<td>Response Request Collected Data</td>
<td>dependency [Constraint]</td>
</tr>
<tr>
<td></td>
<td>self!Positive-&gt;size()=1 implies Negative-&gt;size()=0</td>
<td></td>
<td>invariant [Constraint]</td>
</tr>
<tr>
<td></td>
<td>inv: self!Negative-&gt;size()=1 implies Positive-&gt;size()=0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A dependency applied to a class stereotyped <<InfEnvelope>> does not result in a specific XML schema for a business document.

4.3. **On-the-fly creation of code lists**

Restriction of possible values for an attribute is possible through an “on-the-fly” created code list. On-the-fly means, that a set of coded values is created by OCL-statements in a constraint (as opposed to specifying which coded value from an existing code list may be used in a particular instance). On-the-fly created code lists are in the ebIX® model however only allowed for the specification of a pattern for Resolution (in measured data).

---

4 This kind of dependency will however affect the specification of a web service
5. **Overview of OCL-statements for constraints and their mapping to XSD-lines (in an XML schema)**

The ebIX® Transformation Tool works on the basis of mapping predefined text strings in the XMI-export to predefined results in the textual definition of an XML schema.

The specification of this mapping is found in Annex A: Example of specifications used for the testing of the ebIX® transformation tool.
6. Example: Collected Data (Business Information, Message Assembly)

6.1. OCL constraints for CollectedData <<MA>>

<table>
<thead>
<tr>
<th>#</th>
<th>Specification</th>
<th>Constrained Element</th>
<th>Applied Stereotype</th>
</tr>
</thead>
</table>
| 1 | -- Structure MeasuredData_Request  
-- Request Collected Data  
  self.Payload.Function->size()=0  
  inv: self.Payload.MeterReadingOrigin_Type->size()=0  
  inv: self.Payload.BalanceResponsible_Involved->size()=0  
  inv: self.Payload.TransportCapacityResponsible_Involved->size()=0  
  inv: self.Payload.BalanceSupplier_Involved->size()=0  
  inv: self.Payload.Request_Period.Resolution_Duration->size()=0  
  inv: self.Payload.Request_Period.Start->size()=1  
  inv: self.Payload.Request_Period.End->size()=1 | RequestCollectedData | Invariant [Constraint]  
Document [Constraint]  
Payload [Constraint] |
## Rules for the use of OCL statements

### Specification

<table>
<thead>
<tr>
<th>#</th>
<th>Specification</th>
<th>Constrained Element</th>
<th>Applied Stereotype</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>inv: self.Payload.Request_Period.Continuous -&gt; size() = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Product_Included -&gt; size() = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.MeteringPoint_Used -&gt; size() = 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.InArea_Used -&gt; size() = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.OutArea_Used -&gt; size() = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.MeteringGridArea_Used -&gt; size() = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.BalanceGroup_Used -&gt; size() = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Metering_Installation.Meter_Identification -&gt; size() &lt;= 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Metering_Installation.Register_Installation -&gt; forAll (Register_Identification -&gt; size() = 1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Metering_Installation.Register_Installation.Meter_Read -&gt; isEmpty()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-- BDT MeasuredData_Request</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-- Request Collected Data</td>
<td></td>
<td>RequestCollectedData</td>
</tr>
<tr>
<td></td>
<td>self.Payload.Request_Period.Start.timeZoneCode -&gt; size() = 0</td>
<td></td>
<td>inv</td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Request_Period.End.timeZoneCode -&gt; size() = 0</td>
<td></td>
<td>BDT</td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Metering_Installation-&gt;forall (Meter_Identification.schemeIdentifier -&gt; size() = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Payload.Metering_Installation.Register_Installation-&gt;forall (Register_Identification.schemeIdentifier -&gt; size() = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-- Structure Energy_Document</td>
<td></td>
<td>RequestCollectedData</td>
</tr>
<tr>
<td></td>
<td>-- Request Collected Data</td>
<td></td>
<td>inv</td>
</tr>
<tr>
<td></td>
<td>self.Header.Document_Type -&gt; size() = 1</td>
<td></td>
<td>Rare</td>
</tr>
<tr>
<td></td>
<td>inv: self.Header.Sender.Role -&gt; size() = 0</td>
<td></td>
<td>invariant</td>
</tr>
<tr>
<td>#</td>
<td>Specification</td>
<td>Constrained Element</td>
<td>Applied Stereotype</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>
| 4 | -- BDT Energy_Document  
-- Request Collected Data  
  self.Header.Document_Type.content=DocumentNameCode::E72  
  inv: self.Header.Document_Type.listIdentifier->size()=0  
  inv: self.Header.Creation.timeZoneCode->size()=0  
document [Constraint]  
invariant [Constraint] |
| 5 | -- Structure Energy_Context  
-- Request Collected Data  
document [Constraint]  
abie [Constraint] |
| 6 | -- BDT Energy_Context  
-- Request Collected Data  
invariant [Constraint]  
document [Constraint] |
<table>
<thead>
<tr>
<th>#</th>
<th>Specification</th>
<th>Constrained Element</th>
<th>Applied Stereotype</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>-- Structure Document</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-- Request Collected Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>self.Header-&gt;size()=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Process-&gt;size()=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Proposed-&gt;size()=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-- Structure Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-- Request Collected Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>self.Header-&gt;size()=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Process-&gt;size()=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>inv: self.Proposed-&gt;size()=0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Annex A: Example of specifications used for the testing of the ebIX® transformation tool (version per May 2015)

