Documentation of the Bank of Israel XBRL taxonomy

Abstract

This document describes and explains the architecture of the public consultation version of the Bank of Israel (BoI) XBRL taxonomy. In particular, it explains the semantics and syntax used to express the information requirements of the data point model in XBRL format, and presents modularization of the taxonomy folder and files, naming conventions, and descriptive attributes used.

Version history

Version	Date	Change
1	2015-07-14	Initial version
2	2015-11-30	Modifications in parameters and preconditions definition

Table of contents

Abstract 1
Version history 1
Introduction 3
Relation to other standards and documents 3
Data model 3
XBRL specifications compliance 3
Supporting concepts 4
  Owner 4
  Model supporting schema 5
  Namespaces 5
Public elements 5
Dictionary of concepts 6
  Metrics 7
  Dimensions 8
  Domains 9
    Explicit domain members and hierarchies 10
Reporting requirements layer 11
  Frameworks 11
  Taxonomies 12
  Tables 12
  Modules 14
  Filing Indicators 15
  Validation rules 16
  Assertion sets 16
  Preconditions and filing indicator parameters 17
  Notation 18
Hypercubes 19
Architecture file structure 20

Introduction

This document presents and explains the architecture of the XBRL taxonomy of the Bank of Israel. The expected audience of this document are software developers and IT departments of commercial banks and other entities that are subject of supervision of the Bank of Israel. This document can be also useful for developers of software that produces or consumes instance documents following this taxonomy.

Relation to other standards and documents

Comprehension of the Extensible Business Reporting Language (XBRL) 2.1 Specification and various other XBRL Specifications such as XBRL Dimensions 1.0, XBRL Formula 1.0, Generic Link 1.0 and Table Linkbase 1.0 (Public Working Drafts) is required to understand the content of this document.

For modelling of data (in terms of methodology and format) as well as physical representation in XBRL syntax, the BoI followed the approaches applied for various deliverables of the Eurofiling project[1], especially Data Point Modelling methodology which is extensively described on the Eurofling webpage.

Data model

Prior to the development of an XBRL taxonomy (which is technical format used for data exchange), information requirements need to be identified by specifying reportable pieces of information. This is usually done in form of data models. Data models organize the data for communication purposes (e.g. between business and IT experts, or between various groups of business experts).

In the case of the BoI data model, the inputs for creation of the model are directives consisting of reporting requirements in tabular forms, additional information as well as validation rules.

These materials and underlying regulations are analyzed according to the Data Point Modelling methodology in order to create a Data Point Model format. There are two main deliverables of this process:

- Dictionary – defining properties (and their classifications/breakdowns) that can be used to describe each exchanged piece of information, and hierarchical relations between them.
- Annotated templates – Set of tables where each row/column/sheet is associated with a property or a set of properties defined in the dictionary

XBRL specifications compliance

Following the XBRL standard requirements, the BoI taxonomies, and any XBRL instance documents are compliant with the XBRL 2.1 specification as of December 31, 2003 with Errata Corrections up to January 25, 2012, and the Dimensions 1.0 specification as of September 18, 2006 with errata corrections up to January 25, 2012.

[1] Eurofiling is an European initiative that gathers supervisory authorities, commercial banks, software vendors and other stakeholders in order to share their experiences and innovations related with financial reporting supply chain. http://eurofiling.info

The business rules layer in the form of linkbase files is defined according to the XBRL Formula Specification 1.0 - 2009 – 2011 and supporting specifications (Registry – 2009-2011, Generic Links – June 22, 2009).

Rendering of tables is created according to the Public Working Draft of the Table Linkbase specification published on 17 May 2013.

For convenience for reviewers, the taxonomy files provided contain technical files defined by various XBRL specifications and registries. They are placed in the folder www.xbrl.org. In addition shared files from www.eurofiling.org are also included. The inclusion of these files simplifies the use of the supplied taxonomy files offline if required.

The Taxonomy files reference these files in their official locations. As such mappings will usually be required to be configured in most XBRL software to utilize the local version of these files, rather than those at the official locations, if so desired.

