CEN/TC XBRL

Date: 2013-12

TC XBRL WI XBRL001

CEN/TC XBRL

Secretariat: NEN

Improving transparency in financial and business reporting — Harmonisation topics — Part 3: European XBRL Taxonomy Architecture

Einführendes Element — Haupt-Element — Teil 3: Ergänzendes Element

Élément introductif — Élément central — Partie 3 : Élément complémentaire

ICS:

Descriptors:

Document type: CWA
Document subtype:
Document stage: Formal Vote
Document language: E

L:\2 - CEN CENELEC\CEN WS\WS XBRL\03 Projects\3 Approval\2 Final drafts\CWA_XBRL_WI001-3 (E).doc STD Version 2.5a

Contents

Foreword

This document has been prepared by CEN/WS XBRL, the secretariat of which is held by NEN.

CWA XBRL 001 consists of the following parts, under the general title *Improving transparency in financial and business reporting — Harmonisation topics*:

- *Part 1: European data point methodology for supervisory reporting.*
- *Part 2: Guidelines for data point modelling*
- *Part 3: European XBRL Taxonomy Architecture*
- *Part 4: European Filing Rules*
- *Part 5: Mapping between DPM and MDM*

This CWA is one of a series of related deliverables. The other deliverables are:

CWA XBRL 002 *Improving transparency in financial and business reporting — Metadata container*

CWA XBRL 003-1 *Improving transparency in financial and business reporting — Standard regulatory roll-out package for better adoption — Part 1: XBRL Supervisory Roll-out Guide*

CWA XBRL 003-2 *Improving transparency in financial and business reporting — Standard regulatory roll-out package for better adoption — Part 2: XBRL Handbook for Declarers*

0 Introduction

0.1 General

The purpose of this document is to present and explain the European XBRL Taxonomy Architecture (EXTA) defined by European supervisory authorities. In particular, it explains the scope (coverage of information requirements), the modularization of files, the manner of defining concepts and relations, and other important design aspects [Bund13, p.3]. It is fully compatible with the Data Point Methodology approach. As DPMs are semantic models being created by supervisory experts, they are not formalized from a technical point of view. XBRL, as a formal language, can fill this gap. XBRL, as a data format, is standardized and can therefore be used to enable automated processes. XBRL taxonomies are metadata specifications that provide a formal description of the data requirements to be used as data format in the European reporting process. The document also intends to provide a description for IT experts about the linkage between a DPM, as a source, and the XBRL taxonomy, as a target, of a transformation process.

These pages are an introduction to the guidelines of the EXTA.

- European: This project is funded by the EU commission to support the standardization process for supervisory reporting in Europe, but there is no restriction in applying it anywhere else;
- XBRL: This format has been chosen by the supervisory authorities of the European Banking Authority (EBA) and the European Insurance and Occupational Pensions Authority (EIOPA) for the electronic data exchange between the national competent authorities (NCA) and the European supervisory authorities (ESA);
- Taxonomy Architecture: This architecture has been created to limit the wide variety of options to define an XBRL taxonomy because different European XBRL taxonomies for similar purposes may cause incompatibility and would lead to increased implementation costs for all adopters in the EU market.

0.2 Objective

The objective of the EXTA is to define a set of architecture guidelines that transform a European DPM without a loss in quality of the XBRL format. The taxonomy architecture provides a set of rules for this transformation in order to enable the creation of consistent and predictable XBRL metadata definitions in an automated process. The EXTA supports a modular structure to enable the extensibility of these taxonomies, as well as to facilitate their maintenance. As the EXTA is a formal representation of a DPM, it contains several structural concepts which have no correspondence to other known XBRL architectures.

0.3 Target audience

The EXTA is targeted at taxonomy authors: Initially, for organizations like EBA, EIOPA, European Securities and Markets Authority (ESMA), European Central Bank (ECB) etc. As a spin-off of these taxonomies, local (national) initiatives will emerge, hosted by National Supervisory Agencies (NSA's). To accommodate the demands of local legislation, the European taxonomies may need to be extended for local requirements. The EXTA is also aimed at supporting these national extensions in accordance with the same guiding principles. The main advantage is to have a consistent framework of XBRL taxonomies, which enables a cost efficient implementation of software solutions.

Consistent taxonomies throughout Europe also create the opportunity for cross-EU harmonization of terminology and, at a later stage, consistent reported facts that are more easily analyzed because the underlying structure is the same and the terms used are complementary.

A consequence of a consistent taxonomy framework is that software developers can choose to support only the architectural guidelines of EXTA. Although this limits their software in supporting fully fledged XBRL taxonomies, it can improve implementation costs.

This document is aimed at the users of European supervisory taxonomies, in particular, editors of the taxonomy or producers of instance documents (by applying mappings to internal systems or assigning XBRL

tags with values), as well as developers of the IT solutions that facilitate reporting in the XBRL format or the analysis of XBRL data [Bund13, p.3].

0.4 Relationship to other work

The reader of this EXTA is expected to be familiar with the principles of data modeling, and have a thorough understanding of the XBRL family of specifications, in order to evaluate the impact of the rules set to the XBRL taxonomy that needs to be created.

1 Scope

The EXTA has been defined for the creation of XBRL Taxonomies in the context of European supervisory reporting. XBRL taxonomies following this architecture are published by a European supervisory authority to reflect the data requirements based on a DPM in a machine-readable form.

2 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

2.1
concept
XML element defined in the substitutionGroup xbrli:item or xbrli:tuple

2.2
data point model
DPM
structures of data describing the characteristics of the information exchanged, as it relates to supervisory reporting processes

2.3
dimension
data set of one characteristic area which is composed of individual and non-overlapping data elements

Note 1 to entry: In XBRL terms, a dimension is an abstract concept in the substitutionGroup xbrldt:dimensionItem.

2.4
domain
category with items that share a common semantic identity

Note 1 to entry: A domain is an abstract concept and is the parent in a domain-member relationship.

EXAMPLE: All individual countries can be members in a domain 'Countries'.

2.5
family
groups of Dimensions relevant for presentation or querying purposes.

Note 1 to entry: In XBRL terms, a family is an abstract concept in the substitutionGroup xbrli:item with the @type set to 'model:familyType'.

2.6
framework
business term common to a group of business users that refer to the reporting regulations for a domain specific scope of information, like COREP, FINREP or Solvency II

Note 1 to entry: Frameworks are public elements represented using XBRL abstract items with @type set to 'model:frameworkType' in schema 'fws.xsd'.

2.7
item
abstract XML element in the substitutionGroup xbrli:item that can be used to carry facts or to represent business measurements.

Note 1 to entry: The item is defined by XBRL International as a placeholder for creating concepts.

2.8
(domain) member
discrete, countable, and can be explicitly defined in a domain

Note 1 to entry: A member is an abstract concept in the substitutionGroup xbrli:item and is the child in a domain-member relationship.

2.9
metric
nature of the data with a fixed and unchangeable meaning

Note 1 to entry: Metrics are strongly related to the underlying data type. Mostly, they are numeric and quantitatively measurable, but they can also reflect boolean or date values. A metric also known in XBRL terms as “primary” is a non-abstract concept in the substitutionGroup xbrli:item that is defined based on the unique combination of a (XML) type and xbrli:periodType.

2.10
namespace
unique string in the form of a Uniform Resource Identifier (URI) that is the @targetNamespace content of (a set of) XML schema file(s)

2.11
owner
authorised organisation that defines concepts

Note 1 to entry: The owner's namespace is a URI used to establish the namespace of the concepts defined by that owner.

2.12
perspective
auxiliary concepts meant to group dimensions for presentation purposes, and are used as data views of the supervisor or regulator

Note 1 to entry: They are defined as extended linkroles that have a child node <link:usedOn>link:presentationLink</link:usedOn>.

2.13
public elements
generalisation of a concept of a DPM, which are identified by a code and described by an appropriate label.

2.14
substitutiongroup
XML attribute to classify a complexType of simpleType element so its construction can be re-used by other elements

2.15
tablegroup
set of Tables that represents a business template

Note 1 to entry: In XBRL terms, tableGroups are public elements represented using XBRL abstract items with @type set to 'model:tableGroupType'.

2.16
taxonomy
part of the DPM that represent a version of the framework

Note 1 to entry: Like a 'name' for the set of definitions used in the framework that are not part of the dictionary.

Note 2 to entry: There is a virtual link with the XBRL term 'taxonomy', which is a set of concepts with their relationships and definitions. The XBRL taxonomy has a much wider definition than the EXTA one. Taxonomies are public elements represented using XBRL abstract items with @type set to 'model:taxonomyType' in schema 'tax.xsd'.

3 European XBRL Taxonomy Architecture

3.1 Supporting concepts

3.1.1 Taxonomy levels and owners

Taxonomy concepts could be defined on one of three levels:

1) cross sector concepts and breakdowns (to be shared between different institutions e.g., banking, insurance and securities supervision);

2) concepts from information requirements of an European supervisory authority;

3) concepts defined by a NSA.

Level one are cross sector concepts that shall be eventually defined and agreed by a joint group of experts involved in setting up information requirements for different sectors, such as banking, insurance, and reporting of other commercial and non-profit companies. It is expected that these concepts will mainly include the list of countries and currencies as defined by the ISO codes, and subsequently extended for example economy sectors, classes of financial instruments, accounting portfolios, time intervals, etc. This task will most likely be conducted by the reconciliation of dictionaries which have business concepts defined by the EBA, EIOPA and other interested parties. Cross sector concepts will be published on the Eurofiling website as a set of XBRL schema and linkbase files that can be referenced from level two or level three dictionaries.

Level two are the concepts of a European supervisory authority. They represent the information requirements of COREP, FINREP, Solvency II, and other scope-defined activities at the European level. They may refer to and extend the level one (cross-sector) concepts.

Level three are the concepts of national supervisors. They reflect information requirements set by a national legislation and specific supervisory regulations.

The idea of levels for concept definitions has been addressed in the XBRL taxonomy model by introducing the notion of the **owner**. The owner represents an institution that defines the concepts of the model. The owner is closely related to the idea of extensibility in XBRL. The main properties of the owner are:

— owner's namespace *{ons}*,

— owner's prefix *{opre}*, and

— official location *{oloc}*.

The owner namespace *{ons}* is a URI used to establish the namespace of the concepts defined by that owner. This URI is generally built by adding xbrl to the internet domain of the institution that the owner represents (e.g., "http://www.eba.europa.eu/xbrl").

The prefix *{opre}* is used as the basis to establish namespace prefixes in taxonomy files and for some short representations of the concepts by QNames using shorter prefixes (instead of long namespaces) plus the local name (e.g., "eba").

Official location *{oloc}* is a URL used to specify the location where taxonomy files associated to that owner are to be published. Different owners must have different official locations, even owners with the same internet domain/same namespace. The official location is generally built by adding three parts to the internet domain of the institution (e.g., "http://www.eba.europa.eu/eu/fr/xbrl"):

— representation of the geographical area covered by the institution (e.g., eu in case of the cross sector or the EBA concepts, or es for the national specific concepts applied in Spain);

— indication of the scope of information requirement (e.g., fr for general financial reporting);

— fixed xbrl component identifying the type of standard used to express information requirements.

Additional parts are possible based on the proprietary needs of a regulator

The following table presents examples of owners and applied namespaces, prefixes and official locations of taxonomy files.

Table 1 — Examples of namespaces, prefixes and official locations of taxonomies files for different owners

Owner	Namespace	Prefix	Official location
Cross-sector	http://www.eurofiling.info/xbrl	eu	http://www.eurofiling.info/eu/fr/xbrl
EBA	http://www.eba.europa.eu/xbrl	eba	http://www.eba.europa.eu/eu/fr/xbrl
Banco de España	http://www.bde.es/xbrl	se	http://www.bde.es/es/fr/xbrl

Copyright: Text used as a header in every taxonomy file published by its owner.

Supported languages: List of languages used in taxonomy files defined by an institution. It is used to deduce the location of label linkbases in a certain language given by the owner of the concept. This enables the addition of labels to concepts imported from other taxonomies [EBA12, p. 6].

3.1.2 Filing indicators

Filing indicators serve the purpose of communicating the scope of the reported data based on templates. The main purposes of filing indicators are:

— provide hints to applications using the taxonomy, relative to which templates/forms are included in the filing and, for example, shall be displayed to users;
— trigger execution of business rules (value assertions) to be run on a filing in order to check its correctness, depending on the reported scope of data [Bund13, p. 9].

In technical terms, filing indicators are facts included as part of an instance document where the filer states which templates/forms are being reported. Each template/form is represented as an instance fact of the item “table” under the “fIndicators” tuple element. These elements are defined in the namespace *http://www.eurofiling.info/xbrl/ext/filing-indicators.* The official location of this schema file is *http://www.eurofiling.info/eu/fr/xbrl/ext/filing-indicators.xsd.* This schema file is imported in every taxonomy module [EBA12, p. 21].

Throughout this document, the prefix *find* will be used to make reference to this schema namespace.

The following instance excerpt represents a filing with information about templates/forms with codes C_03.00 and C_22.00:

```
<find:fIndicators>
   <find:fIndicator contextRef="ctx">C_03.00</find:fIndicator>
   <find:fIndicator contextRef="ctx">C_22.00</find:fIndicator>
</find:fIndicators>
```

Context to which facts representing *find:fIndicator* element refer must identify the reporting entity and use the instant date which is the reference date of the reporting.

Table codes to be used on *find:fIndicator* facts are the ones identified by {template code} component of the documentation label of *tabel:table* resource (see 6.4.5.3 Tables). If a template results in multiple tables (as a

result of its original arrangement in multiple physical tables or due to normalization process), it is identified by *find:fIndicator* fact only once [Bund13, pp. 9].

This approach enables a clear separation between business facts (under the xbrli:xbrl element) and additional data required in the reporting process. Moreover, it does not require the addition of filing indicators to the dictionary of concepts, as filing indicators are defined in a generic way in its own schema.

Filing indicators include a Boolean attribute named “find:filed” with default value “true”. Tables not filed can be omitted from the list of filing indicators or have an entry with the “find:filed” attribute equal to “false”. The latter is especially useful if a footnote explaining the reasons for not filing a table want to be included [EBA12, p. 21].

3.1.3 Model supporting schema

The XBRL representation of the model makes use of some schema definitions in the namespace *http://www.eurofiling.info/xbrl/ext/model*. The official location of this schema file is *http://www.eurofiling.info/eu/fr/xbrl/ext/model.xsd*. Throughout this document, the prefix “*model*” will be used to make reference to this schema namespace [EBA12, p. 7].

3.1.4 Namespaces

The following table shows the prefixes used throughout this document as an abbreviated reference to namespaces.

Table 2 — Prefixes and namespaces of the XBRL technical files referenced in this document

Prefix	Namespace
xbrli	http://www.xbrl.org/2003/instance
xbrldt	http://xbrl.org/2005/xbrldt
link	http://www.xbrl.org/2003/linkbase
xl	http://www.xbrl.org/2003/XLink
gen	http://xbrl.org/2008/generic
iso4217	http://www.xbrl.org/2003/iso4217
nonnum	http://www.xbrl.org/dtr/type/non-numeric
num	http://www.xbrl.org/dtr/type/numeric
model	http://www.eurofiling.info/xbrl/ext/model
find	http://www.eurofiling.info/xbrl/ext/filing-indicators
pvar	http://www.eurofiling.info/xbrl/ext/pivot-variable
iaf	http://www.eurofiling.info/xbrl/functions/interval-arithmetics
variable	http://xbrl.org/2008/variable

[Source: EBA12, p. 7]

3.2 Public elements

Public elements are all concepts based on a DPM that are identified by a code in a certain scope and may include some additional information, such as readable labels, definitions and legal references in different languages.

Public elements include two attributes to reflect their creation date: (*model:creationDate*) and the date when they was last modified (*model:modificationDate*).

Language specific information of public elements is represented using the following label resources: XBRL 2.1 labels (link:label) for xbrli:items (or derived) public elements,

- generic labels (label:label) for public elements represented as XLink resources or other construct (e.g. link:roleTypes).
- The default (standard) role (*http://www.xbrl.org/2003/role/link*) is used for the extended links containing the label resources.

The role types used as roles for generic and standard label resources are presented in the following table.