<table>
<thead>
<tr>
<th>Constrained element</th>
<th>constraint</th>
<th>Result in XML Schema</th>
</tr>
</thead>
</table>
| `<ABE>` Customer_Party  
  `<ABE>` Identification : Party_identifierType [0..1]  
  `<ABE>` Name : TextType [0..1] | -- constraint nr 1  
  self.Payload.Identification->size()=1  
  inv: self.Payload.Name->size()=0 |  
  `<xsd:complexType name="CustomerParty">`  
  `<xsd:sequence>`  
  `<xsd:element name="Identification" type="bie:PartyIdentifierType_000112" minOccurs="1" maxOccurs="1"/>`  
  `<xsd:element name="Name" type="bie:TextType" minOccurs="1" maxOccurs="1"/>`  
  `</xsd:sequence>`  
  `</xsd:complexType>` |

Remark:  
The OCL statement ->size()=1 or ->size()=0 is to be used for singular multiplicity (0..1)

| `<ABE>` Domain_Location  
  `<ABE>` Identification : Domain_identifierType [1] | -- constraint nr 2  
  self.one.Identification->size()<=1 |  
  `<xsd:complexType name="DomainLocation">`  
  `<xsd:sequence>`  
  `<xsd:element name="Identification" type="bie:DomainIdentifierType_000122" minOccurs="1" maxOccurs="1"/>`  
  `</xsd:sequence>`  
  `</xsd:complexType>` |

Remark:  
- only “<” is recognized; “>” is not  
- the OCL statement is recognized and processed. But in this (test) case the result remains a cardinality 1..1, since that was the original specification that cannot be overwritten with values that go beyond it (0..1 may become 0, but 1 cannot become 0..1)
Remark:
The OCL statement ->notEmpty() or ->isEmpty() are to be used for plural multiplicity (0..n or 1..n)
### ebIX® Rules for the use of OCL statements

<table>
<thead>
<tr>
<th>See above</th>
<th>--constraint nr 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>self.two.Observation_Interval- &gt;forAll(Sequence-&gt;size()=1)</td>
</tr>
<tr>
<td></td>
<td>inv: self.two.Observation_Interval- &gt;forAll(Observation_Detail-&gt;size()=0)</td>
</tr>
</tbody>
</table>

**From Observation_Period:**

```xml
<xsd:complexType name="ObservationPeriod">
  <xsd:sequence>
    <xsd:element name="Sequence" type="bie:OrdinalType_000123" minOccurs="1" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

<table>
<thead>
<tr>
<th>See above</th>
<th>--constraint nr 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>self.three.Quantity_Missing=true</td>
</tr>
</tbody>
</table>

**From Energy_Observation:**

