ETIM IFX 1.0 Classification Release Format specification

English version - revision 2014-01-24

Contents

DOCUMENT REVISIONS

Revision	Authors	Description
2014-01-24	Marc Habets, ETIM Netherlands Erwin van der Waal, ETIM Netherlands Arian Haeser, ETIM Netherlands	Initial version

1   INTRODUCTION

Starting from ETIM 6.0, ETIM IXF will be considered the international standard ETIM release format. The ETIM IXF format is multilingual, so it can contain multiple language versions of the ETIM model in one file.

1.1   USED SYMBOLS

Optional element with sub elements

Mandatory element (single)

Optional element (multiple)

Mandatory element (multiple)
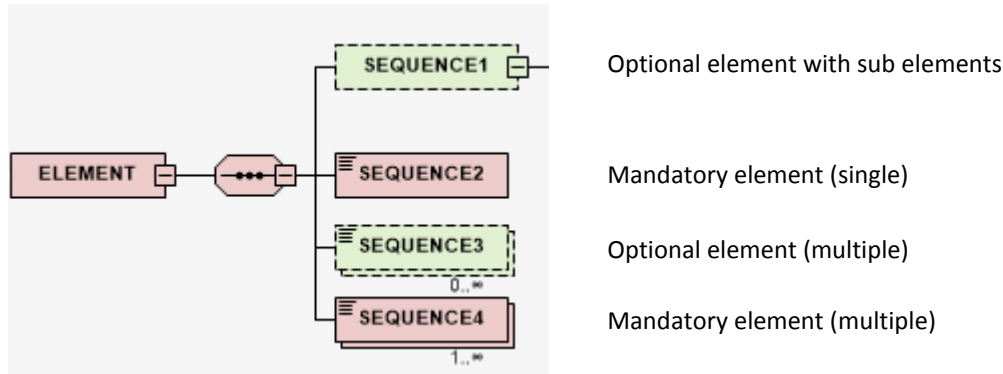
1.2   CONVENTIONS

The schema is build according the "Venetian Blind" design pattern.

In the schema the following conventions are maintained:
- Elements are ordered by its usage except the simple types
- Element names are written in UCC (upper camel case) e.g. "UnitListType"
- Attribute names are written in LCC (lower camel case) e.g. "schemaVersion"

1.3   VERSIONING

Versioning of schema and instance documents are managed by the fixed schemaVersion attribute on the root element. This attribute can eventually be used to determine which schema must be used to import the file.

The XML namespace contains the major part of the version and therefore separates major versions completely.
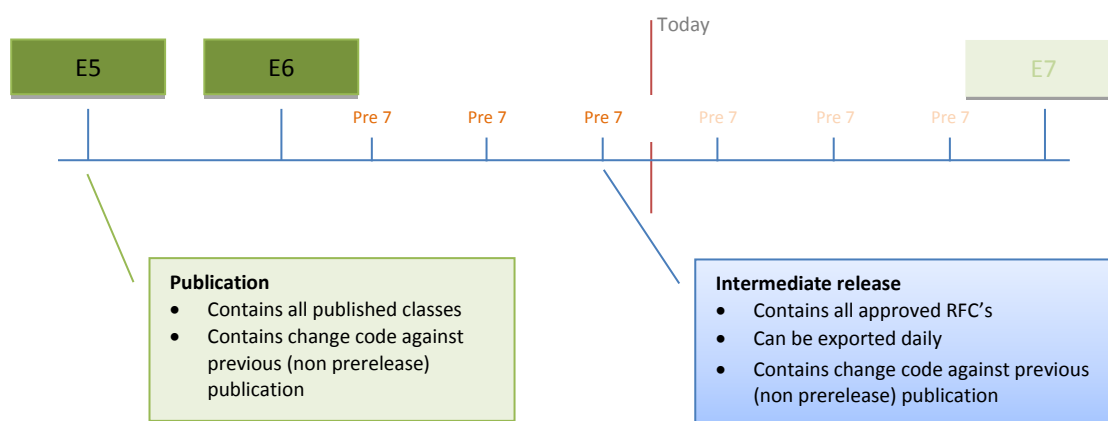
2   CLASSIFICATION PUBLICATIONS

2.1   PUBLICATION

An (official) ETIM publication is the release of a set of unique classes (no multiple versions of one class) fixed at a certain moment in time. A publication has a name and a date. The publication contains change codes reflecting the changes compared to the previous official ETIM publication. A publication contains only classes with status "Published" except when it is an intermediate release as described in chapter 2.2.