Table 3 — Role types used as roles for generic and standard label resources

Property	Generic label role	Standard label role
Name	http://www.xbrl.org/2008/role/label	http://www.xbrl.org/2003/role/label
Definition	http://www.xbrl.org/2008/role/verboseLabel	http://www.xbrl.org/2003/role/verboseLabel
Legal references	http://www.xbrl.org/2008/role/documentation	http://www.xbrl.org/2003/role/documentation

The labels of the concepts of a schema or a linkbase file are represented jointly in a separate label linkbase file distinguished by language, in the same folder as its corresponding schema or linkbase file. Each language is stored in a separate label linkbase file. The naming convention for these label linkbase files is:

- {main-file}-lab-{lang}.xml

where {main-file} corresponds to the name of the schema or linkbase file where the concept is defined (without extension) and {lang} component represents the ISO 639-1 code of the language (lowercase).

In addition to this, some concepts of the dictionary may contain a special linkbase to represent codes needed for different purposes. In particular, the codes given to the columns and rows of tables or codes assigned to the data points are represented using this mechanism. The name of these linkbase files constructed as follows:

- {main-file}-lab-{lang}-codes.xml or {main-file}-lab-codes.xml

In the case of the later, the distinction between codes appearing in different language versions is determined basing on xml:lang attribute on label resources. The labels for these codes are represented as resources with a custom role. The role defined in the model.xsd schema for resources representing codes of rows or columns is http://www.eurofiling.info/xbrl/role/rc-code [EBA12, pp. 8].

3.3 Dictionary of concepts

The core concepts of the dictionary are metrics, dimensions, domains and domain members. Secondary concepts are families and perspectives (auxiliary concepts meant to group dimensions for presentation purposes). All the concepts in the dictionary are public elements.

To cope with changes in reporting, in addition to the properties and language specific information of public elements, dictionary elements include two optional attributes that establish its currency period: the starting date of the period interval (*model:fromDate* attribute) and its end date (*model:toDate* attribute). If the model:fromDate attribute is not included, then the concept is assumed to be valid for any period prior to the model:toDate attribute. If the *model:toDate* attribute is not included, then the concept is assumed to be valid for any period after the model:fromDate attribute. If neither *model:fromDate* nor *model:toDate* attributes are included, then the concept is assumed to be current for any period of time. An example is given in table 4.

Examples of location, target namespace and its prefix for dictionary concepts of the taxonomy are listed below:

Table 4 — Examples of dates of validation in the concepts.

Concept	model:fromDate	model:toDate	Dates of validation
Assets	2011/2/13	2012/2/13	Only valid from 2011/2/13 to 2012/2/13
Liabilities	2011/2/13		Valid from 2011/2/13
Real Estate		2012/2/13	Valid only until 2012/2/13
Profit			Valid forever

The first versions of the dictionary will not generally include these attributes. As new versions are released, and some concepts become obsolete and replaced by others, these attributes will be updated. These attributes don't have any impact on the reporting process itself; they are meant to simplify the management of the concepts of the dictionary.

All files in the dictionary of concepts are placed under the folder *dict* in the official location of its owner (see Table 3). Its namespace is obtained by adding a suffix that depends on the type of element to the namespace of the owner. The prefix to represent that namespace is obtained by adding a predefined suffix to the prefix of its owner (as presented in Table 5) where *{oloc}* represents the official location of taxonomy files of the owner of the concepts, *{ons}* its base namespace, *{opre}* the prefix of its base namespace, and *{dc}/{DC}* the code of a domain in lower and capital case.

Table 5 — Pattern for location, target namespace and its prefix for dictionary concepts

Dictionary concept	Official location	Target namespace	Namespace prefix
Metrics	{oloc}/dict/met/met.xsd	{ons}/dict/met	{opre}_met
Dimensions	{oloc}/dict/dim/dim.xsd	{ons}/dict/dim	{opre}_dim
Explicit domains	{oloc}/dict/dom/exp.xsd	{ons}/dict/exp	{opre}_exp
Typed domains	{oloc}/dict/dom/typ.xsd	{ons}/dict/typ	{opre}_typ
Explicit domain members	{oloc}/dict/dom/{dc}/mem.xsd	{ons}/dict/dom/{DC}	{opre}_{DC}

Examples of location, target namespace and its prefix for dictionary concepts of the taxonomy are listed below:

Table 6 — Examples of location, target namespace and its prefix for dictionary concepts

Dictionary concept	Official location	Target namespace	Namespace prefix
Cross-sector metrics	http://www.eurofiling.info/xbrl/dict/met	http://www.eurofiling.info/eu/fr/xbrl/dict/met/met.xsd	eu_met
Cross-sector dimensions	http://www.eurofiling.info/xbrl/dict/dim	http://www.eurofiling.info/eu/fr/xbrl/dict/dim/dim.xsd	eu_dim
EBA explicit domains	http://www.eba.europa.eu/xbrl/dict/exp	http://www.eba.europa.eu/eu/fr/xbrl/dict/dom/exp.xsd	eba_exp
EBA typed domains	http://www.eba.europa.eu/xbrl/dict/typ	http://www.eba.europa.eu/eu/fr/xbrl/dict/dom/typ.xsd	eba_typ
ES explicit domain members example (domain CP)	http://www.bde.es/xbrl/dict/dom/CP	http://www.bde.es/es/fr/xbrl/dict/dom/CP	es_CP
EBA families	http://www.eba.europa.eu/xbrl/dict/fam	http://www.eba.europa.eu/eu/fr/xbrl/dict/dim/fam.xsd	eba_fam
EBA perspectives	http://www.eba.europa.eu/xbrl/dict/pers	http://www.eba.europa.eu/eu/fr/xbrl/dict/dim/pers.xsd	eba_pers

[Source: EBA12, pp. 9]

3.3.1 Metrics

Metrics define the nature of the measure to be performed. Metrics determine the data type, the period type (instant / duration) plus additional semantics of their corresponding data points. Metrics are represented in XBRL as primary items and defined in schema files named met.xsd that reference label linkbase files. A local name of metrics is composed of three components:

4) a letter that represents the data type in lower case (for available options see the table below),

5) a letter that represents the period type: i for instant and d for duration,

6) a number that corresponds to the numeric code in the model (no zero padding or predetermined length).

Table 7 — Model and XBRL data type, local name codification letter and reporting unit

Model data type	XBRL data type	Local name codification letter	Reporting unit
Monetary (currency)	xbrli:monetaryItemType	m	Adequate currency using ISO 4217 codification (e.g.: iso4217:EUR)
Percent	num:percentItemType	p	xbrli:pure
Decimal	xbrli:decimalItemType	p	xbrli:pure

Integer	xbrli:integerItemType	i	xbrli:pure
Date	xbrli:dateItemType	d	not applicable
Boolean	xbrli:booleanItemType	b	not applicable
Text	xbrli:stringItemType	s	not applicable
Explicit domain	xbrli:qnameItemType	e	not applicable
Typed domain	typed domain corresponding data type, e.g. xbrli:stringItemType if a typed domain is xs:string		

In case of domain based data types, an additional attribute (*model:domain*) is included to identify the qualified name of the explicit or typed domain (e.g. *model:domain="eba:GA"*). Where the acceptable set of values for such a metric is a subset of the full set of values within an explicit domain, an additional attribute (model:hierarchy) is included to identify the URI of the role of a hierarchy containing the acceptable subset of domain values.

The id of the element (necessary for XLink locators) is composed as:

{opre}_{name}

where *{opre}* represents the prefix of the base namespace of the owner of the base item and {name} represents the name described above.

The following table contains a few examples of metrics declared in a taxonomy:

Table 8 — Examples of metrics in the taxonomy

Owner	Data / period type	Code	Name	Id	Namespace	Prefix
Cross-sector	Boolean / duration	3	bd3	eu_bd3	http://www.eurofiling.info/xbrl/dict/met	eu_met
Cross-sector	Monetary / duration	7	md7	eu_md7	http://www.eurofiling.info/xbrl/dict/met	eu_met
EBA	Monetary / instant	7	mi7	eu_mi7	http://www.eba.europa.eu/xbrl/dict/met	eba_met
EBA	Text / instant	7	si7	eu_si7	http://www.eba.europa.eu/xbrl/dict/met	eba_met
BdE	Boolean / duration	3	bd3	es_bd3	http://www.bde.es/xbrl/dict/met	es_met
BdE	Monetary / duration	7	md7	es_md7	http://www.bde.es/xbrl/dict/met	es_met

[Source: EBA12, pp. 10]

Since a DPM is the source of a transformation process with an XBRL taxonomy as its target graph, transformations are being used to describe how the structure of a DPM, represented as class diagram in UML, is mapped to an UML class diagram visualizing the underlying XBRL structure. To express the model transformation, the UML source model of the DPM and the UML target model for representing the XBRL structure are given. Between the two graphs, an abstract transformation syntax is used to describe the transformation rules from the source to the target model. The graph transformation intends to facilitate the understanding of the correlation between data structures reflected in Data Point Models, and how these

objects and structures can be matched to an XBRL data format. The graph transformation is limited to an essential part of the whole process. UML class diagrams reflecting the XBRL specification structure are visualized in the Aspect Model 2.0 of XBRL International.

The transformation process starts from the DPM UML graph on the left-hand side marked in blue to the UML class diagram using red squares to visualize the XBRL structure. The black arrows between both UML class diagrams are not part of the general UML language, but customized extensions which are used to describe the graph transformation. The square between two black arrows contains an abbreviation of what is being mapped. We distinguish eleven different types of mappings rules between the two graphs [Tae+, p. 1-9]:

- C2C - transformation from an UML class in the DPM meta-model in an UML class representing an XBRL concept,
- C2A - transformation from UML class in the DPM meta-model to an attribute of an XBRL element,
- C2R - transformation from UML class in the DPM meta-model to a XBRL resource as part of an XBRL linkbase,
- A2A - transformation from an attribute of a DPM meta-class to an attribute of an XBRL element,
- A2R - transformation from an attribute of a DPM meta-class to a XBRL resource as part of an XBRL linkbase,
- A2E - transformation from an attribute of a DPM meta-class to its XML element representation in XBRL,
- C2RS - transformation of an UML class of the DPM meta-model to an UML class representing a Relationship which is located in an XBRL linkbase,
- C2RSS - transformation of an UML class of the DPM meta-model to an UML class representing a set of Relationships which is located in an XBRL linkbase,
- RS2RS - transformation of an UML class of the DPM meta-model representing a Relationship to an UML class representing a Relationship which is located in an XBRL linkbase,
- C2Arc - transformation of an UML class of the DPM meta-model to an XBRL arc combining two XBRL elements, in the given examples resources,
- C2E - transformation of an UML class of the DPM meta-model to a table linkbase specific XBRL element which links to an XBRL concept.

In the following graph, depicted in Figure 1, transformations are used to show how the DPM objects of the meta-model are mapped to XBRL objects. By providing a more abstract view, these graphs should facilitate the understanding of the correlation between DPM and EXTA.

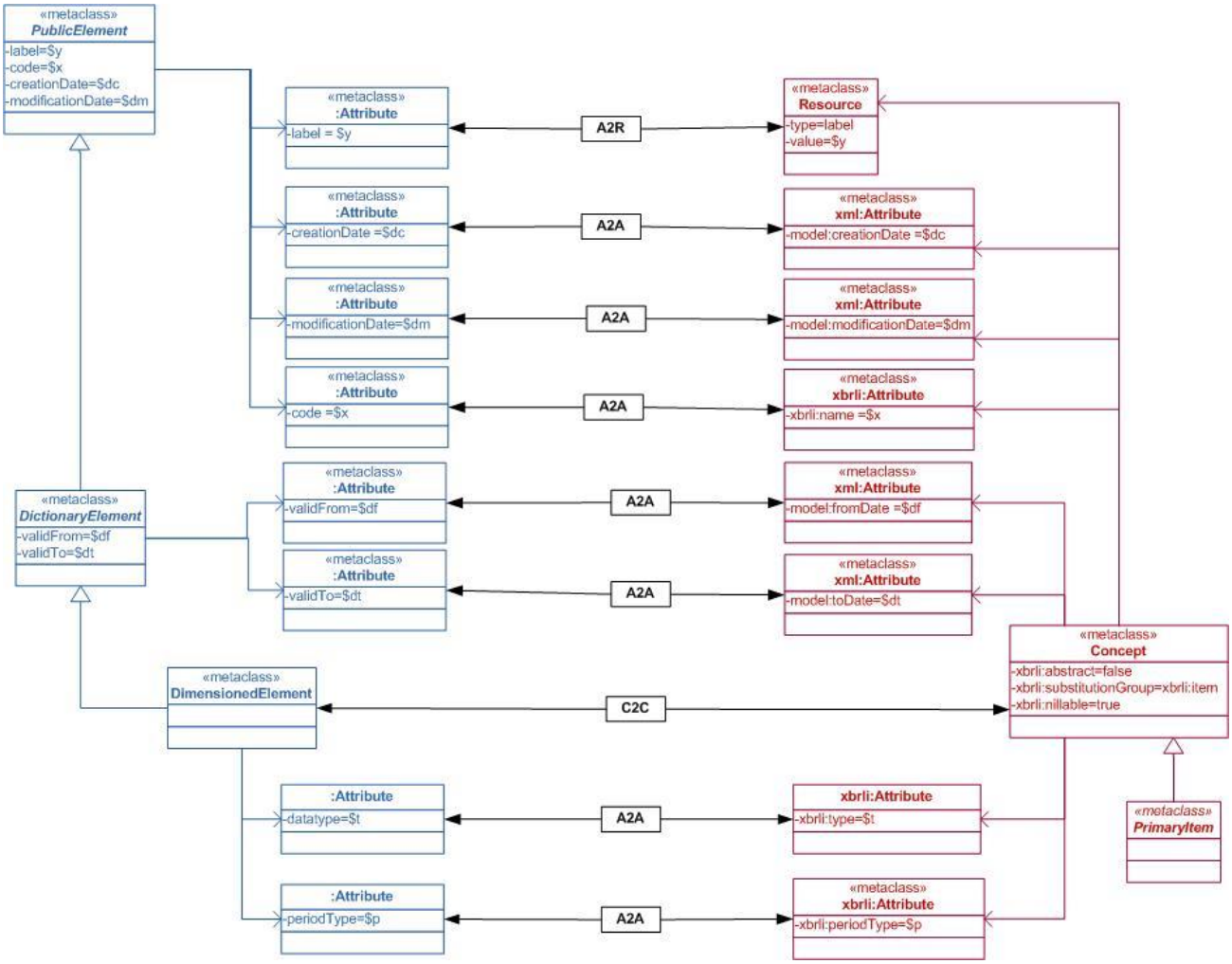

Figure 1 — Graph transformation of dimensioned elements of the DPM to primary items in XBRL

The graph above (Figure 1) shows how the meta-class DimensionedElement of the DPM meta-model is mapped to an XBRL concept with its corresponding XBRL attributes, according to the EXTA definitions. Dimensioned elements are known as metrics in the EXTA (synonym in XBRL: Primary item). Dimensioned elements are also part of the dictionary, so they inherit the attributes defined by the abstract concept of a dictionary element. The abstract dictionary element is a subordinated element of the abstract meta-class representing a public element, so the dimensioned Element also inherits the attributes defined by this superordinated element. In XBRL terms, a metric in the EXTA must include at least the mandatory attribute *model:creationDate* of the custom attributes defined in the model.xsd schema file.

3.3.2 Dimension items

Dimension items' representation in XBRL is defined in the XBRL Dimensions 1.0 specification.

The local name of each dimension corresponds to its code in the model: a short sequence of capital case letters (usually two, but may be more).

Value of the *id* attribute of the element representing a dimension item (necessary for XLink locators) is composed according to the following pattern:

```
    {opre}_{name}
```

where *{opre}* represents the prefix of the base namespace of the owner of the dimension and {name} represents the name described above. A few examples of dimension items are presented in the table below:

Table 9 — Examples of dimensions items

Owner	Code	Name	Id	Namespace	Prefix
Cross-sector	IC	IC	eu_IC	http://www.eurofiling.info/xbrl/dict/dim	eu_dim
EBA	PL	PL	eba_PL	http://www.eba.europa.eu/xbrl/dict/dim	eba_dim
EBA	MC	MC	eba_MC	http://www.eba.europa.eu/xbrl/dict/dim	eba_dim
BdE	RM	RM	es_RM	http://www.bde.es/xbrl/dict/dim	es_dim

Names of schema files defining dimension items is *dim.xsd,* and they include references to label linkbase files and a definition linkbase whose file name is *dim-def.xml,* and are placed in the same folder as the schema file.