```xml
<xsd:element name="QuantityMissing" type="bie:IndicatorType_000099" fixed="true" minOccurs="0" maxOccurs="1"/>
```

<table>
<thead>
<tr>
<th>See above</th>
<th>--constraint nr 6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>self.two.Function.content=DocumentFunctionCode::9</td>
</tr>
</tbody>
</table>

**From Energy_TimeSeries:**

```xml
<xsd:element name="Function" minOccurs="0" maxOccurs="1">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:restriction base="bie:DocumentFunctionCodeType_000118">
        <xsd:enumeration value="9"/>
      </xsd:restriction>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```
constraint nr 7

self.three.MeterTimeFrame_Type.content=MeterTimeFrameCode::E10 or
self.three.MeterTimeFrame_Type.content=MeterTimeFrameCode::E11

From Energy_Observation:

- `<xsd:element name="MeterTimeFrameType" minOccurs="0" maxOccurs="1">`
- `<xsd:simpleContent>`
- `<xsd:restriction base="bie:MeterTimeFrameCodeType_000126">`
  - `<xsd:enumeration value="E10"/>`
  - `<xsd:enumeration value="E11"/>`
- `<xsd:simpleContent>`
- `<xsd:complexType>`
- `<xsd:element name="MeterTimeFrameType" type="bie:MeterTimeFrameCodeType_000126" minOccurs="0" maxOccurs="1"/>

constraint nr 8 has been removed and is replaced by

constraint nr 20

--constraint nr 8

self.five.MeteringPoint_Type->size()=1 implies self.five.MeteringMethod_Type->size()=0 and
self.five.SettlementMethod_Type->size()=0

inv: self.five.MeteringMethod_Type->size()=1 implies
self.five.MeteringPoint_Type->size()=0 and self.five.SettlementMethod_Type->size()=0

inv: self.five.SettlementMethod_Type->size()=1 implies
self.five.MeteringMethod_Type->size()=0 and self.five.MeteringPoint_Type->size()=0

Remark:
The multiplicity for neither one of the three properties included in the "implies"-statements is
redefined and as a consequence remains the same as specified in the ABIE: optional.

As a consequence the choice itself becomes optional.
See above

Remark:
The multiplicity for “VoltageLevel_Type” and for “StandardLoadProfile_Type” are defined outside the “implies”-statement. Both become required. As a consequence the choice becomes required.

--constraint nr 10
inv: self.five.VoltageLevel_Type->size()=1
inv: self.five.StandardLoadProfile_Type->size()=1 implies
self.five.StandardLoadProfile_Type->size()=0
inv: self.five.StandardLoadProfile_Type->size()=1 implies
self.five.VoltageLevel_Type->size()=0

From Measurement_MeteringPoint_Characteristics:
<xsd:choice
  minOccurs="0" maxOccurs="1"/>
  <xsd:element name="SettlementMethodType"
type="bie:SettlementMethodCodeType_000145"
minOccurs="0" maxOccurs="1"/>
</xsd:choice>

See above

Remark:
The multiplicity for the property “ScheduledMeterReading_Date” is redefined into 1..1. The multiplicity for the property “MeterReadingFrequency_Duration” remains optional. As a consequence the choice becomes optional.

--constraint nr 11
inv: self.five.ScheduledMeterReading_Date->size()=1
inv: self.five.ScheduledMeterReading_Date->size()=1 implies
self.five.MeterReadingFrequency_Duration->size()=0
inv: self.five.MeterReadingFrequency_Duration->size()=1 implies
self.five.ScheduledMeterReading_Date->size()=0

From Measurement_MeteringPoint_Characteristics:
<xsd:choice
  minOccurs="0" />
  <xsd:element name="ScheduledMeterReadingDate"
type="bie:DateTimeType_000111" minOccurs="1"
maxOccurs="1"/>
  <xsd:element name="MeterReadingFrequencyDuration"
type="bie:DurationType_000151" minOccurs="0"
maxOccurs="1"/>
</xsd:choice>
Remark:
Also a combination of ">size()=.." and "isEmpty" or "notEmpty" leads to a choice when using the statement "implies".