2.2   INTERMEDIATE (DYNAMIC) RELEASE

An intermediate release can provide a flexible solution that makes it possible to anticipate on the next official publication. This means using new classes, features or values already for internal applications. This means being able to communicate this information bilaterally between two trading partners, if both support this information. This means having an option of spreading the internal work load involved in adapting the classified product information to a new ETIM release. The IXF format supports the ability to export an intermediate release. An intermediate release is identified in the ETIM IXF format by the header element "Prerelease" and always gives the changes related to the current official ETIM release using the change codes as defined in chapter 5.12.

2.3   TIMELINE PUBLICATION & PRERELEASES

Today

| E5 | E6 | | | | | | | E7 |

Pre 7   Pre 7   Pre 7   Pre 7   Pre 7   Pre 7

Publication
- Contains all published classes
- Contains change code against previous (non prerelease) publication

Intermediate release
- Contains all approved RFC's
- Can be exported daily
- Contains change code against previous (non prerelease) publication

3 ELEMENT STRUCTURE

In this chapter the most important elements of the IXF format are described in order of appearance.

3.1 IXF (ROOT ELEMENT)

The format is built up with a header section, master tables with all the used groups, features, values, units and the class section defining classes and relating them to groups, features, values and units.
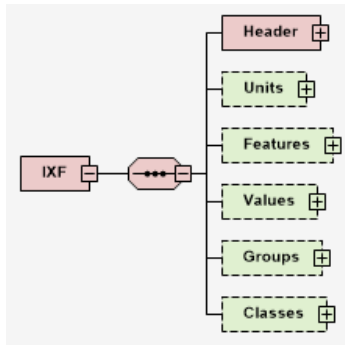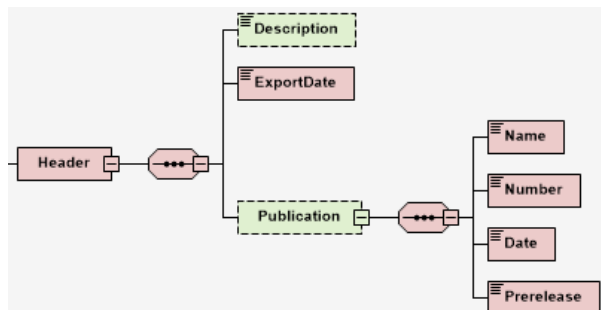

Element	Format		Description
Header	Element *HeaderType*	1..1	Group of header fields. Contains identifying information etc.
Units	Element *UnitListType*	0..1	Master-table with definition of all used units in this export
Features	Element *FeatureListType*	0..1	Master-table with definition of all used features in this export
Values	Element *ValueListType*	0..1	Master-table with definition of all used values in this export
Groups	Element *GroupListType*	0..1	Master-table with definition of all used groups in this export
Classes	Element *ClassListType*	0..1	The classes structure

Attribute	Format		Description
schemaVersion	Decimal	1..1	The schema version is identified with this attribute, you can use this attribute to select the correct schema to import the data. (The namespace will only give the major part of the version number) **Fixed value: 1.0**

3.2 HEADER

In the header section general information about the release is given. Since the ETIM IXF format can and will also be used for exports from the database that are not official ETIM releases, the header section is divided into two elements that at first impression might seem to overlap each other.