[Source: EBA12, p. 11]

The graph below, depicted in Figure 2, visualizes the mapping between an enumerable dimension of a DPM, which is part of the dictionary, to an explicit dimension in XBRL. For each enumerable dimension defined, an abstract concept of substitution group xbrldt:dimensionItem is created in EXTA. The meta-class concept has a specialization to an abstract class "ExplicitDimension". This term is introduced for this specific concept in the XBRL Dimension 1.0 specification.

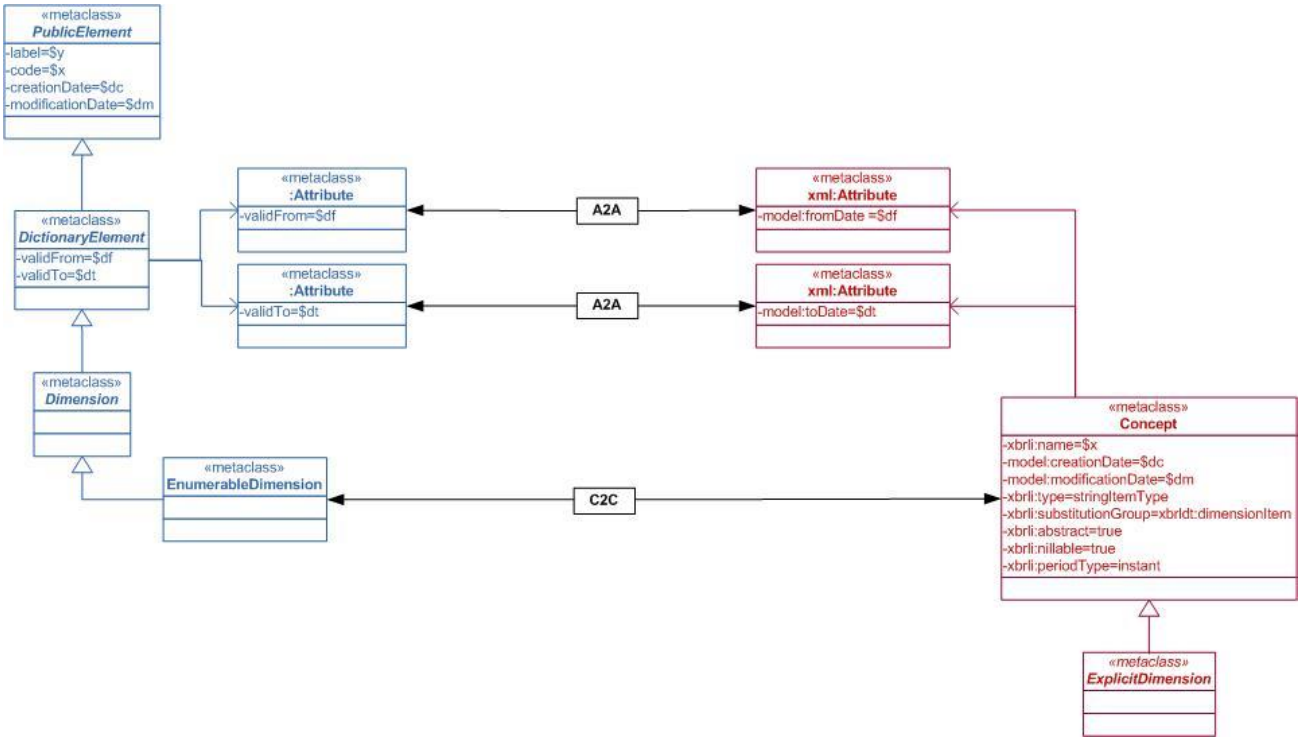

Figure 2 — Graph transformation of enumerable dimensions into explicit dimensions in XBRL

This definition linkbase includes the following information about explicit dimensions:

— reference to the domain associated with the dimension by means of a dimension-domain relationship (with *xbrldt:usable* attribute equal to false) pointing to a domain item defined in respective exp.xsd or typ.xsd schema file of any referenced or defined owner,

— reference to the default member of that dimension by means of a *dimension-default* relationship (note that although the model defines default members at the domain level, the XBRL Dimensions specification establishes this relationship at dimension level; thus, each dimension using a domain with a default member must include this relationship).

These relationships associating dimension with a domain and its default members are defined in an extended linkrole whose role is the standard one (i.e. http://www.xbrl.org/2003/role/link) [EBA12, p. 11].

The graph transformation depicted in Figure 3, shows how the dimensional objects relate to each other. The different relationships of XBRL forming a dimensional space are represented in the right hand side of the graph which illustrates the structure by an UML class diagram. To facilitate the understanding of the diagram, abstract classes for the dimensional objects are used, which are introduced throughout the graphs in this document. The relationship between a dimension and its default member could be included as meta-class DimensionDefaultRS in parallel to the DimensionDomainRS linking an abstract Dimension with a Member which correspondence Defined Member in the DPM has an attribute isDefault set to true.

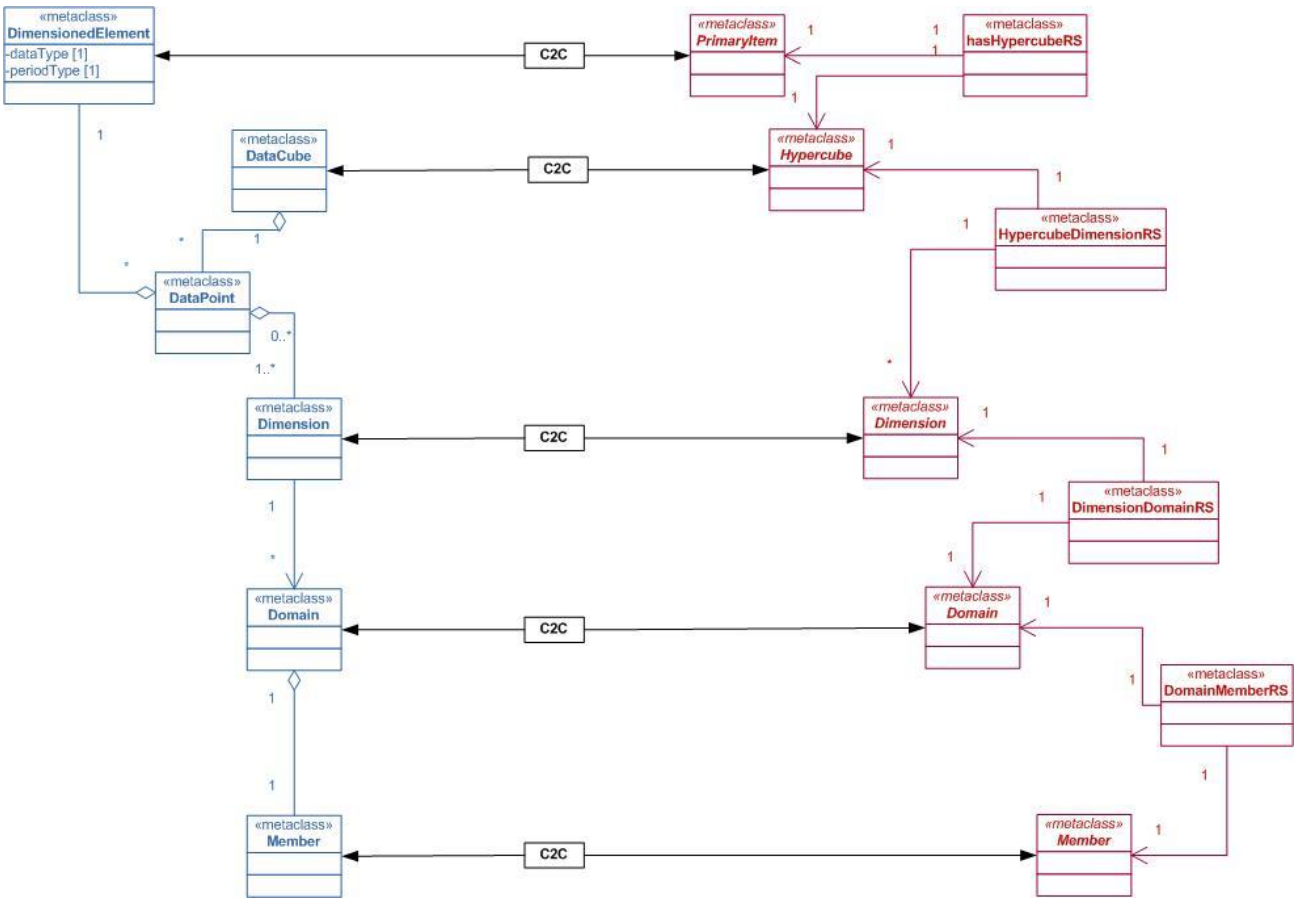

Figure 3 — UMLs representing the relations between dimensional object in DPMs and XBRL

[Source: XBRL12]

3.3.3 Families

Families are placed in the same folder as dimensions. Families are represented as XBRL abstract items of data type *model:familyType*. The local name of each family corresponds to its numeric code in the model preceded by a lower case letter f. Value of the id attribute of family elements is composed as follows:

```
{opre}_{name}
```

where *{opre}* represents the prefix of the base namespace of the owner of the family and {name} represents the local name described above. Some examples are presented in the table below.

Table 10 — Examples of families

Owner	Code	Name	Id	Namespace	Prefix
EBA	1	f1	eba_f1	http://www.eba.europa.eu/xbrl/dict/fam	eba_fam
EBA	2	f2	eba_f2	http://www.eba.europa.eu/xbrl/dict/fam	eba_fam
BdE	1	f1	es_f1	http://www.bde.es/xbrl/dict/fam	es_fam

[Source: EBA12, pp. 11]

The graph transformation, depicted in Figure 4, visualizes the mapping between the DPM object Family to an abstract XBRL concept of xbrli:item in the EXTA specific custom type family listed the model.xsd schema file. The DPM meta-class Family is introduced to group a set of dimensions that is related to a specific business point of view, and also has its correspondence in a taxonomy based on the EXTA.

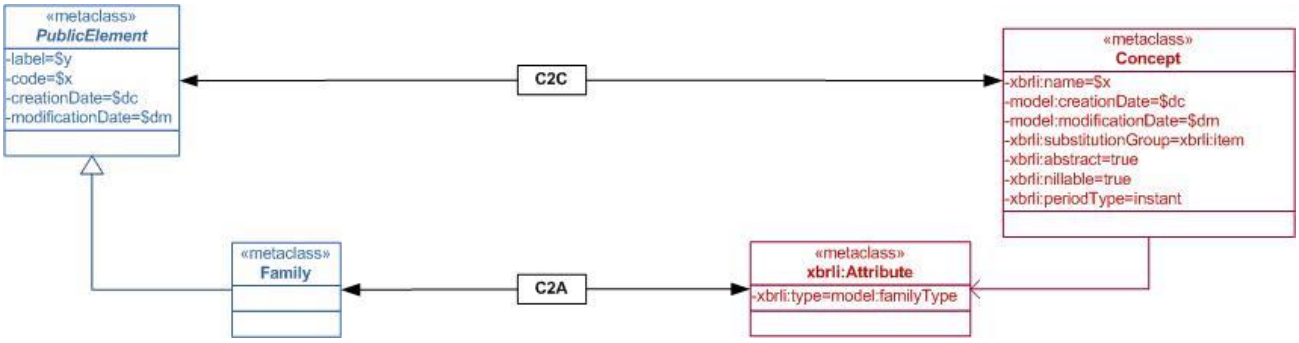

Figure 4 — Mapping between family objects in DPMs to concepts representing families in XBRL

3.3.4 Perspectives

Perspectives are placed in the same folder as dimensions and families. Perspectives are represented as extended link roles that are used in presentation linkbases, where the relationships between dimensions and families are formalized.

The URI of these roles is built according to the following pattern:

```
{ons}/role/dict/pers/{code}
```

where *{ons}* corresponds to the owner namespace and {code} represents the numeric code given to the perspective in the model. Value of the *id* attribute of the role type is composed according to the following pattern:

```
{opre}_p{code}
```

where *{opre}* corresponds to the owner prefix and {code} represents the numeric code of the perspective in the model.

Examples of perspectives are presented in the table below:

Table 11 — Examples of perspectives

Owner	Code	Role	Id
EBA	1	http://www.eba.europa.eu/xbrl/role/dict/pers/1	eba_p1
EBA	2	http://www.eba.europa.eu/xbrl/role/dict/pers/2	eba_p2
BdE	1	http://www.bde.es/xbrl/role/dict/pers/1	es_p1

The schema file defining perspectives (*pers.xsd*) imports the schemas of families and dimensions, and refers to a generic linkbase containing generic labels of perspectives and a presentation linkbase *(pers-pre.xml)* including extended links that correspond to each perspective. The arcs in these extended links have families as source (root) nodes and dimensions as target nodes [EBA12, p. 12].

3.3.5 Domains

Explicit domains are represented using XBRL abstract items of domain type *model:explicitDomainType* in the schema file *exp.xsd*.

Typed domains are represented as XML elements that are not in the substitution group of *xbrli:item*. These elements are defined in the schema file *typ.xsd*. Explicit domains are *xbrli:items,* whereas typed domains are not. Because of this, labels for the former ones are defined using standard label links and labels for the latter using generic label links.

The local name of each domain corresponds to its code in the model model *{dom-code}*: a short sequence of capital case letters (usually two, but may be more). Value of the *id* attribute of the element of the base namespace of the owner is composed according to the following pattern:

```
{opre}_{name}
```

where *{opre}* represents the prefix of the base namespace of the owner of the domain and *{name}* represents the name described above.

Some examples of domains items are presented in the table below:

Table 12 — Examples of domain items

Owner	Code	Element Name	Type	Id	Namespace	Prefix
Cross-sector	GA	GA	Explicit	eu_GA	http://www.eurofiling.info/xbrl/dict/exp	eu_exp
EBA	CO	CO	Explicit	eba_CO	http://www.eba.europa.eu/xbrl/dict/exp	eba_exp
EBA	MI	MI	Typed	eba_MI	http://www.eba.europa.eu/xbrl/dict/typ	eba_typ
BdE	CG	CG	Explicit	es_CG	http://www.bde.es/xbrl/dict/exp	es_exp
BdE	ICTX	ICTX	Typed	es_ICTX	http://www.bde.es/xbrl/dict/typ	es_typ

Although the namespace of explicit and typed domains is different, different local names should be used to avoid confusion [EBA12, pp. 12].

3.3.5.1 Explicit domain members and hierarchies

Explicit domain members are represented using XBRL abstract items of domain item type defined in the non-numeric set of types of the XBRL International type registry: *nonnum:domainItemType*.

The local name of each explicit domain member corresponds to its numeric code in the model preceded by a lower case x. Local names are XML schema tokens and thus, are not allowed to start with a numeric character. If the concept represented already has a widely accepted standard codification, like ISO codes or UN code lists, the local name will match the existing codification in lower case. More specifically, the following ISO codes are used:

- ISO 4217: standard currency codes composed of three alphabetical characters,
- ISO 3166-1 alpha-2: standard country codes composed of two alphabetical characters.

Value of the *id* attribute of explicit domain members follows the general rule:

```
{opre}_{name}
```

The default domain member of a domain (usually the one with numeric code component of the name set to 0) is marked with an attribute: *model:isDefaultMember="true"* [EBA12, p. 13].

The graph transformation, depicted in Figure 5, visualizes how explicit domains which are defined in a DPM are mapped to the EXTA taxonomy structure. A DPM to be generated in an EXTA taxonomy needs to follow specific rules which are summarized in the DPM Meta- model (chapter 6.6.3). One of these rules defines that all defined members must be referenced by a domain. If one of the member of this domain has the isDefault attribute set to true, this default member is the dimension default in XBRL terms for each dimension referring to this domain. If a dimension references only a sub-domain excluding this default member, no other default based on this subset of members is being defined.