See above
Remark:
The multiplicity for "Direction_Type" and for "StandardLoadProfile" are defined outside the "implies"-statement. Both become required. As a consequence the choice becomes required.
See above

Remark:
The property “Total” has in the ABIE multiplicity 1 and property “MeterTimeFrame_Type” has multiplicity 0..1. These multiplicities are not redefined in the “implies”-statement. Therefore “MeterTimeFrame” remains optional, also in the choice. And as a consequence, the choice itself becomes optional.

Remark:
The fixed value for “Total” and enumeration values for “MeterTimeFrame_Type” stem from constraint 20.

--constraint nr 14

inv: self.six.Annual_Period->forAll (Total->size()=1 implies MeterTimeFrame_Type->size()=0)
inv: self.six.Annual_Period->forAll (MeterTimeFrame_Type->size()=1 implies Total->size()=0)

```xml
<xs:complexType name="EstimatedMetrics">
  <xs:sequence>
    <xs:choice minOccurs="0">
      <xs:element name="Total"
        type="bie:IntegerQuantityType_000119"
        fixed="20"
        minOccurs="1" maxOccurs="1"/>
      <xs:element name="MeterTimeFrameType"
        minOccurs="0" maxOccurs="1">
        <xs:simpleContent>
          <xs:restriction base="bie:MeterTimeFrameCodeType_000126">
            <xs:enumeration value="E10"/>
            <xs:enumeration value="E11"/>
          </xs:restriction>
        </xs:simpleContent>
      </xs:element>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```
--constraint nr 15

self.Process-
>forall(Energy_BusinessProcess->size()=1)
xor
Energy_BusinessProcessRole-
>size()=1)

<xsd:complexType name="EnergyContext">
  <xsd:sequence>
    <xsd:element name="EnergyBusinessProcess" type="bie:BusinessReasonCodeType_000115" minOccurs="1" maxOccurs="1"/>
    <xsd:element name="EnergyBusinessProcessRole" type="bie:BusinessRoleCodeType_000113" minOccurs="1" maxOccurs="1"/>
    <xsd:element name="EnergyIndustryClassification" type="bie:SectorAreaIdentificationCodeType_000116" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
See above

Remark: the coded values for DocumentName are the result of constraint 19.

--constraint nr 16


```xml
<xsd:complexType name="EnergyDocument">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="Identification" type="bie:TransactionIdentifierType_000361" minOccurs="1" maxOccurs="1"/>
      <xsd:element name="DocumentType" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
          <xsd:simpleContent>
            <xsd:restriction base="bie:DocumentNameCodeType_000098">
              <xsd:enumeration value="C02"/>
              <xsd:enumeration value="E30"/>
            </xsd:restriction>
          </xsd:simpleContent>
          <xsd:complexType>
            <xsd:complexContent>
              <xsd:restriction base="bie:DocumentNameCodeType_000098">
                <xsd:enumeration value="C02"/>
                <xsd:enumeration value="E30"/>
              </xsd:restriction>
            </xsd:complexContent>
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
      <xsd:element name="Creation" type="bie:DateTimeType_000111" minOccurs="1" maxOccurs="1"/>
    </xsd:element>
    <xsd:element ref="bie:SenderEnergyParty" minOccurs="1" maxOccurs="1"/>
    <xsd:element ref="bie:RecipientEnergyParty"/>
  </xsd:sequence>
</xsd:complexType>
```

Generated by XMLSpy www.altova.com
**Remark:**

- "xor" creates choice with required elements
- "implies" allows for the creation of choices with optional elements
- if a choice contains one or more optional elements, the choice itself will become optional

```
<required>
--constraint nr 17

self.seven.BalanceSupplier_Involved->size()=1 xor
self.seven.MeteredDataCollector_Involved->size()=1 xor
self.seven.MeteredDataResponsible_Involved->size()=1 xor
self.seven.GridAccessProvider_Involved->size()=1 xor
self.seven.Consumer_Involved->size()=1

inv: self.seven.Identification->size()=1 implies self.seven.Occurrence->size()=0 and
self.seven.OriginalBusinessDocument_Reference->size()=0 and
self.seven.BusinessProcess_Reference->size()=0 and
self.seven.SnapShot_Occurrence->size()=0

inv: self.seven.Occurrence->size()=1 implies self.seven.Identification->size()=0 and
self.seven.OriginalBusinessDocument_Reference->size()=0 and
self.seven.BusinessProcess_Reference->size()=0 and