Element	Format		Description
Description	String 1..80 *ShortDescriptionType*	0..1	Description of this export
ExportDate	DateTime ISO 8601 UTC date/time combined	1..1	Date and time **Example:** 2013-03-22T16:25:38.6166178Z
Publication	Element *PublicationType*	0..1	When this file is an official release this element contains information such as name, number etc. When this file is a simple export this element is omitted.

There are no attributes defined in the header element

3.3 PUBLICATION (PUBLICATIONTYPE)

In case of an export that is not an official release the "Publication" element can be omitted. The element "Prerelease", a Boolean that indicates if this release is a pre-release or not, is related to the possibility of using ETIM IXF for a more dynamic release of ETIM.

Element	Format		Description
Name	String 1..80 *ShortDescriptionType*	1..1	The name of the publication. **Example:** "ETIM 7.0"
Number	Integer	1..1	The (sequential) publication number of this publication. On a prerelease it is the number of the next coming publication **Example:** 7
Date	Date	1..1	The official release date of the publication **Example:** 2013-03-26T12:55:24.1163138Z
Prerelease	Boolean	1..1	Identifies if this release is an intermediate release as described in chapter 2.2. **Example:** true

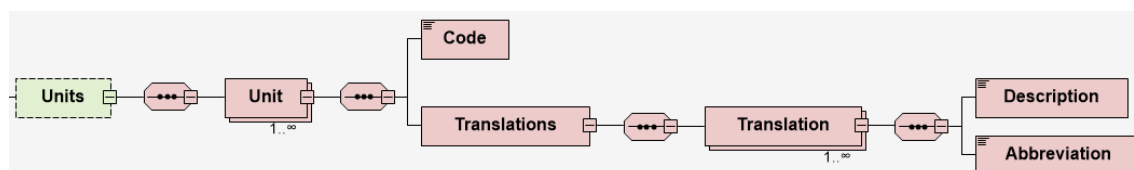*There are no attributes defined in the publication element*

Example*:*

```xml
<Header>
  <Description>Intermediate release</Description>
  <ExportDate>2013-03-22T21:25:38.6156154Z</ExportDate>
  <Publication>
    <Name>ETIM 7.0</Name>
    <Number>7</Number>
    <Date>2013-03-26T12:55:24.1163138Z</Date>
    <Prerelease>true</Prerelease>
  </Publication>
</Header>
```

3.4 UNITS (UNITLISTTYPE)

The master table for units contains all the units used in this release or export, with its official code (ID) and the respective translations of description and abbreviation, as illustrated in the example below.

Element	Format		Description
Unit	*Element* *UnitType*	1..∞	Each unit is defined by type UnitType.

There are no attributes defined in the units element

3.4.1 Unit (UnitType)

Element	Format		Description
Code	String 8..8 *UnitCodeType*	1..1	Identifying Code (ID) for the UNIT. Format: "EU" + 6 digits **Example:** "EU570036"
Translations	Element *UnitTranslationListType*	1..1	Translations contains translation elements (UnitTranslationListType) for different languages

There are no attributes defined in the unit element

3.4.2 Translations (UnitTranslationListType)

Element	Format		Description
Translation	*Element* *UnitTranslationType*	1..∞	Each translation is defined by type UnitTranslationType.

There are no attributes defined in the units element

3.4.3 Translation (UnitTranslationType)

Element	Format		Description
Description	String 1..80 *ShortDescriptionType*	1..1	Each unit translation contains the description for the language specified by the *language* attribute.
Abbreviation	String 1..15 *UnitAbbreviationType*	1..1	Each unit translation contains an abbreviation (e.g. UOM) for the language specified by the *language* attribute.