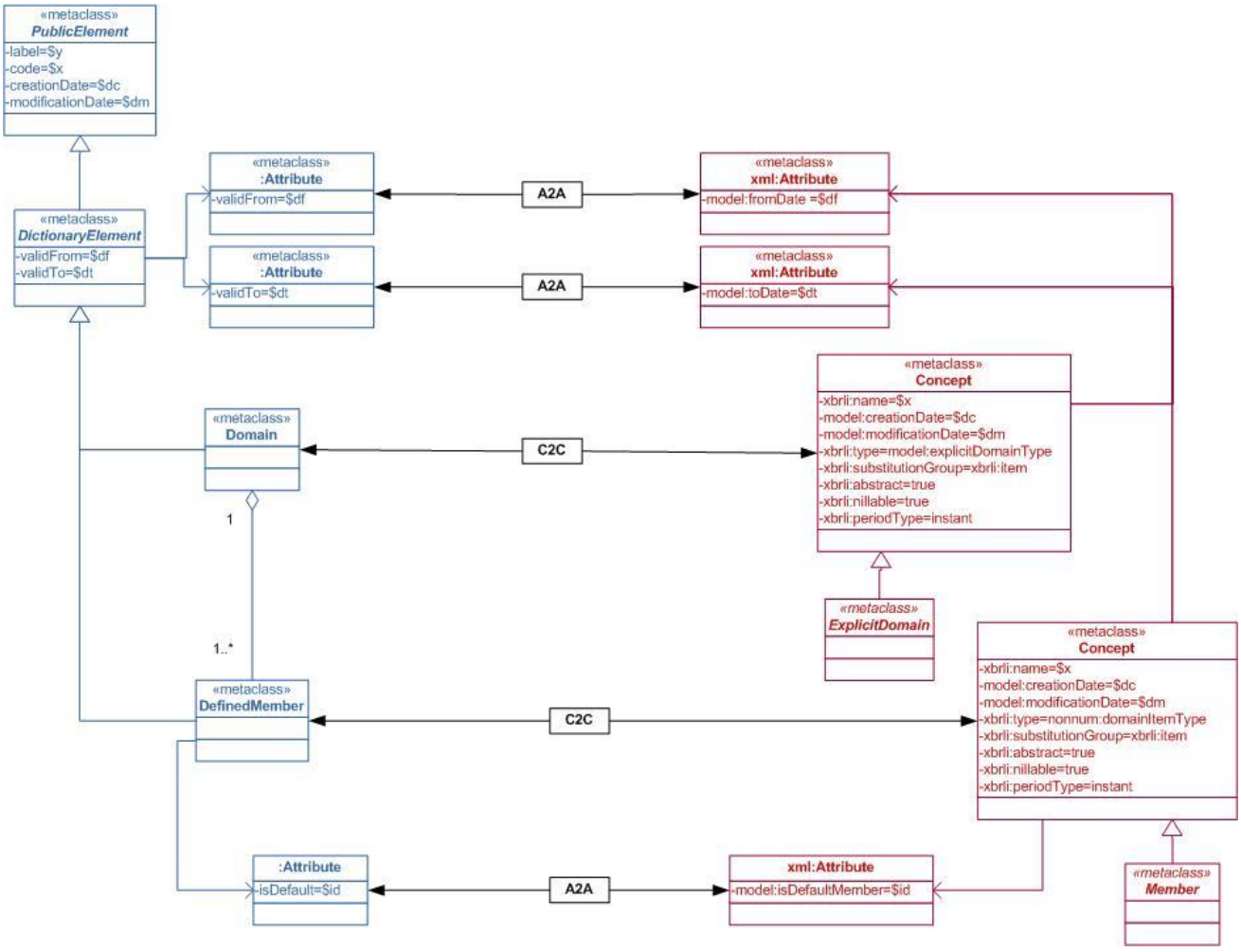

Figure 5 — Graph transformation representing the mapping of domains and defined members in DPMs to domain member relations in XBRL

The schema file that represents explicit members is placed in a folder with the name of its corresponding domain. The schema file for explicit domain members is called mem.xsd. Examples of schema files defining explicit domain members in the taxonomies are presented in the table below:

Table 13 — Examples of schema files defining explicit domain members

Owner	Domain code	Domain members schema	Namespace	Prefix
Cross-sector	GA	http://www.eurofiling.info/xbrl/dict/dom/ga/mem.xsd	http://www.eurofiling.info/xbrl/dict/dom/GA	eu_GA
EBA	CO	http://www.eba.europa.eu/xbrl/dict/dom/co/mem.xsd	http://www.eba.europa.eu/xbrl/dict/dom/CO	eba_CO
EBA	MI	http://www.eba.europa.eu/xbrl/dict/dom/mi/mem.xsd	http://www.eba.europa.eu/xbrl/dict/dom/MI	eba_MI
BdE	CG	http://www.bde.es/xbrl/dict/dom/cg/mem.xsd	http://www.bde.es/xbrl/dict/dom/CG	es_CG

[Source: EBA12, pp. 13]

Hierarchies are represented using XBRL extended link roles whose role type URI is built according to the following pattern:

```
{ons}/role/dict/dom/{dom-code}/{hierarchy-code}
```

where *{ons}* represents the namespace of the owner, *{dom-code}* represents the code of the domain and *{hierarchy-code}* the numeric code of the hierarchy. Value of *id* attribute of these roles is composed following the pattern: *{opre}_{code}* Examples of extended link roles used for hierarchies of domains are presented in the table below:

Table 14 — Extended link roles used for hierarchies of domains

Owner	Domain code	Hierarchy code	Role URI	Role id
Cross-sector	GA	1	http://www.eurofiling.info/xbrl/role/dict/dom/GA/1	eu_1
EBA	CO	1	http://www.eba.europa.eu/xbrl/role/dict/dom/CO/1	eba_1
EBA	CS	1	http://www.eba.europa.eu/xbrl/role/dict/dom/CS/1	eba_1
EBA	GA	2	http://www.eba.europa.eu/xbrl/role/dict/dom/GA/2	eba_2
BdE	CG	4	http://www.bde.es/xbrl/role/dict/dom/CG/4	es_4

The schema file that represents hierarchies (defining role types and referring linkbases) is placed in the same folder as members and it is named *hier.xsd*. Examples of such schema files, their namespaces and prefixes are presented in the table below:

Table 15 — Examples of schema files defining hierarchies for domain members

Owner	Domain code	Hierarchies schema	Namespace	Prefix
Cross-sector	GA	http://www.eurofiling.info/xbrl/dict/dom/ga/hier.xsd	http://www.eurofiling.info/xbrl/dict/dom/GA/hier	eu_GA_h
EBA	CO	http://www.eba.europa.eu/xbrl/dict/dom/co/hier.xsd	http://www.eba.europa.eu/xbrl/dict/dom/CO/hier	eba_CO_h
EBA	MI	http://www.eba.europa.eu/xbrl/dict/dom/mi/hier.xsd	http://www.eba.europa.eu/xbrl/dict/dom/MI/hier	eba_MI_h
BdE	CG	http://www.bde.es/xbrl/dict/dom/cg/hier.xsd	http://www.bde.es/xbrl/dict/dom/CG/hier	es_CG_h

Each hierarchy of domain members (represented by an extended link role as explained above) must indicate at least one dimension related to a hierarchy. Identification of this dimension is made:

— in the standard role of generic label (of a hierarchy) by indication of a dimension label (optional);

— in the documentation role of generic label (of a hierarchy) by indication of a QName (qualified name) of a dimension item declaration (obligatory).

Example of hierarchy reference to a dimension QName with documentation label role is presented below:

```
<link:loc xlink:type="locator" xlink:href="hier.xsd#eu_1" xlink:label="loc_eu_1" />
<label:label xlink:type="resource" xlink:label="label_eu_1_1" xml:lang="en"
xlink:role="http://www.xbrl.org/2008/role/documentation">eu_dim:CS</label:label>
<gen:arc xlink:type="arc" xlink:arcrole="http://xbrl.org/arcrole/2008/element-label"
xlink:from="loc_eu_1" xlink:to="label_eu_1_1" />
```

This relation between hierarchies and dimensions is “many-to-many”, which means that a number of dimensions can be attached to one hierarchy and more than one hierarchy can be referred by the same dimension.

In addition to labels, these schemas refer to three additional linkbases with information about hierarchies.

— a presentation linkbase *(hier-pre.xml)*, which represents the hierarchical disposition of members in hierarchies using *parent-child* relationships (like depicted in Figure 6),[EBA12, p. 14].

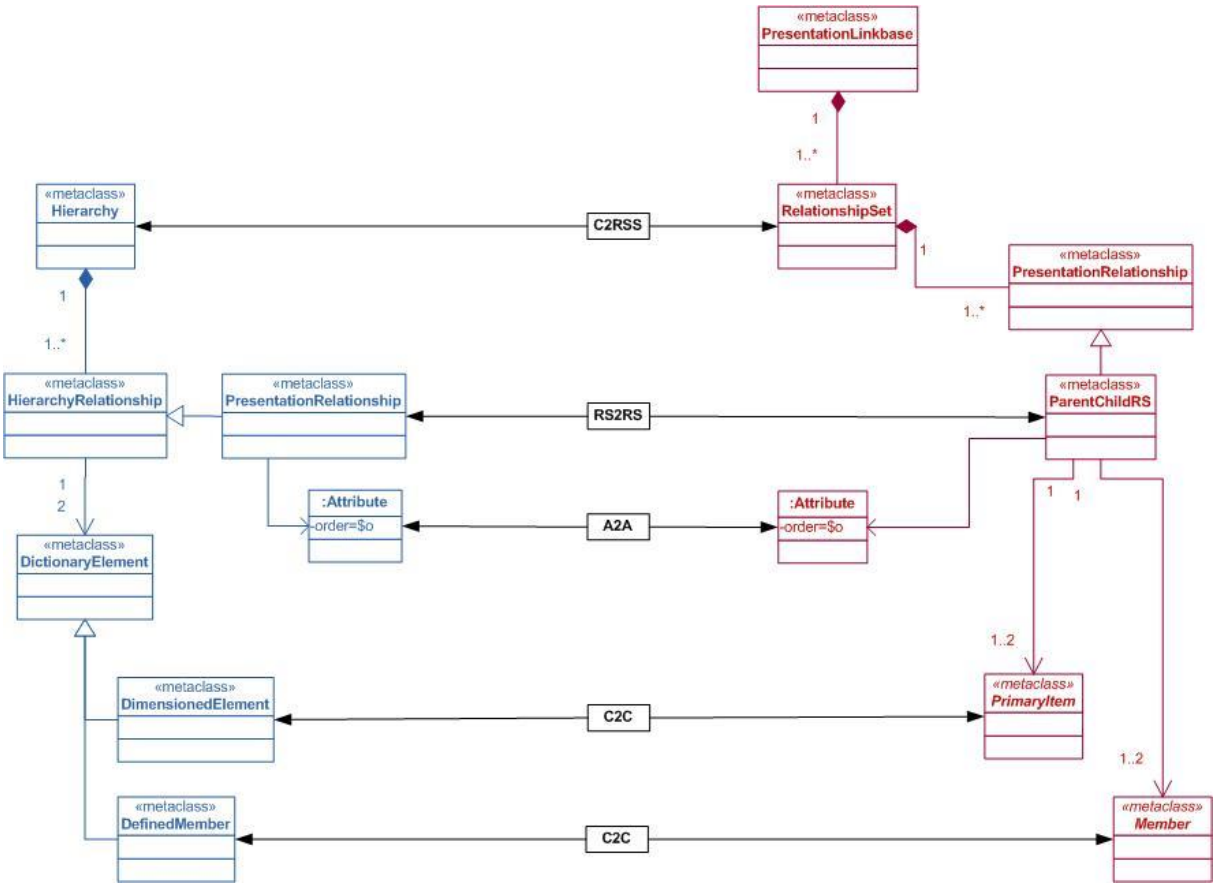

Figure 6 — Mapping between presentation relationships in DPMs to hierarchies defined in presentation linkbases in XBRL

In reference to the EXTA, the DPM meta model defines three different types of hierarchies: the presentation relationship, the basic relationship and the rule relationship.

The presentation relationship, which is shown in the graph transformation above, links the hierarchies defined for presentational purposes in the DPM to parent-child relationships defined in a presentation linkbase of an EXTA taxonomy. Primary items and domain members are built in separate parent-child relationships. There will be no mixing of primary items and domain members in the same presentation relationship. The given

order defined in the DPM will be transferred to the taxonomy by using the order attribute of the parent-child relationship.

— a definition linkbase *(hier-def.xml)*, which enables the inclusion of the members of a hierarchy in dimensional combinations using *domain-member* relationships, [EBA12, p. 14]

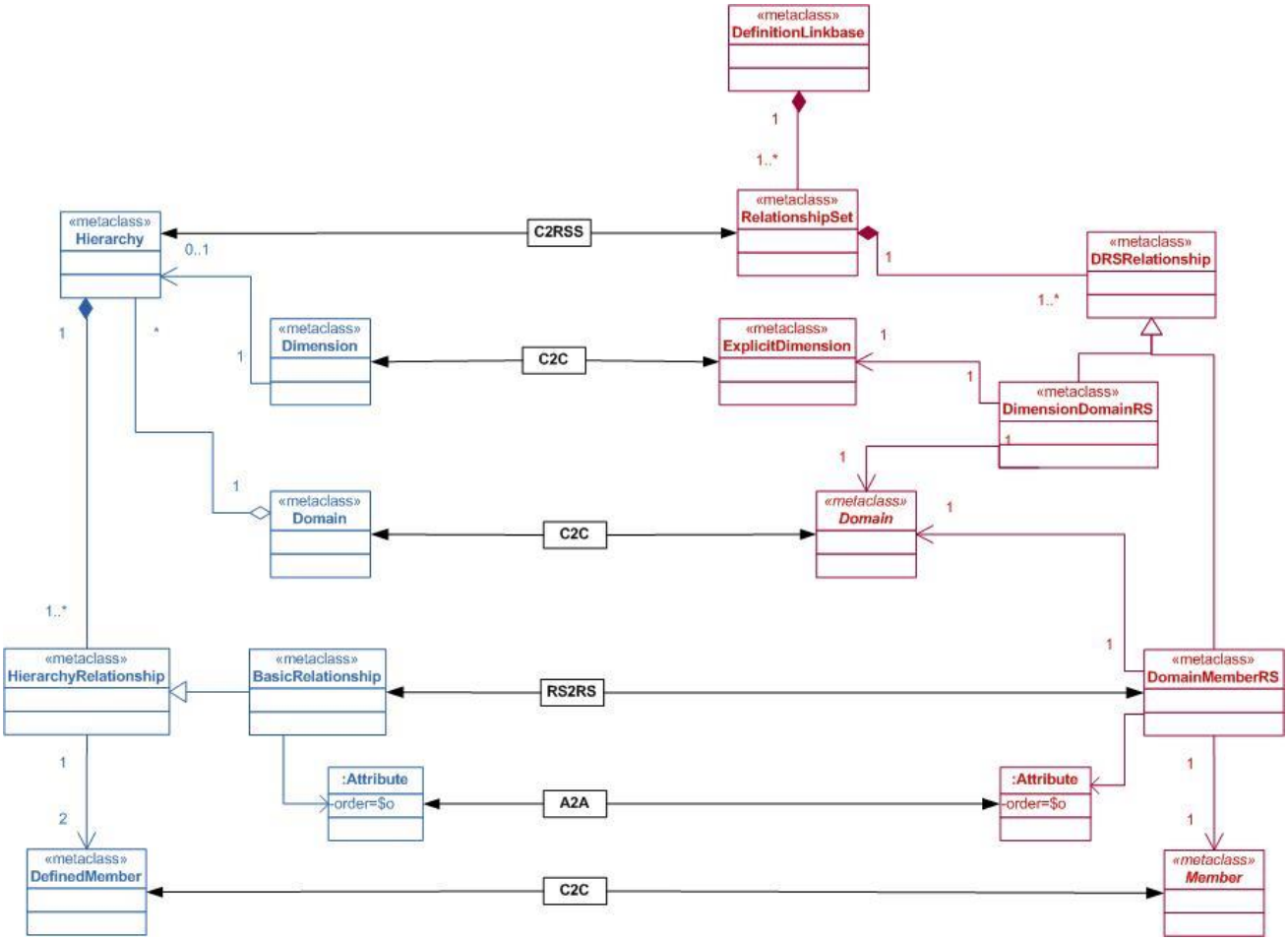

Figure 7 — Mapping of basic relationships in DPMs to domain-member relationships in XBRL

[Source: XBRL12]

In a DPM, each member that is listed under a domain is transformed into a domain-member relationship, included in a definition linkbase. The EXTA defines that the parent of this relationship is a domain and the child is a member. The concept representing a domain builds a dimension-domain relationship, with the dimension as the parent and the domain as the child. In general, XBRL allows linking a member directly to an explicit dimension and to define nested structures between members by using the order attribute. The EXTA limits this freedom by the constraint that an EXTA specific domain element is to be created to reference all defined members, with no specific relevance to the order of the members in a domain (Figure 8).