Required choice:
<xsd:choice>
</xsd:choice>
```
self.seven.SnapShot_Occurrence->size()==0
inv: self.seven.OriginalBusinessDocument_Reference->size()==1 implies
self.seven.Identification->size()==0 and
self.seven.Occurrence->size()==0 and
self.seven.BusinessProcess_Reference->size()==0 and
self.seven.SnapShot_Occurrence->size()==0
inv: self.seven.BusinessProcess_Reference->size()==1 implies self.seven.Identification->size()==0 and
self.seven.Occurrence->size()==0 and
self.seven.OriginalBusinessDocument_Reference->size()==0 and
self.seven.SnapShot_Occurrence->size()==0
inv: self.seven.OriginalBusinessDocument_Reference->size()==1 implies self.seven.Identification->size()==0 and
self.seven.Occurrence->size()==0 and
self.seven.BusinessProcess_Reference->size()==0 and
self.seven.SnapShot_Occurrence->size()==0

Optional choice:
<xsd:choice minOccurs="0">
  <xsd:element name="Identification" type="bie:TransactionIdentifierType_000361" minOccurs="0" maxOccurs="1"/>
  <xsd:element name="Occurrence" type="bie:DateTimeType_000111" minOccurs="0" maxOccurs="1"/>
  <xsd:element name="OriginalBusinessDocumentReference" type="bie:TransactionIdentifierType_000361" minOccurs="0" maxOccurs="1"/>
  <xsd:element name="BusinessProcessReference" type="bie:TransactionIdentifierType_000361" minOccurs="0" maxOccurs="1"/>
  <xsd:element name="SnapShotOccurrence" type="bie:DateTimeType_000111" minOccurs="0" maxOccurs="1"/>
</xsd:choice>
See above,

Purpose of this requirement:
A constraint without any ebIX stereotype (only with the UML <<invariant>> constraint) has to be ignored for transformation purposes.
This requirement is defined because it must be possible to specify a usage rule that has no effect on the XML schema representation, since the UMM model shall be technology/syntax independent.

---constraint nr 18
--this constraint is to be ignored for transformation and is only a representation of specific business requirements
self.seven.MP_Address->size() = 0

From MasterDataMP_Event:
<xsd:element ref="bie:MPAddressMeteringPointAddress"
minOccurs="0" maxOccurs="1"/>

---constraint nr 19-A

self.Header.Document_Type.content=DocumentNameCode::E30 or
self.Header.Document_Type.content=DocumentNameCode::C02
inv:
self.Header.Document_Type.listIdentifier->size() <= 1
inv:
self.Header.Document_Type.listIdentifier = CodeListIdentificationCode::EBO
inv:
self.Header.Document_Type.listAgencyIdentifier=CodeListResponsibleAgencyCode::260

---constraint nr 19-B
--this constraint is to be ignored for transformation and is only a representation of specific business

This constraint replaces constraint number 8

---constraint nr 20