Attribute	Format		Description
language	String 1..8 *LanguageCode*	1..1	The (translation) language is identified with this attribute. The language pattern used is based on the ISO 639-1 code **Example:** "nl-NL"

Example

```xml
<Unit>
  <Code>EU570036</Code>
  <Translations>
    <Translation language="en-GB">
      <Description>Meter</Description>
      <Abbreviation>m</Abbreviation>
    </Translation>
    <Translation language="nl-NL">
      <Description>Meter</Description>
      <Abbreviation>m</Abbreviation>
    </Translation>
  </Translations>
</Unit>
```
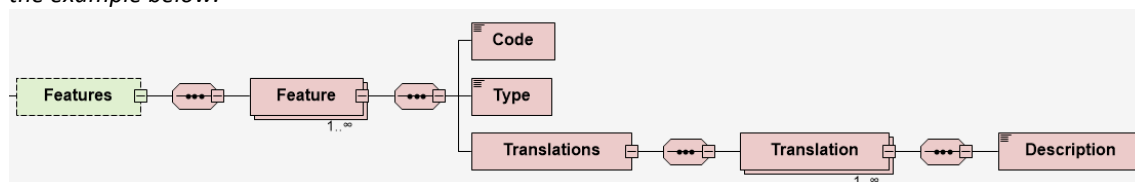
3.5 FEATURES (FEATURESLISTTYPE)

The master table for features contains all the features used in this release or export, with its official code (ID), the type of feature (A, N, R or L) and the respective translations of the description, as illustrated in the example below.

Element	Format		Description
Feature	Element *FeatureType*	1..∞	Each feature is defined by type FeatureType.

There are no attributes defined in the features element

Example
```
<Feature>
   <Code>EF000007</Code>
   <Type>Alphanumeric</Type>
   <Translations>
     <Translation language="en-GB">
        <Description>Colour</Description>
     </Translation>
     <Translation language="nl-NL">
        <Description>Kleur</Description>
     </Translation>
   </Translations>
</Feature>
```

3.5.1 Feature (FeatureType)

Element	Format		Description
Code	String 8..8 *UnitCodeType*	1..1	Identifying Code (ID) for the feature. Format: "EF" + 6 digits **Example:** "EF000007"
Type	Enumeration *FeatureTypeType*	1..1	A feature type can be specified as being *Alphanumeric*, *Numeric*, *Range* or *Logical*
Translations	Element *ShortTranslationListType*	1..1	Translations contains translation elements for different languages

There are no attributes defined in the feature element

3.6 VALUES (VALUELISTTYPE)

The master table for values contains all the values used in this release or export, with its official code (ID) and the respective translations of the description, as illustrated in the example below.
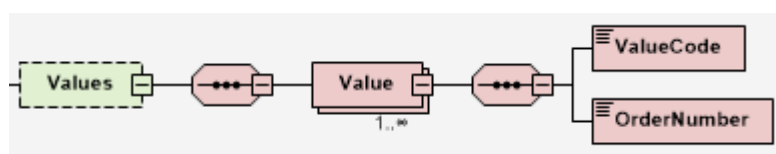
Element	Format		Description
Value	Element *ValueType*	0..1	List of possible values. Only specified for use with alphanumeric features.

There are no attributes defined in the values element

Example
```
<Value>
   <Code>EV000134</Code>
   <Translations>
     <Translation language="en-GB">
        <Description>Built-in</Description>
     </Translation>
     <Translation language="nl-NL">
        <Description>Inbouw</Description>
     </Translation>
   </Translations>
</Value>
```

3.6.1   Value (ValueType)

Element	Format		Description
Code	String 8..8 *ValueCodeType*	1..1	Identifying code of a value. Format: "EV" + 6 digits. **Example:** "EV000134"
Translations	Element *ShortTranslationListType*	1..1	Translations contains translation elements for different languages

There are no attributes defined in the value element.