— a calculation linkbase *(hier-cal.xml)*, which establishes some basic arithmetical relationships between a member of the hierarchy and its children:

  — a member is equal to the addition of its child members in the hierarchy: *complete-breakdown* relationships,
  — a member is greater or equal than the addition of its child members in the hierarchy: *partial-breakdown* relationships,

— a member is less or equal than the addition of its child members in the hierarchy: *superset-breakdown* relationships [EBA12, p. 14].

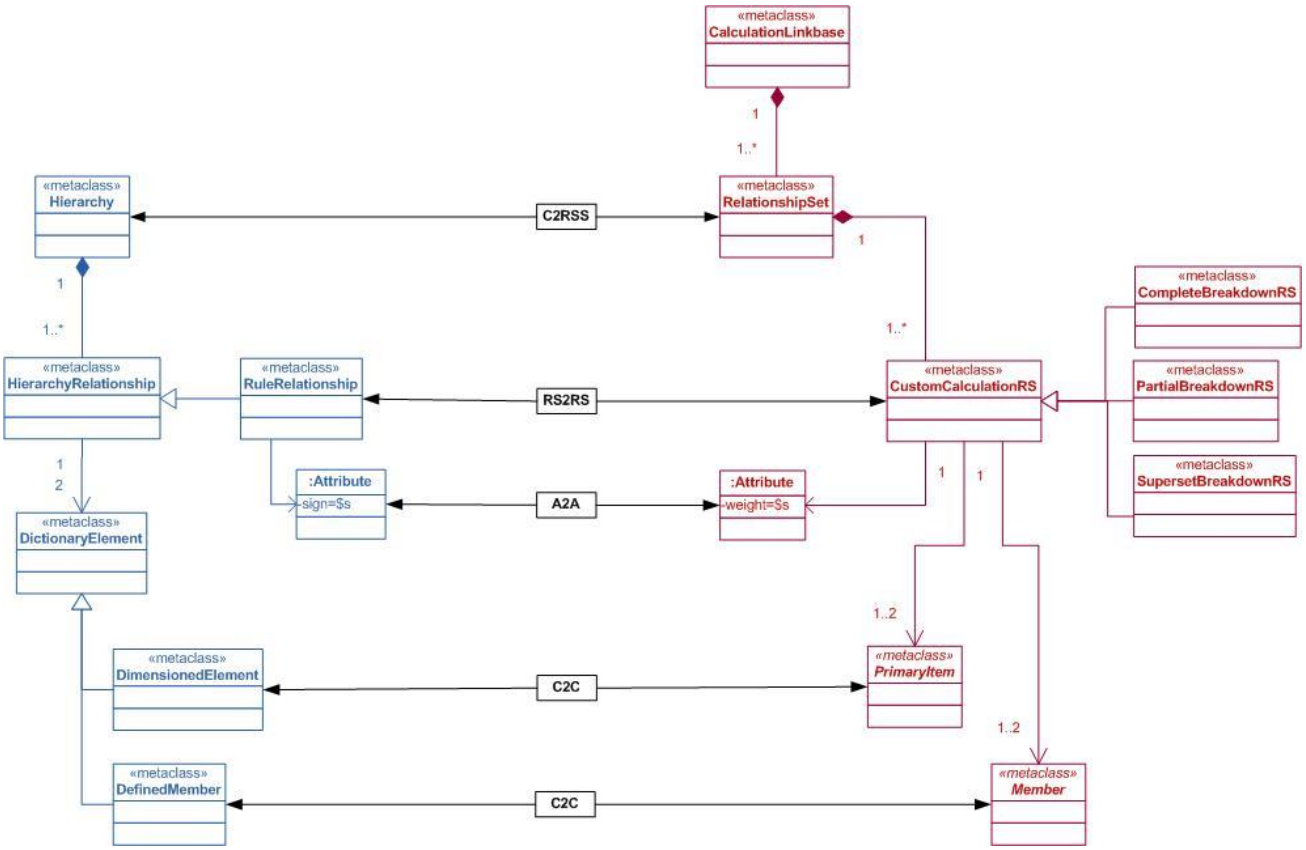

Figure 8 — Graph transformation of rule relationships into custom calculation relationships stored in calculation linkbases in XBRL

The arithmetical relationship between members is defined in a DPM by adding an appropriate value to the sign attribute on the rule relationship. Rule relationships can be defined between dimensioned elements that are mapped to primary items and between defined members that correspond to members of a domain. Depending on the sign of the relationships in the DPM, the relationships are mapped to custom calculation relationships that form a complete, a partial, or a superset breakdown. All relationship sets that are generated on the basis of the DPM definitions are summarized in the calculation linkbase. The calculation linkbase serves as a container. The defined structures are used to define assertions according to XBRL Formula 1.0 specification.

These calculation arcs include a weight attribute to indicate whether the child member contributes to the aggregation positively (+1) or negatively (-1). The roles that represent these calculation relationships are defined in the model.xsd schema that supports the model, and are presented in the table below.

Table 16 — Arcroles defined in the model.xsd schema, reflecting different forms of aggregations of members

Arc role id	Arc role URI
complete-breakdown	http://www.eurofiling.info/xbrl/arcrole/complete-breakdown
partial-breakdown	http://www.eurofiling.info/xbrl/arcrole/partial-breakdown
superset-breakdown	http://www.eurofiling.info/xbrl/arcrole/superset-breakdown

[Source: EBA12, p. 15]

The root member of the definition and presentation relationship networks is the domain item defined in the schema exp.xsd that is associated with the owner.

Domain members that extend the domain of another owner are placed in a folder preceded by the prefix of the extended owner. Examples of the extension of the cross-sector domain with EBA specific concepts are presented in the table below.

Table 17 — Examples of the extension of the cross-sector domain by with EBA specific concepts

Code	Extending domain members schema	Namespace	Prefix
GA	http://www.eba.europa.eu/eu/xbrl/dict/dom/eu_ga/mem.xsd	http://www.eba.europa.eu/xbrl/dict/dom/GA	eba_eu_GA
CU	http://www.eba.europa.eu/eu/xbrl/dict/dom/eu_cu/mem.xsd	http://www.eba.europa.eu/xbrl/dict/dom/CU	eba_eu_CU

[Source: EBA12, p. 15]

3.3.5.2 Typed domains

Members of typed domains are neither listed as XBRL items with labels, nor arranged in hierarchies. The content of typed domains is restricted by XML data type constraints, as each domain (according to the XBRL Dimension specification) is an XML element (see Figure 9).

In most of the cases, a typed domain would be represented by an XML element with a simple data type (e.g. *xs:string* or *xs:decimal*). Further restrictions on the expected value are introduced by means of the Formula specification, where value assertions define detailed constraints on the allowed values of a typed domain in association with a certain dimension. For example, the typed domain *Code*, being a string data type, may follow a certain pattern defined by a value assertion when referenced from the *Related entity code* dimension, and another pattern when used by the Instrument code dimension [Bund13, p. 17].

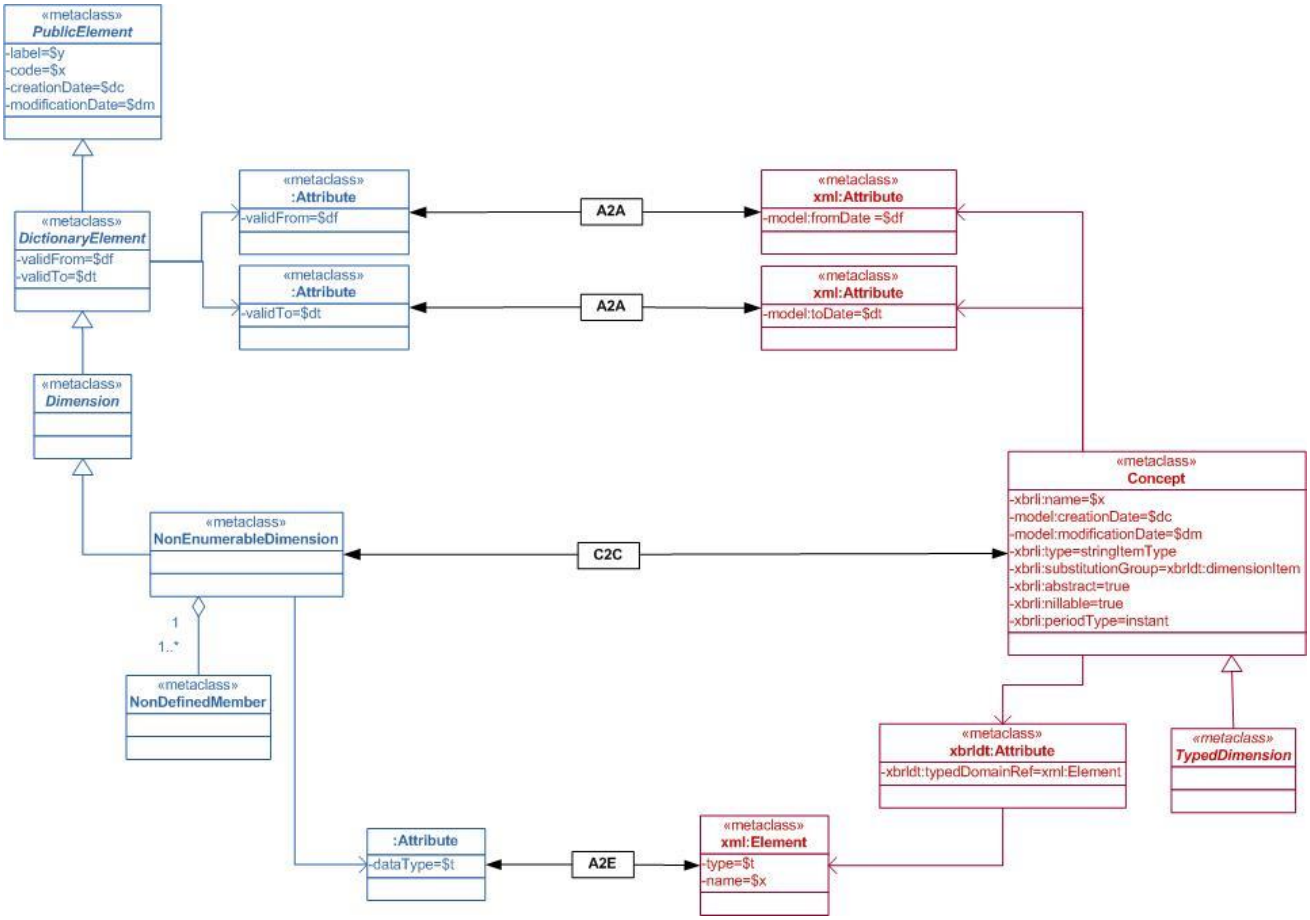

Figure 9 — Mapping between non-enumerable dimensions into typed dimensions in XBRL

The graph on the right-hand side in Figure 9 emphasizes the textual description above. In the DPM, the typed domain is represented by an attribute data type of the meta-class NonEnumerableDimension defining a constraint on the values. The DPM definition is mapped to a concept of xbrli:item with the substitution group dimensionItem, and forms together with the dimensional attribute typedDomainRef an XBRL representation of a typed dimension according to the XBRL Dimension 1.0. The data type is mapped, as described above, into an XML element which is referenced by the typed dimension.

3.4 Reporting requirements layer

Frameworks, taxonomies, tables, modules and other concepts constitute the layer of the model where the actual reporting requirements are specified, in accordance with the financial concepts defined in the dictionary.

All the files that correspond to this layer are placed under the folder *fws* in the official location of its owner. Its namespace is obtained by adding the suffix *fws* to the base namespace of the owner, plus some additional suffixes that depend on the type of the concept represented [EBA12, p. 16].

3.4.1 Frameworks

Frameworks are public elements represented using XBRL abstract items of the framework type (*model:frameworkType*) in the schema file *fws.xsd*. General framework properties are presented in the table below:

Table 18 — Framework properties

Schema property	Value
Official location	{oloc}/fws/fws.xsd
Target namespace	{ons}/fws
Target namespace prefix	{opre}_fws
Element local name	{framework}
Element id	{opre}_{framework}

The local name of each framework element corresponds to its code in the model, and its id follows the general pattern.

Examples of frameworks defined by the EBA taxonomies are presented in the following table:

Table 19 — Examples of frameworks

Schema property	Value
Official location	http://www.eba.europa.eu/eu/fr/xbrl/fws/fws.xsd
Target namespace	http://www.eba.europa.eu/xbrl/fws
Target namespace prefix	eba_fws
Local name example	finrep, corep
Element id example	eba_finrep, eba_corep

Each framework has a folder where the files of its taxonomies are placed. This folder has the name of its code in the mode,l as presented by the following examples:

Table 20 — Examples of framework folders

Description	Framework folder
EBA FINREP	http://www.eba.europa.eu/eu/fr/xbrl/fws/finrep/
EBA COREP	http://www.eba.europa.eu/eu/fr/xbrl/fws/corep/

[Source: EBA12, p. 16]

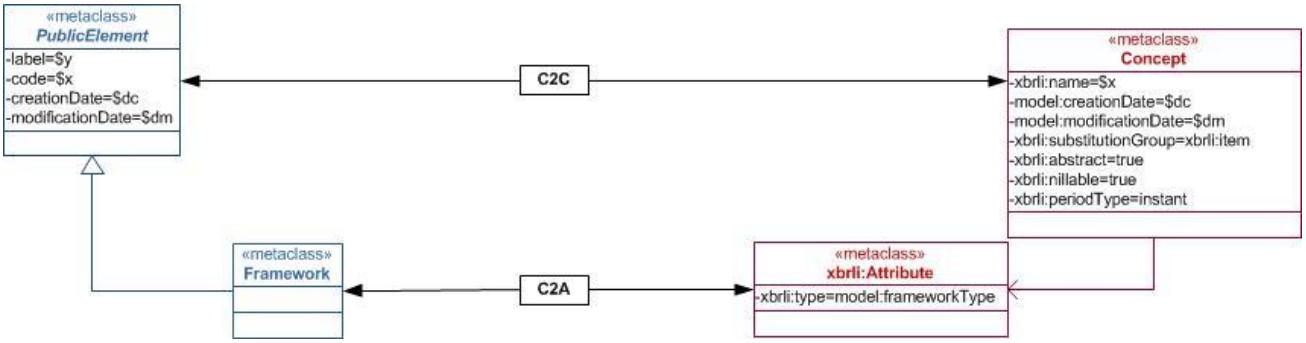

Figure 10 — Graph transformation from DPM to XBRL for frameworks

Because an EXTA taxonomy is a representation of a DPM in a formal language, the taxonomy holds all information defined. A DPM can be defined for more than one framework and can hold information for different taxonomy versions. All this meta information, apart from the elements that are defined to determine the content of an XBRL instance, needs to be specified in XBRL. The EXTA specifies abstract concepts to represent the DPM structure. They are just structural elements, where the attributes periodType as well as nillable, have no meaning. As they are concepts of XBRL item type, they can be linked to resources like labels or documentation.

In the graph transformation above, the DPM representation of a framework is mapped to a concept of the custom type framework defined in the model.xsd schema file.

3.4.2 Taxonomies

Taxonomies are public elements represented using XBRL abstract items of the taxonomy type (*model:taxonomyType*). These elements are stored in the schema file *tax.xsd* under the folder of its framework, a subfolder that corresponds to its normative code or legislation publication date (using the ISO 8601 codification), and another subfolder with the date of taxonomy version.

Thus, the file *tax.xsd* includes a single element. Its local name corresponds to its code in the model and the value of its id attribute is constructed according to the general pattern (*{opre}_{taxonomy code}*). General taxonomy properties are presented in the table below.