```ocla
self.four.Resolution_Duration.content="PT15M" or self.four.Resolution_Duration.content="PT30M"
```

Remark:
The constraint 20 defines an “on-the-fly”
enumeration (i.e. a set of values not being literals in an existing enumeration). In this case the values are the combination of pattern and value (such as PT15M) used to specify the “Resolution”.

An “on-the-fly” enumeration is by ebIX decision only allowed to be specified for “Resolution” in the <<ABIE>> “TimeSeries_Period”.

Remark:
The first part of constraint 20 tries to define an “on-the-fly” enumeration (values 10 or 20 for “Total”). However, this part of the constraint leads to one fixed value only, since an “on-the-fly” enumeration is by ebIX decision only allowed to be specified for “Resolution” in the <<ABIE>> “TimeSeries_Period”. The fixed value in the XML schema is 20, since this is the last value specified and therefore overriding the preceding value.

The second part of the constraint specifies two possible values for “MeterTimeFrame_Type”. This is allowed since it uses an already existing enumeration in the UML model.

Remark:
The choice in XML is the result of constraint 14.
Facets added to ABIE / MBIE by means of OCL-statements

**Remark:**
If the tag type is integer, a “=” and if the tag type is string a “=“ or “::” may be used.

ebIX ETC has decided to limit the options (“=” or “::”) to only use “=” for the facets and to continue using “::” for the enumeration literals used.

Allowed facets are: length, maxLength, minLength, maxInclusive, maxExclusive, minInclusive, minExclusive, fractionalDigits, totalDigits, pattern

**Remark:** The part of the XML-text representing the actual facet is highlighted yellow

---

<table>
<thead>
<tr>
<th>Customer_Party</th>
<th>-- constraint nr 22</th>
</tr>
</thead>
<tbody>
<tr>
<td>self.Payload.Identification.content.length =13</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain_Location</th>
<th>-- constraint nr 23</th>
</tr>
</thead>
<tbody>
<tr>
<td>self.one.Identification.content.maxLength =18</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TimeSeries</th>
<th>--constraint nr 24</th>
</tr>
</thead>
<tbody>
<tr>
<td>self.two.Identification.content.minLength =10</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Observation</th>
<th>--constraint nr 25</th>
</tr>
</thead>
<tbody>
<tr>
<td>self.three.Energy_Quantity.content.maxInclusive=10</td>
<td></td>
</tr>
</tbody>
</table>

Because of constraint 31

---

\[ \text{[Because of constraint 31]} \]
---constraint nr 27
self.three.Energy_Quantity.content.minInclusive=4
---constraint nr 28
self.three.Generated_Quantity.content.minInclusive=6
---constraint nr 29
self.three.Energy_Price.content.fractionalDigits =3
---constraint nr 30
self.three.Energy_Price.content.totalDigits =6

**Customer_Party**
--- constraint nr 31
self.Payload.Name.content.pattern=test[0-9]{4}--end

```xml
<xs:complexType name="DomainLocation">
  <xs:sequence>
    <xs:element name="Identification" minOccurs="1" maxOccurs="1">
      <xs:simpleType>
        <xs:restriction base="bie:DomainIdentifierType_000122">
          <xs:maxLength value="18"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EnergyTimeSeries">
  <xs:sequence>
    <xs:element name="Identification" minOccurs="0" maxOccurs="1">
      <xs:simpleType>
        <xs:restriction base="bie:TransactionIdentifierType_000361">
          <xs:minLength value="10"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EnergyObservation">
  ...
</xs:complexType>
```
<xsd:sequence>
  <xsd:choice minOccurs="0">
    <xsd:element name="EnergyQuantity" minOccurs="0" maxOccurs="1">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:restriction base="bie:IntegerQuantityType_000119">
            <xsd:maxInclusive value="10"/>
            <xsd:maxExclusive value="12"/>
            <xsd:minInclusive value="4"/>
            <xsd:minExclusive value="6"/>
          </xsd:restriction>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="GeneratedQuantity" minOccurs="0" maxOccurs="1">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:restriction base="bie:IntegerQuantityType_000119">
            <xsd:minExclusive value="6"/>
          </xsd:restriction>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
  </xsd:choice>
</xsd:sequence>
Import UN/CEFACT XML schema for date/time formats (patterns)

For each data/time type for which we define a BDT we specify the mapping to one particular option in XML

Handling of the primitive data type Binary.

UN/CEFACT XBT file containing the additionally specified patterns for date/time is imported in the BIE-files (generic ABIE and customized MBIE)
**Remark:** primitive data type Binary in UMM has to be mapped to xsd: base64Binary in XML.

**Additional note:** ebIX ETC has decided that mapping of other primitive datatypes in UMM such as Double, Float, NormalizedString and Token is not needed.

<table>
<thead>
<tr>
<th>Issue</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Do we need double (precision)?</td>
<td>• No, we use decimal only</td>
</tr>
<tr>
<td>• Do we need Float?</td>
<td>• No, we use decimal only</td>
</tr>
<tr>
<td>• NormalizedString?</td>
<td>• No, we replace this with datatype for codelist</td>
</tr>
<tr>
<td>• Do we need Token?</td>
<td>• No, we replace this with datatype for codelist</td>
</tr>
</tbody>
</table>