3.7   GROUPS (GROUPSLISTTYPE)

The master table for groups contains all the groups used in this release or export, with its official code (ID) and the respective translations of the description, as illustrated in the example below.
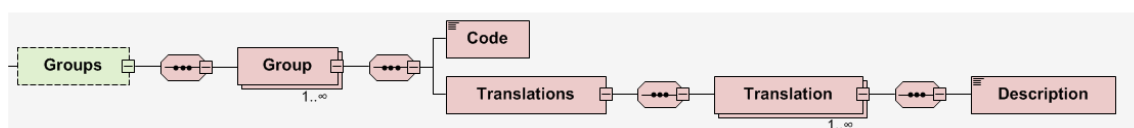
Element	Format		Description
Group	Element *GroupType*	1.. ∞	Contains Group elements (GroupType) for each group referenced.

There are no attributes defined in the groups element

Example

```
<Group>
  <Code>EG000001</Code>
  <Translations>
    <Translation language="en-GB">
      <Description>Cables and wires</Description>
    </Translation>
    <Translation language="nl-NL">
      <Description>Kabels / Leidingen</Description>
    </Translation>
  </Translations>
</Group>
```

3.7.1   Group (GroupType)

Each group is uniquely identified by a Code and contains one or more translations.

| Element	Format		Description
Code	String 8..8 *GroupCodeType*	1..1	Identifying code of the group. Refers to the base table of Values in the XML file. Format: "EG" + 6 digits. **Example:** "EG000001"
Translations	Element *ShortTranslationListType*	1..1	Translations contains translation elements for different languages

There are no attributes defined in the group element

3.8 CLASSES (CLASSLISTTYPE)

The "classes" section defines all the classes included in the release or export.

The following elements are defined per class:
- The class code (ID)
- The class version. Please note that this class version is no longer related to the release version, as it was in the past! A new class will have version 1 and the class version only changes if the class has been changed.
- The translation part gives the translations of the official class name, but also the language dependent synonyms per language version.
- The relations between the class and its allocated features, values and/or units including the display order.
- The status of a class, which will always be "Published" if it concerns an official release, in case of a pre-release the value "ReadyForPublication" can occur for classes that are changed after the latest official release.
- Finally the class is attributed to a group
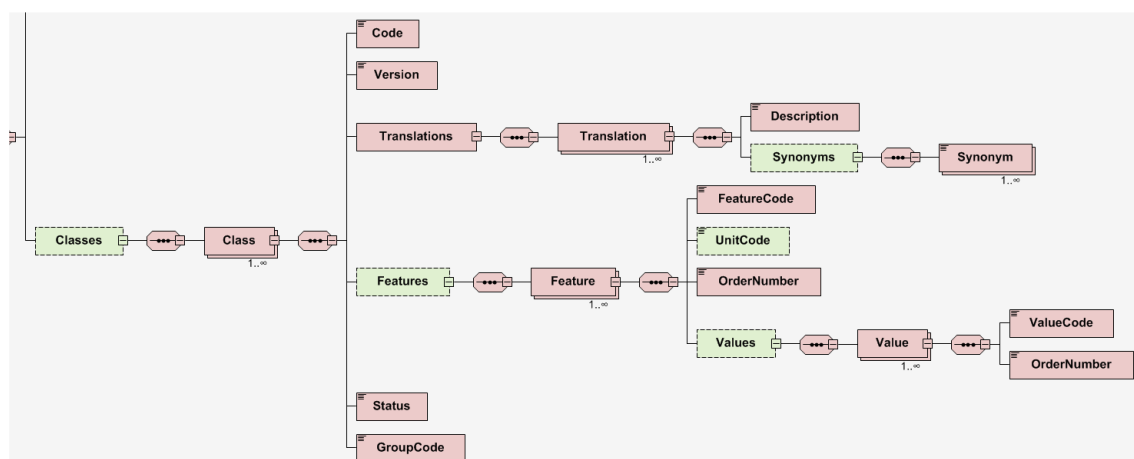
What the above diagram does not indicate is that the ETIM IXF format also contains change codes on CLASS level, on CLASSFEATURE level and on CLASSFEATUREVALUE level. These change codes indicate if an element is new, changed, unchanged or deleted.