Table 21 — Taxonomy properties

Schema property	Value
Official location	{oloc}/fws/{framework}/{normative code or legislation publication date with “pub_” prefix}/{taxonomy publication date}/tax.xsd
Target namespace	{ons}/fws/{framework}/{normative code or legislation publication date with “pub_” prefix}/{{taxonomy publication date}
Target namespace prefix	{opre}_tax
Element local name	{taxonomy}
Element id	{opre}_{taxonomy}

To facilitate the specification of additional taxonomy resources in this document, the following statements apply:

— {**taxonomy-loc}** to the URL {oloc}/fws/{framework}/{normative code or legislation publication date with “pub_” prefix}/{taxonomy publication date};

— **{taxonomy-ns}** to the URI {ons}/fws/{framework}/{normative code or legislation publication date with "pub_" prefix}/{taxonomy publication date}.

The mapping from DPM to an EXTA XBRL structure shows how the meta-class Taxonomy, as the specialization of a public element, is transformed to a concept of custom type taxonomy. The attributes defined in the meta-class Taxonomy do not have a direct correspondence to the discoverable taxonomy set of XBRL. The attribute version data will be included in the normative reference folder (prefix *pub_*) inside the EXTA file structure. An example has been given in the graph below [EBA12, pp. 16].

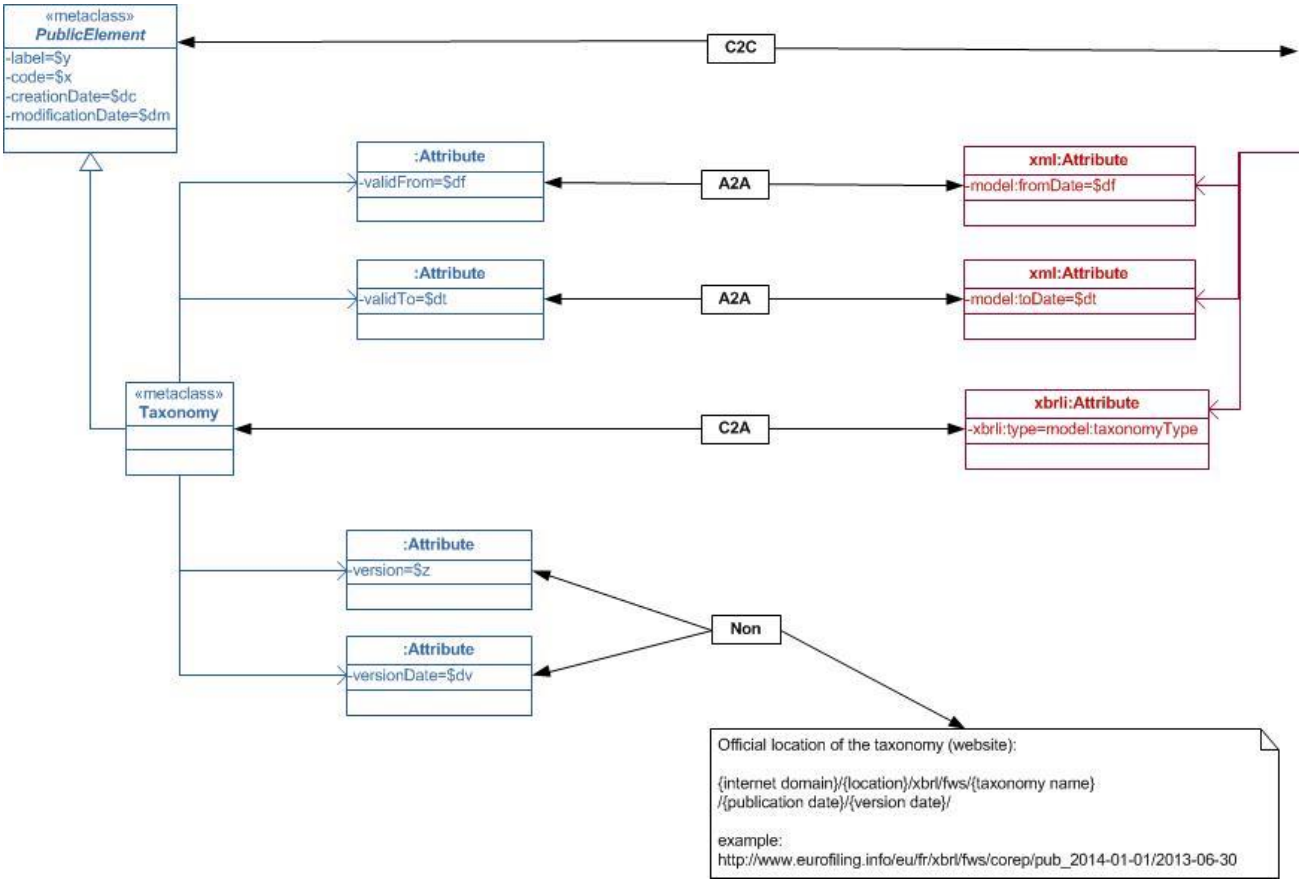

Figure 11 — Mapping between taxonomies in DPM into concepts representing taxonomies in XBRL

The folder of the taxonomy includes three subfolders for:

— tables (*tab*),

— modules (*mod*) and

— validations (*val*).

The content of these subfolders is explained in the next sections of this document [EBA12, p. 17].

3.4.3 Tables

The table folder includes a schema file (*tab.xsd*) that references a generic linkbase with the hierarchy of table groups and tables (*tab-pre.xml*), and a label linkbase for table groups (*tab-lab-en.xml*). The schema includes the definitions of table groups (if any), which are represented using XBRL abstract items of the table group

type (*model:tableGroupType*). Name of a table group item is composed by adding the prefix *tg* to the code of a table group in the model. General properties of a table group are presented in the table below:

Table 22 — Table group properties

Schema property	Value
Official location	{taxonomy-loc}/tab/tab.xsd
Target namespace	{taxonomy-ns}/tab
Target namespace prefix	{opre}_tab
Element local name	tg{table-group-code} or other custom name
Element id	{opre}_{local-name}

[Source: EBA12, p. 17]

The graph transformation (Figure 12) maps the meta-class TableGroup to an abstract concept of the custom type tableGroup listed in the model.xsd schema file. The tables grouped below have no correspondence as concepts. They are represented as resources of the rendering linkbase. Their mapping is described in the next model.

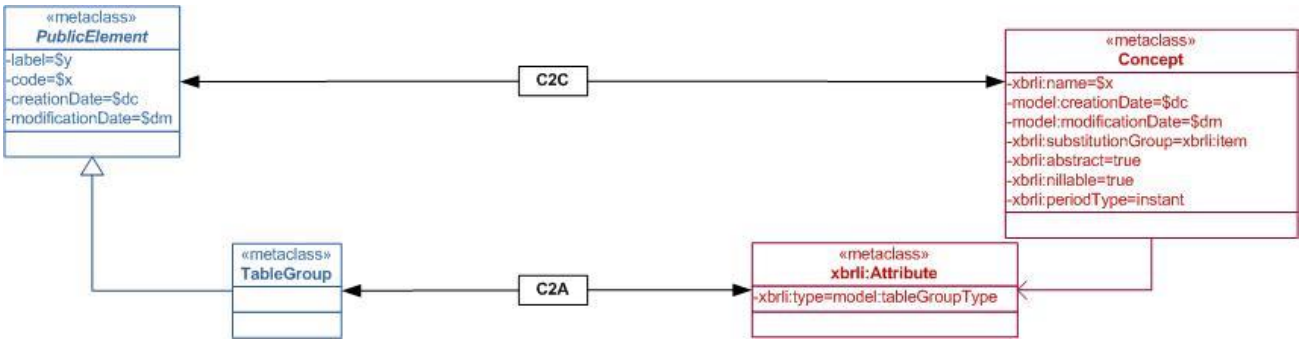

Figure 12 — Graph transformation from DPM to XBRL for table groups

Table groups are used to link numerous tables defined by one template, or resulting from the normalization of templates, or if an original template is composed of two or more physical tables.

The files that define the content of each table are placed in a folder whose name corresponds to the code of the table in the model (*{table code}*). General properties of a table are presented below:

Table 23 — General properties of a table

Schema property	Value
Official location	{taxonomy-loc}/tab/{table code}/{table code}.xsd
Target namespace	{taxonomy-ns}/tab/{table code}
Target namespace prefix	{opre}_tab_{table code}
Element local name	N/A (elements defined as resources in linkbases)
Element id	{opre}_{table code} (element defined as a resource in the table linkbase)

Schema file for a table refers to a table linkbase (*{table}-rend.xml*), a definition linkbase (*{table}-def.xml*), and a label linkbase (*{table}-{lang}-lab.xml*).

The table linkbase includes the definition of the table according to the Table Linkbase specification. The relationships of each table are placed in an extended link whose role is built according to the following pattern:

```
{ons}/role/fws/{framework}/{normative code or legislation publication date
with "pub_" prefix}/{taxonomy publication date}/tab/{table code}
```

In this linkbase, the different components of tables are represented using resources. Value of the *id* attribute of these resources is based on the code or sequential number plus a prefix to obtain a unique code in the context of the linkbase file.

Table 24 — Table linkbase resources and their ids

Model class	Table linkbase resource	Id
Table	Table	{opre}_t{code}
Predefined axis	ruleAxis (abstract = true)	{opre}_a{code
Variable axis	filterAxis	{opre}_a{code}
Coordinate	ruleAxis	{opre}_c{code}
Base items hierarchy reference	conceptRelationshipAxis	{opre}_h{code}
Dimension hierarchy reference	dimensionRelationshipAxis	{opre}_h{code}

[Source: EBA12, pp. 17]

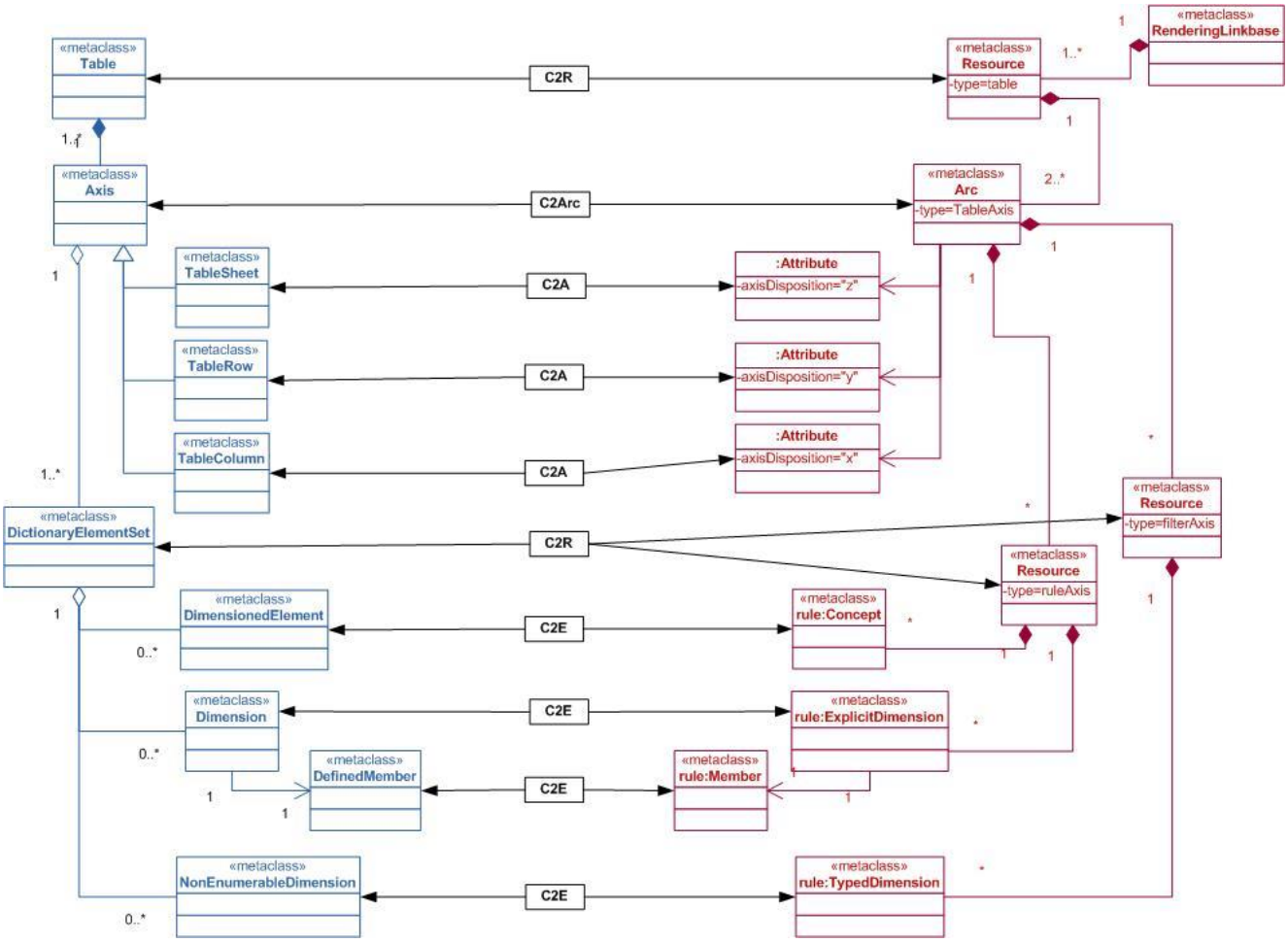

Figure 13 — Graph transformation showing the relations between tables defined in DPMs and their representation in elements of the table linkbase in XBRL

[Source: XBRL12]

The graph transformation from the DPM class diagram, representing the structure of a table, to an XBRL structure, describing a table with the components defined for the rendering linkbase (Figure 13). A table is represented in the current Public Working Draft of the Table Linkbase as an XBRL resource. The table resource is linked with two or more arcs that represent the table axis. Depending on the specific type of axis, the attribute axisDisposition is set to "x", "y" or "z". A table axis combines zero or more rules or filter axis, which are resource components inside linkbases. The content of these resources is compared to more complex labels. The resources include XML elements that refer to dictionary concepts. Whereas filter axes include only typed dimensions in EXTA, the rule axes can hold primary items as well as dimension member combinations. Due to the complexity of the model, the header label included in the presentation perspective of the DPM meta-model is not reflected in this model. The same applies to the table cell which is formed by the combination of the defined axes and their corresponding resources for a specific coordinate in the table.

The roles of the extended links necessary to express these combinations are built adding numeric suffixes to the role previously defined for the table. For example:

- *{ons}/role/fws/{framework}/{normative code or legislation publication date with "pub_" prefix}/{taxonomy publication date}/tab/{table code}/1*
- *{ons}/role/fws/{framework}/{normative code or legislation publication date with "pub_" prefix}/{taxonomy publication date}/tab/{table code}/2* [EBA12, p. 19]

According to the Table Linkbase specification, aspect rules are used to specify the concepts represented in predefined axes. In the case of duration type metrics, the size of the period to be reported is constrained using time aspect rules in the context of a table. The default value is a reference period. Reference period is defined as the period that starts at the beginning of the period covered by a report, and ends at the reference date. Other possible values are month, quarter, year, etc. In the case of data points that correspond to an instant or period of time prior to the reference period, temporal aspect rules include a temporal offset relative to the reference date.

The reference date and the reference period shall be defined in terms of two parameters defined in a linkbase file (*http://www.eurofiling.info/eu/fr/xbrl/func/params.xml*):

- *refPeriodEndDate*: reference date and end date of the reference period,
- *refPeriodStartDate*: starting date of the reference period.

Examples of use of time references are presented below:

Table 25 — Examples of time references

Time reference	Aspect rule
Reference date	instant = refPeriodEndDate
Reference period	duration.end = refPeriodEndDate duration.start = refPeriodStartDate
Beginning of the reference period	instant = refPeriodStartDate
A quarter ending at the reference	date duration.end = refPeriodEndDate duration.start = refPeriodEndDate - P3M + P1D
Previous quarter	duration.end = refPeriodEndDate - P3M duration.start = refPeriodEndDate - P6M + P1D

The definition linkbase includes dimensional relationships valid in the context of the table. Valid combinations are defined using only positive (*all*) closed hypercubes obtained from the set of valid cells of the table, following certain algorithms in order to optimize their number.