Element	Format		Description
Class	Element *ClassType*	1.. ∞	Each class within classes is specified by ClassType

There are no attributes defined in the classes element

4 CLASS (CLASSTYPE)
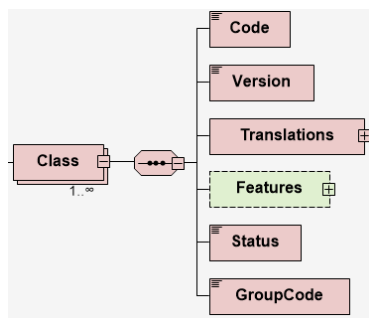
Element	Format		Description
Code	String 8..8 *ClassCodeType*	1..1	Each class is identified by a class Code (ID). Format: "EC" + 6 digits. **Example:** "EC000001"
Version	Integer	1..1	The class version number (auto incremented number for each class change).
Translations	Element *ClassTranslationListType*	1..1	Each class contains translations.
Features	Element *ClassFeatureListType*	0..1	Each class contains class features.
Status	Enumeration *ClassStatusType*	1..1	A class can have the status *"Defined"*, *"UnderConstruction"*, *"ReadyForPublication"* or *"Published"*
GroupCode	String 8..8 *GroupCodeType*	1..1	This specifies to which group this class is assigned to. Format: "EG" + 6 digits **Example:** "EG000001"

Attribute	Format		Description
changeCode	Enumeration *ChangeCodeType*	1..1	The change code indicates if an element is *"New"*, *"Changed"*, *"Unchanged"* or *"Deleted"*.

These change codes are automatically generated at the export from the ETIM database, related to the (latest) release version.

Example

```
    </Class>
    <Class changeCode="Changed">
      <Code>EC000001</Code>
      <Version>1</Version>
      <Translations>
        <Translation language="en-GB">
          <Description> Busbar terminal</Description>
          <Synonyms>
            <Synonym> Busbar terminal</Synonym>
          </Synonyms>
        </Translation>
        <Translation language="nl-NL">
          <Description> Aansluitklem stroomrail</Description>
          <Synonyms>
            <Synonym> Aansluitklem stroomrail</Synonym>
            <Synonym>Kabelklem</Synonym>
          </Synonyms>
        </Translation>
      </Translations>
      <Features>
        <Feature changeCode="New">
          <FeatureCode>EF000010</FeatureCode>
          <OrderNumber>1</OrderNumber>
          <Values>
            <Value changeCode="New">
              <ValueCode>EV003652</ValueCode>
              <OrderNumber>1</OrderNumber>
            </Value>
            <Value changeCode="New">
              <ValueCode>EV008730</ValueCode>
              <OrderNumber>2</OrderNumber>
            </Value>
          ………
```
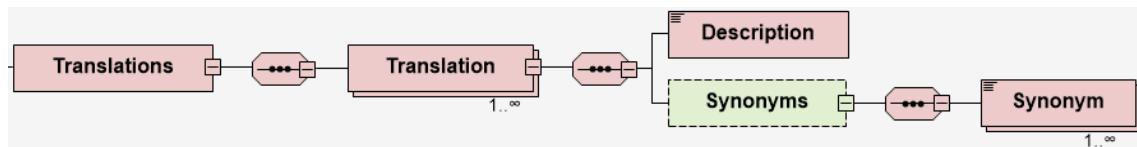
4.1 TRANSLATIONS (CLASSTRANSLATIONLISTTYPE)

The translation part gives the translations of the official class name, but also the language dependent synonyms per language version.

Element	Format		Description
Translation	Element *ClassTranslationType*	1.. ∞	Contains a list of translation elements for this class

There are no attributes defined in the translations element

4.1.1 Translation (ClassTranslationType)

Element	Format		Description
Description	String 1..80 *ShortDescriptionType*	1..1	Each class translation contains the description for the language specified by the *language* attribute.
Synonyms	Element *ClassSynonymListType*	0..1	For each language a list of synonyms may be specified.