Each extended link role contains one or more primary items and a single hypercube. The model schema includes a hypercube element to be used. There is no need to define hypercube elements in each table or taxonomy. In the case of multiple primary items, the first one will be used to group the rest and reduce the number of *all* arcs. The domain element will be used as the target of *dimension-domain* arcs to avoid cycles. The *@xbrldt:targetRole* attribute might be necessary in the case of hypercubes with dimensions sharing the same domain [Bund13, pp. 23].

3.4.4 Modules

Modules are represented using XBRL abstract items of the module type (*model:moduleType*). Each module is stored in a different schema file, whose name is the same as the code of the module in the model plus the extension *.xsd*. These schema files import the schemas of all the tables required by that module and, additionally, the header taxonomy and filing indicators. General properties of a module are presented below:

Table 26 — Properties of modules

Schema property	Value
Official location	{taxonomy-loc}/mod/{module}.xsd
Target namespace	{taxonomy-bns}/mod/{module}
Target namespace prefix	{opre}_mod_{module}
Element local name	mod_{module}
Element id	{opre}_mod_{module}

[Source: EBA12, p. 19]

The meta-class Module of a DPM is mapped to an abstract concept of an EXTA custom type module (Figure 14). Modules serve as entry points of a taxonomy, so they control the validation of an XBRL instance. Modules can hold labels and documentation to describe their use case.

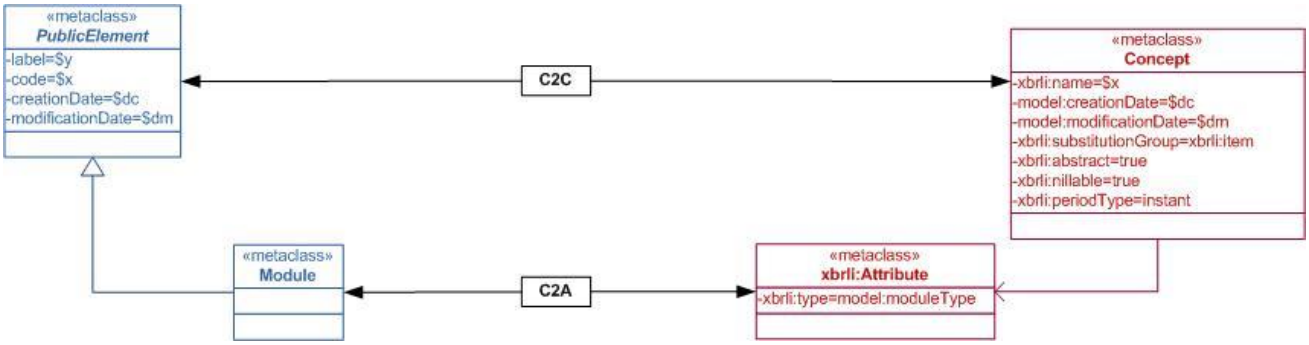

Figure 14 — Graph transformation from DPM to XBRL for modules

In addition to label linkbases, each module includes a presentation linkbase (*{module}-pre.xml*), where the relationships between modules and tables or table groups are expressed using *group-table* arcs (defined in the *model.xsd* schema file), whose source is the module element and whose target is a table or a table group element. The module schema also imports the formula linkbases and, optionally, the linkbases with the preconditions of filing indicators [EBA12, p. 19].

3.4.5 Validations

Validations are expressed using XBRL assertions. In order to handle the error margin caused by the imprecision of input data, assertions can make use of a set of functions implemented according to the Custom Functions Implementation specification. These functions use the same name as the ones defined in the XPath 2.0 Functions specifications, but are defined in the following namespace and placed in the following location:

Namespace:

— *http://www.eurofiling.info/xbrl/func/interval-arithmetics*

Official location:

— *http://www.eurofiling.info/eu/fr/xbrl/func/interval-arithmetics.xml*

Some example functions are:

— *iaf:numeric-equal(arg1, arg2): true* : if two values are equal, or are within the tolerance interval derived from its reported precision.

— *iaf:numeric-less-than(arg1, arg2)*: checks whether arg1 is less than arg2, considering their precision.

An entry point for these functions, and additional ones that could be provided in the future, is placed in the following location: *http://www.eurofiling.info/eu/fr/xbrl/func/functions.xsd*

Variables used are defined in no namespace; this way, there is a clear separation between variables and filing indicator parameters and the pivot-variable (see below). The naming convention for variables is lower camel case notation (*http://en.wikipedia.org/wiki/CamelCase*). Whenever an expression involves a certain number of facts used uniformly, sequence variables shall be used in order to improve the readability of the expression and the performance of the processor [EBA12, pp. 19].

3.4.5.1 Assertion sets

Validations are grouped into assertion sets that correspond to the tables to which they are to be applied. In the context of a table, not reported or nil numeric values will be assumed to be zero; consequently, fallback values shall be used in their corresponding assertion definitions. The link between an assertion set and the table (or tables) to which it applies, is represented using applies-to-table arcs from the assertion set to the resource that corresponds to the table. The URI of this arc is as follows: *http://www.eurofiling.info/xbrl/arcrole/applies-to-table*

If an assertion applies to multiple tables individually, or to multiple sets of tables, then it will be associated with different assertion sets.

Table 27 — Examples of assertions, assertion sets and tables they apply to

Example	Assertion example (textual description)	Assertion sets	Tables
1	$a > 0 (where $a represents data in table 1)	assertion set 1	table1
2	$a > 0 (where $a represents data in tables 1, 2 and 3)	assertion set 1	table1
		assertion set 2	table 2
		assertion set 3	table 3
3	$a = $b (where $a represents data in table 1 whereas $b represents data in table 2)	assertion set 1	table 1
			table 2
4	$a = $b (where in some cases, $a represents data in table 1 and $b data in table 2; in other cases, $a represents data in table 3 and $b represents data in table 4)	assertion set 1	table 1
			table 2
		assertion set 2	table 3
			table 4

Assertion sets resources might include the attributes *model:fromDate* and *model:toDate* to constrain the reference date where their associate assertions should be applied.

Assertions are identified by a unique code, so that it enables the identification of errors in a validation process with the corresponding definition. Assertions might include a description and custom error messages, as defined by business experts.

As suggested by the XBRL specification, assertion sets can be used as a mechanism to control the set of assertions to be evaluated in a validation process. Following this approach, an application that processes a certain filing would configure the processor to skip all those assertion sets that are linked to a table that is not reported.

However, at the time of writing of this document, the XBRL specifications do not provide a standard API to pass this information to XBRL processors, nor a standard way for the filer to indicate that only a subset of all the tables in an entry point is being submitted. To overcome this situation, a mechanism based on preconditions and filing indicators is provided [EBA12, pp. 20].

3.4.5.2 Preconditions and filing indicator parameters

Each value assertion defined is associated with a precondition of filing indicators. Assertions might have additional preconditions, as required by the logic of the assertion to be performed. But these additional preconditions do not depend on filing indicators. To avoid XBRL instance syntactic dependencies, rather than including directly an XPath expression, preconditions include a reference to a filing indicator parameter (no variableset-variable arc are required). This approach is quite convenient, as there is a new line of work in the Formula WG to define a subset of validations considered to be portable. This will enable the possibility of having XBRL processors performing validations on data in a database, rather than information on a single XML document. The default value of this parameter is an XPath expression to obtain the information from the filing indicators in the instance document.

This way, there is no need to provide externally a value to the processor (the value from the instance is used). The parameter is guaranteed to be only evaluated once (providing more chances for processors to perform optimizations). Precondition expressions are simpler, and it makes possible, for more advanced uses, to override this value at the application level (for instance, if the filing requirements of a credit institution are known, an application could override the values for filing indicator parameters rather than accepting the values provided by the filter). Filing indicators parameters are defined in the namespace of the filing indicators schema, and have a name according to the following convention:

```
t{table-code}
```

where table-code represents the code of the corresponding table. Thus, the definition of one of these parameters would look like this:

```
<variable:parameter name="find:t{table-code}"
select="//find:fIndicators/find:table = '{table-code}'" as="xs:boolean"
…/>
```

Each precondition is composed as a sequence of. or expressions that correspond to each set of tables, where the validation is to be applied. Each is composed of a sequence of and expressions on the tables involved:

```
$find:t{c1.1} and $find:t{c1.2} and …
or $find:t{2.1} and $find:t{2.2} and …
or …"
```

See examples provided below:

Table 28 — Example on the usage of filing indicators as preconditions for assertions

Expression	Explanation
$find:t1	Assertion applies only to table 1
$find:t1 and $find:t2	Assertion crosses information between tables 1 and 2
$find:t1 or $find:t2	Assertion applies to both table 1 and table 2, but considered in an individual way (there are no cross checks)
$find:t1 and $find:t2 or $find:t3 and $find:t4	Assertion performs cross-checks between information in table 1 and table 2 on the one hand. On the other hand, it cross-checks information between table 3 and 4.

[Source: EBA12, pp. 21]

3.4.5.3 Existence assertions

Existence assertions are not compatible with the precondition-based control schema proposed in the previous chapter. Existence assertions perform a test on the number of evaluations of a set of variables. Preconditions restrict the number of evaluations of the assertion, but not the evaluation of the assertion itself. Consequently, existence assertions are always evaluated (unless controlled using assertion sets); if a filing indicator precondition is added to an existence assertion, it will raise false errors. Preconditions are not a mechanism designed to control the evaluation of assertions, but to restrict the set of data where a validation is applied. In fact, there is a new line of work in the Formula WG to add preconditions at variable-set level. This way, preconditions could be used to control the evaluation of the assertions in the scope of a certain assertion set. However, this new and better approach is not expected to be ready in time for the release of EBA taxonomies. Nevertheless, the approach proposed in this document is ready to be adapted to the new standard in future releases of EBA taxonomies.

However, most existence assertions can be re-defined as value assertions using the following pattern:

- The assertion includes a "*pivot-variable*": a fact variable that matches data in the instance document known to always.be reported. The variable uses aspect cover filters to avoid any interference with the rest of variables. This variable is defined once as a sequence variable that matches the filing indicators.
- The rest of the variables in the original existence assertion are included with a fallback value (a value given to the variable if the fact is not found in the instance document).

The pivot-variable is defined in the namespace *http://www.eurofiling.info/xbrl/ext/pivot-variable*. The official location of this schema file is *http://www.eurofiling.info/eu/fr/xbrl/ext/pivotvariable.xsd*. The pivot variable is linked to the assertion using the name *{pvar:pivot}* in the pivotvariable namespace.

Although it is unlikely, there might be the case when validations cannot be defined using value assertions. For instance, checking that the number of rows (in a table with a variable number of rows) that verify a certain condition on several columns is greater than a certain number can be easily implemented using an existence assertion. The equivalent rule using a value assertion is extremely contrived and will have a very negative impact on the performance of the processor. If such a kind of rule were required, it should be implemented using an existence assertion. The *id* attribute of such assertions should follow a predefined naming convention to help applications not relying on validation sets to discard such evaluations [EBA12, pp. 22].

3.4.5.4 Notation

Assertions will be identified by a unique code, so that it enables the identification of errors in a validation process with the corresponding definition. It must be noted that an XBRL assertion might produce several evaluations covering different sets of data points. Assertions might include a description and custom error messages, as defined by business experts. Existence assertions shall only be used to detect errors in the case of data that should have been reported. Whenever possible, a value assertion shall be used instead of

existence assertion, as the former enables more comprehensive error messages and enables the usage of preconditions on filing indicators.

The files that define assertions and assertion sets are grouped into files depending on their scope. These files are placed in the “val” folder of the corresponding taxonomy, together with files to define preconditions and filters of common use, shared by different assertions in the taxonomy and parameters. These filters and preconditions should be independent of the assertion to which they apply, and thus, should not depend on the variables defined by specific assertions.

Table 29 — Examples of location and names of linkbase files containing value assertions and shared parameters, filters and preconditions

Resource description	File location
Assertions and assertion sets location that apply to a single table (example 1)	{taxonomy-loc}/val/val-{tab1}.xml
Assertions and assertion sets location that apply to multiple tables individually (example 2)	{taxonomy-loc}/val/val-{tab1}.{tab2}.xml
Assertions and assertions sets location that cross information in a set of tables (example 3)	{taxonomy-loc}/val/val-{tab1}_{tab2}.xml
Assertions and assertions sets location that cross information in a multiple sets of tables (example 4)	{taxonomy-loc}/val/val-{tab1}_{tab2}.{tab3}_{tab4}.xml
Parameters	{taxonomy-loc}/val/params.xml
Filters common to multiple assertions in the taxonomy	{taxonomy-loc}/val/filt.xml
Preconditions common to multiple assertions in the taxonomy	{taxonomy-loc}/val/prec.xml
Filing indicators parameters	{taxonomy-loc}/val/find-params.xml

Any of these linkbases can have its corresponding set of label linkbases, following the convention defined in this document. In the case of assertions, an additional set of linkbases might be included for error messages expressed in different languages:

```
{assertions-file}-err-{lang}.xml
```

or

```
{assertions-file}-err-{lang}-{country}.xml
```

Where *{assertions-file}* corresponds to the name of the file with the assertions whose error message are described, without the extension. These files will be included by the modules defined in the taxonomy [EBA12, pp. 23].

3.5 Architecture file structure

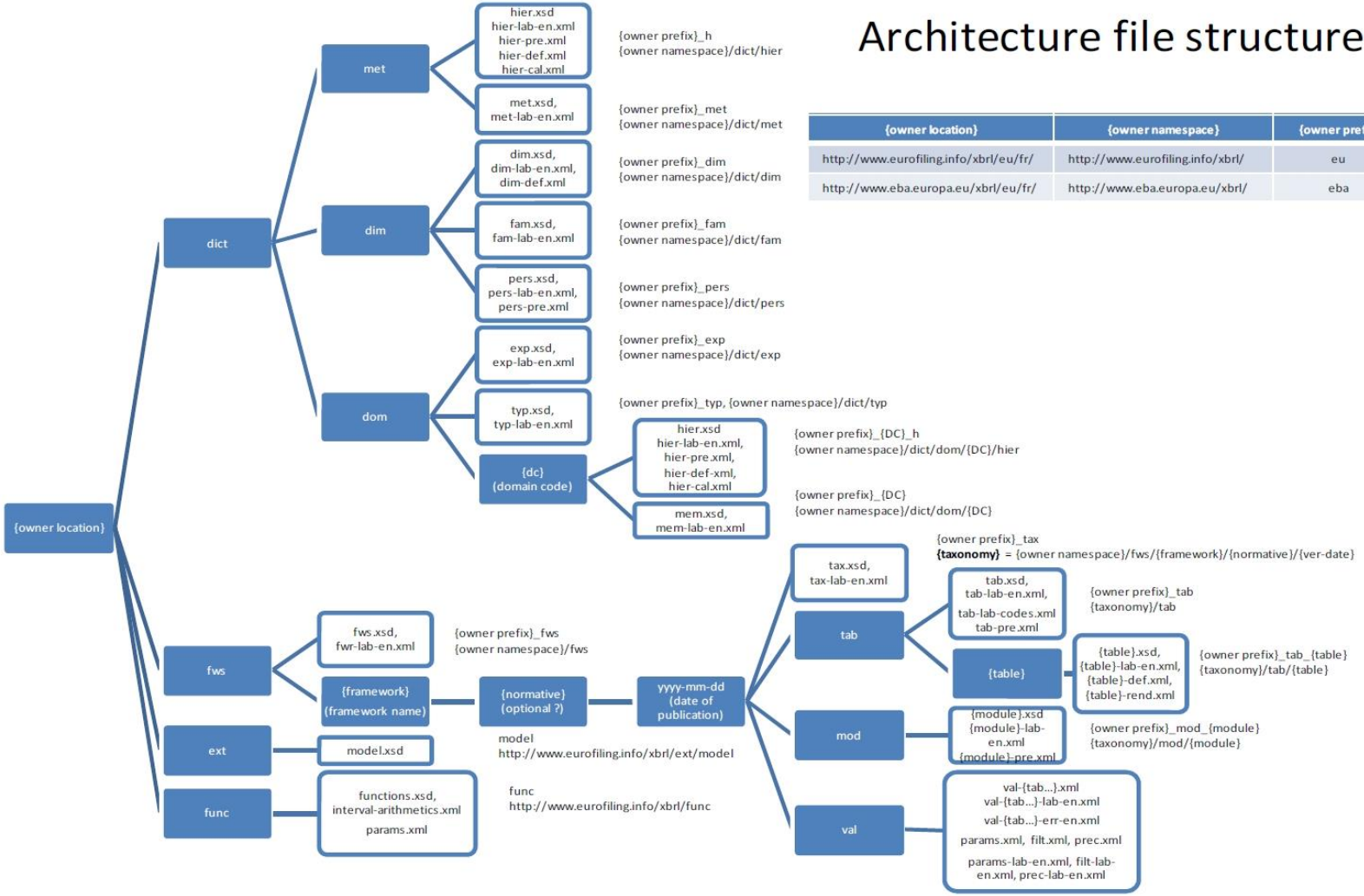

Figure 15 — Architecture file structure [Source: EBA12, p. 27]

3.6 Extending European taxonomies

3.6.1 Types of extensions

Extensions on the basis of EXTA taxonomies might include the following additions and changes:

- label translation (to a local/national language), including documentation (references to legal acts);
- addition of public elements:
  - dictionary elements:
    - metrics;
    - typed domains, explicit domains, explicit domain members and dimensions;
    - families of dimensions and perspectives;
    - hierarchies of metrics and hierarchies of explicit domain members.
  - frameworks and taxonomies;
  - table groups, tables, axes, their nodes and other properties;
  - modules;
  - validation rules.
- modification of information requirements, e.g., by not requiring facts that could be otherwise computed/derived from another reported facts [Ocho13, p. 1].