Attribute	Format		Description
language	String 1..8 *LanguageCode*	1..1	The language is identified with this attribute. The language pattern used is based on the ISO 639-1 code **Example:** "nl-NL"

4.1.2 Synonyms (ClassSynonymListType)

Element	Format		Description
Synonym	Element *ShortDescriptionType*	1.. ∞	Synonym contains class synonyms for the language specified by the language attribute.

There are no attributes defined in the synonyms element

Example

```xml
<Translations>
  <Translation language="en-GB">
    <Description>Busbar terminal</Description>
    <Synonyms>
      <Synonym>Busbar terminal</Synonym>
    </Synonyms>
  </Translation>
  <Translation language="nl-NL">
    <Description>Aansluitklem stroomrail</Description>
    <Synonyms>
      <Synonym>Aansluitklem stroomrail</Synonym>
      <Synonym>Kabelklem</Synonym>
    </Synonyms>
  </Translation>
</Translations>
```
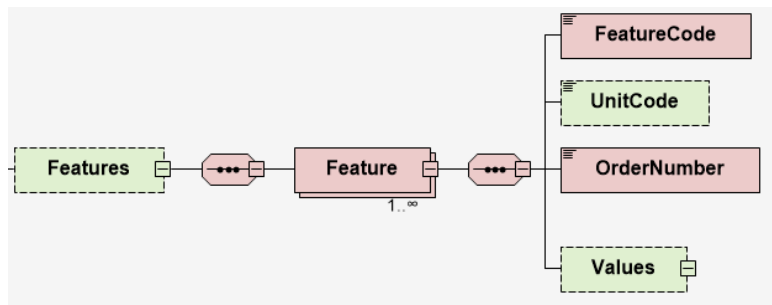
4.2 FEATURES (CLASSFEATURELISTTYPE)

This element lists all features that belongs to this class

Element	Format		Description
Feature	Element *ClassFeatureType*	1.. ∞	List of features that belongs to this class

There are no attributes defined in the features element

4.2.1 Feature (ClassFeatureType)

Element	Format		Description
FeatureCode	String 8..8 *FeatureCodeType*	1..1	Identifying code of a feature. Refers to the base table of Features in the XML file. Format: "EF" + 6 digits. **Example:** "EF000134"
UnitCode	Element *UnitCodeType*	0..1	For each feature of FeatureType *Numeric* or *Range a unit code* may be specified.
OrderNumber	Integer	1..1	Sort order of the features. **Example:** 1
Values	Element *ClassFeatureValueListType*	0..1	This element contains the value list for this feature. Only applicable for features of FeatureType *Alphanumeric*

Attribute	Format		Description
changeCode	Enumeration *ChangeCodeType*	1..1	The change code indicates if an element is "*New*", "*Changed*", "*Unchanged*" or "*Deleted*".

Example

```xml
<Feature changeCode="Unchanged">
  <FeatureCode>EF005644</FeatureCode>
  <OrderNumber>7</OrderNumber>
  <Values>
   …
  </Values>
</Feature>
<Feature changeCode="Unchanged">
  <FeatureCode>EF000010</FeatureCode>
  <OrderNumber>8</OrderNumber>
   <Values>
   …
  </Values>
</Feature>
<Feature changeCode="Unchanged">
  <FeatureCode>EF005448</FeatureCode>
  <UnitCode>EU570508</UnitCode>
  <OrderNumber>9</OrderNumber>
</Feature>
<Feature changeCode="Unchanged">
  <FeatureCode>EF005449</FeatureCode>
  <OrderNumber>10</OrderNumber>
</Feature>
```
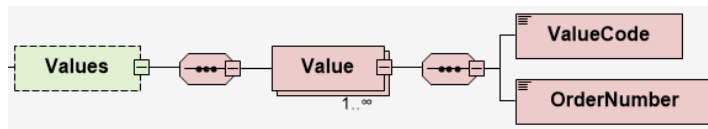
4.3 VALUES (CLASSFEATUREVALUELISTTYPE)