3.6.2 Extension use cases and patterns

The list below identifies the areas of potential extensions, as well as the patterns for addressing these extensions (identification of affected files and their content):

- **General settings** - Identification of the NSA which becomes the owner for extension concepts:
  1) Owner's root namespace (*{ons}*): NSAs official domain followed by /xbrl; example: http://www.nsa.gov.eu/xbrl/
  2) Owner's recommended prefix (*{opre}*): NSAs abbreviation, e.g., nsa
  3) Owner's root location (*{oloc}*): NSAs official domain, followed by the geographical area covered by the institution, plus fr/xbrl component, e.g.: http://www.nsa.gov.eu/eu/fr/xbrl/
- **Dictionary level extensions**

  1) Metrics

  i) Label translation of a metric defined by the European authority; label located in the standard extended link role (http://www.xbrl.org/2003/role/link) of *{oloc}*/dict/met/met-lab-*{lang}*-*{country}*.xml file, which @*href* attributes point to @*ids* of metrics defined by the European authority - *{lang}* corresponds to the ISO 639-1 code of the language. *{country}* is optional corresponds to the ISO 639-2 code of the region or country. Probably a schema file *{oloc}*/dict/met/met.xsd (with namespace *{ons}*/dict/met and prefix *{opre}*_met) would be

necessary as well to refer to the linkbase file (needs confirmation). The same situation would apply for label translation of all other public elements.

ii) Addition of a metric (and its properties, i.e., code, data type, period type, label, reference); e.g., NSA requires a placeholder for "Weighted amount for liquidity purposes", which is a monetary instant metric (hence code "mi1" and @id "nsa_mi1"); defined in *{oloc}*/dict/met folder in files: met.xsd, met-lab-*{lang}*-*{country}*.xml.

iii) Label translation of a metrics' hierarchy defined by the European authority; label located in standard generic extended link role of *{oloc}*/dict/met/hier-lab-*{lang}*-*{country}*.xml (hrefs point to ids of ELRs defined by the European authority).

iv) Inclusion of a new metric in hierarchies

I) Extension of an existing hierarchy defined by the European authority (using appropriate relationship type); defined in *{oloc}*/dict/met folder in the following files: hier-pre.xml, hier-def.xml and (optionally) hier-cal.xml files, in the ELRs of the European authority;

II) New hierarchy (of the NSA) in *{oloc}*/dict/met folder: definition of an ELR (according to the pattern {ons}/role/dict/met{sequential number}) in hier.xsd (with namespace *{ons}*/dict/met/hier and prefix *{opre}*_met_h), its label in hier-lab-*{lang}*-*{country}*.xml and relationships in hier-pre.xml, hier-def.xml and (optionally) hier-cal.xml.

7) Domains

i) Label translation of an explicit domain defined by the European authority: in standard extended link role of {oloc}/dict/dom/exp-lab-*{lang}*-*{country}*.xml (hrefs point to ids of explicit domains defined by the European authority).

ii) Label translation of a typed domain defined by the European authority: in standard generic extended link role of *{oloc}*/dict/dom/typ-lab-*{lang}*-*{country}*.xml (hrefs point to ids of typed domains defined by the European authority).

iii) Addition of an explicit domain: code and label in *{oloc}*/dict/dom/exp.xsd (with namespace *{ons}*/dict/exp and prefix *{opre}*_exp) and *{oloc}*/dict/dom/exp-lab-*{lang}*-*{country}*.xml files respectively.

iv) Addition of a typed domain: code and type in *{oloc}*/dict/dom/typ.xsd (with namespace *{ons}*/dict/typ and prefix *{opre}*_typ), label in {oloc}/dict/dom/exp-lab-*{lang}*-*{country}*.xml.

v) Label translation of an explicit domain member defined by the European authority: in standard extended link role of *{oloc}*/dict/dom/{European authority owner prefix}_{domain code}/mem-lab-*{lang}*-*{country}*.xml (domain code identical as the one applied by the European authority; hrefs point to ids of domain members defined by the European authority).

vi) Label translation of a domain members' hierarchy defined by the European authority: in standard extended link role of *{oloc}*/dict/dom/{European authority owner prefix}_*{domain code}*/hier-lab-*{lang}*-*{country}*.xml (domain code identical as the one applied by the European authority; hrefs point to ids of ELRs defined by the European authority).

vii) Addition of an explicit domain member of a domain defined by the European authority: declaration of an element (code) in *{oloc}*/dict/dom/{European authority owner prefix}_*{domain code}*/mem.xsd (with namespace *{ons}*/dict/dom/{European authority owner prefix}_*{domain code}* and prefix *{opre}_{European authority owner prefix}_{domain code}*), its label in *{oloc}*/dict/dom/{European authority owner prefix}_{domain code}/mem-lab-*{lang}*-*{country}*.xml and relationship in *{oloc}*/dict/dom/*{European authority owner prefix}_{domain*

code}/mem-def.xml (in standard extended link role, from explicit domain item to a domain member).

viii) Addition of an explicit domain member of a domain defined by the NSA : declaration of an element (code) in *{oloc}*/dict/dom/*{domain code}*/mem.xsd (with namespace *{ons}*/dict/dom/{domain code} and prefix *{opre}_{domain code}*), its label in *{oloc}*/dict/dom/*{domain code}*/mem-lab-*{lang}-{country}*.xml and relationship in *{oloc}*/dict/dom/*{domain code}*/mem-def.xml (in standard extended link role, from explicit domain item to a domain member) - In case of domains defined by NSAs, one member must become a default member.

ix) Inclusion of a new explicit domain member in hierarchies

I) Extending an existing hierarchy defined by the European authority (using appropriate relationship); defined in *{oloc}*/dict/dom/*{European authority owner prefix}_{domain code}* folder in the following files: hier-pre.xml, hier-def.xml and (optionally) hier-cal.xml files, in the ELRs of the European authority;

II) New hierarchy (of the NSA) in *{oloc}*/dict/dom/*{European authority owner prefix}_{domain code}* folder for a domain initially defined by the European authority or *{oloc}*/dict/dom/*_{domain code}* for an extension domain defined by the NSA: definition of an ELR in hier.xsd (with namespace *{ons}*/dict/dom/{European authority owner prefix}*_{domain code}*/hier and prefix *{opre}_{European authority owner prefix}_{domain code}*_h or a domain initially defined by the European authority or namespace *{ons}*/dict/dom/*{domain code}*/hier and prefix *{opre}_{domain code}*_h for an extension domain defined by the NSA), its label in hier-lab-*{lang}-{country}*.xml and relationships in hier-pre.xml, hier-def.xml and (optionally) hier-cal.xml.

8) Dimensions

i) Label translation of an explicit or typed dimension defined by the European authority: in standard extended link role of *{oloc}*/dict/dim/dim-lab-*{lang}-{country}*.xml (hrefs point to ids of explicit or typed dimensions defined by the European authority).

ii) Addition of an explicit dimension: declaration and code in *{oloc}*/dict/dim/dim.xsd (with namespace *{ons}*/dict/dim and prefix *{opre}*_dim), label in *{oloc}*/dict/dim/dim-lab-*{lang}-{country}*.xml, assignment to a domain (defined by the European authority or the NSA) and default member (defined by the European authority or the NSA) in *{ons}*/dict/dim/dim-def.xsd.

iii) Addition of a typed dimension: declaration, code and assignment to a typed domain (defined by the European authority or NSA) in *{oloc}*/dict/dim/dim.xsd (with namespace *{ons}*/dict/dim and prefix *{opre}*_dim), label in *{oloc}*/dict/dim/dim-lab-*{lang}-{country}*.xml.

— **Framework and taxonomy extensions**

1) Frameworks

i) Label translation of a framework defined by the European authority: in standard extended link role of *{oloc}*/fws/fws-lab-*{lang}-{country}*.xml (hrefs point to ids of frameworks defined by the European authority) - Probably a schema file *{ons}*/fws/fws.xsd would be necessary as well to refer the linkbase file.

iv) Addition of a framework: code and label in *{oloc}*/fws/fws.xsd (with namespace *{ons}*/fws and prefix *{opre}*_fws) and *{oloc}*/fws/fws-lab-*{lang}-{country}*.xml files, respectively.

9) Taxonomies

i) Label translation of taxonomy defined by the European authority: in standard extended link role of *{oloc}*/fws/{normative code} /{date of publication} /tax-lab-*{lang}-{country}*.xml (hrefs point to ids of taxonomies defined by the European authority).

ii) Addition of a taxonomy: code and label in *{oloc}*/fws/{normative code}/*{date of publication}*/tax.xsd (with namespace *{ons}*/fws/{normative code}/*{date of publication}* and prefix *{opre}*_tax) and *{oloc}*/fws//*{normative code}*/*{date of publication}*/tax-lab-*{lang}-{country}*.xml files, respectively.

10) Tables

i) Label translation of a table (title and headers, including codes, of rows/columns) defined by the European authority: in standard generic extended link role of *{oloc}*/fws/*{normative code}*/*{date of publication}*/tab/*{table code}* /*{table code}*-lab-*{lang}-{country}*.xml and *{oloc}*/fws/{normative code}/*{date of publication}*/tab/*{table code}*/*{table code}*-codes.xml (hrefs point to ids of table linkbase resources defined by the European authority).

ii) Extension by the NSA of a table defined initially by the European authority: extension of table linkbase for additional axes or their ordinates (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-rend.xml, *{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-lab-*{lang}-{country}*.xml, *{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-codes.xml), reference to all linkbases of a table and declaration of ELRs required by definition linkbase (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*.xsd with namespace *{ons}*/fws/*{normative code}/{date of publication}*/tab/*{table code}* and prefix *{opre}*_tab_*{table code}*) and definition linkbase itself (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-def.xml).

iii) Addition of a table by the NSA (for a taxonomy defined initially by the European authority or a new taxonomy of the NSA) in folder *{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}*, reference to all linkbases of a table and declaration of ELRs required by definition linkbase (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*.xsd with namespace *{ons}*/fws/*{normative code}/{date of publication}*/tab/*{table code}* and prefix *{opre}*_tab_*{table code}*)), definition linkbase itself (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-def.xml), table linkbase (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-rend.xml, *{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-lab-*{lang}-{country}*.xml, *{oloc}*/fws/*{normative code}/{date of publication}*/tab/*{table code}/{table code}*-codes.xml); inclusion of a relationship between the taxonomy and table in (*{oloc}*/fws/*{normative code}/{date of publication}*/tab/tab-pre.xml).

11) Table groups

i) Label translation of a table group defined by the European authority: in standard extended link role of *{oloc}*/fws/*{normative code}/{date of publication}*/tab/tab-lab-*{lang}-{country}*.xml (hrefs point to ids of table groups defined by the European authority).

ii) Extension by the NSA of a table group defined initially by the European authority - inclusion of more tables in a table group by defining appropriate relationships in *{oloc}*/fws/*{normative code}/{date of publication*}/tab/tab-pre.xml.

iii) Addition of a table group - definition of a table group in *{oloc}*/fws/*{normative code}/{date of publication}*/tab/tab.xsd (with namespace *{ons}*/fws/*{normative code}/{date of publication}*_tab and prefix *{opre}*_tab) and its label in *{oloc}*/fws/*{normative code}/{date of publication}*/tab/tab-lab-*{lang}-{country}*.xml and assignment of grouped individual tables (or other table groups) in *{oloc}*/fws/*{normative code}/{date of publication}*/tab/tab-pre.xml.

12) Module

i) Label translation of a module defined by the European authority: in standard extended link role of *{oloc}*/fws/*{normative code}/{date of publication}*/mod/*{module code}*-lab-*{lang}-{country}*.xml (hrefs point to ids of modules defined by the European authority).

ii) Extension by the NSA of a module defined initially by the European authority - inclusion of more table groups and tables by defining appropriate relationships in *{oloc}*/fws/*{normative code}/{date of publication}*/mod/*{module code}*-pre.xml, and defining imports in *{oloc}*/fws/*{normative code}/{date of publication}*/mod/*{module code}*.xsd (including references to Formula specification based generic linkbase files including appropriate validation rules).

iii) Addition of a module by the NSA - definition of a module in {oloc}/fws/{normative code}/{date of publication}/mod/{module code}.xsd and its label in {oloc}/fws/{normative code}/{date of publication}/mod/{module code}-lab-{lang}-{country}.xml, and assignment of groups or individual tables in {oloc}/fws/{normative code}/{date of publication}/mod/{module code}-pre.xml; schema file should also contain all necessary imports of assigned groups or individual tables, as well as references to Formula specification based generic linkbase files including appropriate validation rules.

13) Validation

i) Label/error message translation of a validation defined by the European authority ({oloc}/fws/{normative code}/{date of publication}/val/val-{table codes}-lab-{lang}-{country}.xml and {oloc}/fws/{normative code}/{date of publication}/val/val-{table codes}-err-{lang}-{country}.xml).

ii) Extension by the NSA of a validation defined initially by the European authority - variables and filters in appropriate files, following the naming pattern of files applied by the European authority (less or more tables may be involved in evaluations).

iii) Addition of a validation by the NSA [Ocho13, pp. 2].

Bibliography

[1] [EBA12] European Banking Authority (2012): Representation in XBRL of the Data Point Model. London. Online available at *http://eurofiling.info/finrepTaxonomy/EBA-DPM-XBRL-Mapping.pdf. last update on 2012-12-20*

[2] [XBRL] The eXtensible Business Reporting Language (XBRL), see *http://www.xbrl.org/TheStandard*

[3] [XML] The Extensible Markup Language (XML), see *http://www.w3.org/XML/*

[4] [Bund13]Deutsche Bundesbank (2013): XBRL Taxonomy Architecture. Status Draft. Frankfurt. Internal Working Draft. 2013-01-31.

[5] .[Ocho13] Ochocki, Bartosz (2013): Extensions of XBRL Taxonomies in European Supervision. Internal Working Draft. 2013-06-14

[6] [XBRL12] XBRL International Inc.(2012): XBRL Abstract Model 2.0. Public Working Draft dated 2012-06-06. Online available at *http://www.xbrl.org/Specification/abstractmodel-primary/PWD-2012-06-06/abstractmodel-primary-pwd-2012-06-06.html*

[7] .[Tae+] Taentzer, Gabriele; Ehrig, Karsten; Guerra, Esther; de Lara, Juan; Lengyel, Laszlo; Levendovszky, Tihamer; Prange, Ulrike; Varro, Daniel; Varro-Gyapay, Szilvia: Model Transformation by Graph Transformation: A Comparative Study. In Model Transformations in Practice Workshop 2005 (MTIP 2005), October 3rd, 2005. Technische Universit Berlin, Universidad Carlos III de Madrid, Universidad Autonoma de Madrid, Budapest University of Technology and Economics. Available at *http://www.mathematik.uni-marburg.de/~swt/Publikationen_Taentzer/TEG%2B05.pdf*