Element	Format		Description
Value	Element *ClassFeatureValueType*	1.. ∞	List of value elements

There are no attributes defined in the values element

4.3.1 Value (ClassFeatureValueType)

Element	Format		Description
ValueCode	String 8..8 *ValueCodeType*	1..1	Identifying code of a value. Refers to the base table of Values in the XML file. Format: "EV" + 6 digits. **Example:** "EV000134"
OrderNumber	Integer	1..1	Sort order of the values. For values with equal order number, sorting should be done alphabetically on the description for the relevant language. **Example:** 2

Attribute	Format		Description
changeCode	Enumeration *ChangeCodeType*	1..1	The change code indicates if an element is "*New*", "*Changed*", "*Unchanged*" or "*Deleted*".

Example

```xml
<Features>
  <Feature changeCode="New">
    <FeatureCode>EF000010</FeatureCode>
    <OrderNumber>1</OrderNumber>
    <Values>
      <Value changeCode="New">
        <ValueCode>EV003652</ValueCode>
        <OrderNumber>1</OrderNumber>
      </Value>
      <Value changeCode="New">
        <ValueCode>EV008730</ValueCode>
        <OrderNumber>2</OrderNumber>
      </Value>
```
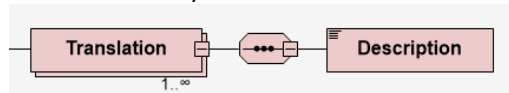
5 GENERIC ELEMENTS

5.1 TRANSLATIONS (SHORTTRANSLATIONLISTTYPE)

This element only contains a list of 1..∞ translations of the type ShortTranslationType.

Element	Format		Description
Translation	Element *ShortTranslationType*	1.. ∞	Each translation contains a translation of the type

There are no attributes defined in the translations element

5.2 TRANSLATION (SHORTTRANSLATIONTYPE)

Element	Format		Description
Description	String 1..80 *ShortDescriptionType*	1..1	Each translation contains a translation of the type *ShortDescriptionType* for the language specified by the *language* attribute.

Attribute	Format		Description
language	String 1..8 *LanguageCode*	1..1	The language is identified with this attribute. The language pattern used is based on the ISO 639-1 code **Example:** "nl-NL"

6 GENERIC TYPES

Below you will find a table of generic types:

Element	Format / restriction	Description
ClassCodeType	Token (string), fixed length of 8 **Pattern**: EC[0-9]{6}	**Example:** "EC000001"
ClassVersionType	Unsigned Integer	**Example:** 1
FeatureCodeType	Token (string), fixed length of 8 **Pattern**: EF[0-9]{6}	**Example:** "EF000134"
ValueCodeType	Token (string), fixed length of 8 **Pattern**: EV[0-9]{6}	**Example:** "EV000134"
UnitCodeType	Token (string), fixed length of 8 **Pattern**: EU[0-9]{6}	**Example:** "EU570036"
GroupCodeType	Token (string), fixed length of 8 **Pattern**: EG[0-9]{6}	**Example:** "EG000001"
ShortDescriptionType	String 1..80	The ShortDescriptionType is a simple type string having a length of minimal 1 and maximal 80 characters.
UnitAbbreviationType	String 1..15	
LanguageCode	Token (string), max. length of 8 **Pattern**: [a-zA-Z]{1,8}(-[a-zA-Z0-9]{1,8})*	Pattern specifies the content of section 2.12 of XML 1.0e2 and RFC 3066 (Revised version of RFC 1766). See also: http://www.w3.org/TR/xmlschema-2/#language Source: http://www.ietf.org/rfc/rfc3066.txt
ChangeCodeType	Token (string) **Enumeration of**: • New, • Changed, • Unchanged, • Deleted	
FeatureTypeType	Token (string) **Enumeration of**: • Alphanumeric, • Numeric, • Range, • Logical	
ClassStatusType	Token (string) **Enumeration of**: • Defined, • UnderConstruction, • ReadyForPublication, • Published	