Supporting concepts

This chapter describes some concepts that are used in the Data Point Model taxonomies

Owner

The owner represents an institution that defines 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)* and *owner's prefix (opre)*: the owner namespace is a URI used to establish the namespace of the concepts defined by that owner. This URI is generally built by adding the "xbrl" particle to the internet domain of the institution that the owner represents plus an optional particle.
  The prefix is used as the basis to establish namespace prefixes in taxonomy files and for some short representations of the concepts. Namespace prefixes do not impose any constraints on instance files. Namespace prefixes are local to XML documents and XML elements, thus, instance files and taxonomy consumers should never presume any particular use of prefixes; XML documents consumption must be based on namespaces.

Owner	Internet domain	Namespace	Prefix
Bank of Israel	http://www.boi.org.il	http://www.boi.org.il/xbrl	boi
Eurofiling	http://www.eurofiling.info	http://www.eurofiling.info/xbrl	eu

- *Official location (oloc):* 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 particles to the internet domain of the institution: one that represents the geographical area covered by the institution, plus two fixed ones: "fr" (for financial reporting) and "xbrl":

Owner	Official location
Bank of Israel	
Eurofiling	http://www.eurofiling.info/eu/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 the owner of the concept. This enables the addition of labels to concepts imported from other taxonomies.

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.

Namespaces

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

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
variable	http://xbrl.org/2008/variable

Public elements

Public elements are concepts of the model 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 the creation date of the element (*model:creationDate)* and the date when it was last modified *(model:modificationDate)*.

Language specific information is represented using label resources (generic ones for concepts represented as XLink resources and standard ones for concepts represented as XBRL items). The default role (http://www.xbrl.org/2003/role/link) will be used for the extended links containing this information. The following roles must be used for 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

The labels of the concepts of a schema file are represented together in label linkbases by language, in the same folder as its corresponding schema file. The naming convention for these linkbases 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} corresponds to 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. More specifically, the codes given to the columns and rows of tables are represented using this mechanism. The name of this linkbase is as follows:

{main-file}-lab-codes.xml

The labels for these codes will be represented as resources with the following role, as defined in the model schema:

http://www.eurofiling.info/xbrl/role/rc-code

Extensions might use this mechanism to add their own application specific codifications using different roles.

Dictionary of concepts

The core concepts of the dictionary are metrics, dimensions, domains and domain members. All the concepts in the dictionary are public elements. 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 "fromDate" attribute is not included, then the concept is assumed to be current for any period prior to the "toDate" attribute. If the "toDate" attribute is not included, then the concept is assumed to be current for any period after the "fromDate" attribute. If neither "fromDate" nor "toDate" attributes are included, then the concept is assumed to be current for any period of time. The first versions of the dictionary won't include this attribute. 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 make easier 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. 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:

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 of domain	{oloc}/dict/dom/{dc}/mem.xsd	{ons}/dict/dom/{DC}	{opre}_{DC}

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. In the case of the dictionary of concepts of the BoI:

Dictionary concept	Official location	Target namespace	Prefix
Metrics	http://www.boi.org.il/fr/xbrl/dict/met/met.xsd	http://www.boi.org.il/xbrl/dict/met	boi_met
Dimensions	http://www.boi.org.il/fr/xbrl/dict/dim/dim.xsd	http://www.boi.org.il/xbrl/dict/dim	boi_dim
Explicit domains	http://www.boi.org.il/fr/xbrl/dict/dom/exp.xsd	http://www.boi.org.il/xbrl/dict/exp	boi_exp
Typed domains	http://www.boi.org.il/fr/xbrl/dict/dom/typ.xsd	http://www.boi.org.il/xbrl/dict/typ	boi_typ
Explicit domain members of domain (domain MC)	http://www.boi.org.il/fr/xbrl/dict/dom/MC/mem.xsd	http://www.boi.org.il/xbrl/dict/dom/MC	boi_MC

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 in the model they're also referred as Amount types (AT).

All the contexts in an instance document are expected to include an xbrli:period element with the same value: the reference period in the case of metrics of duration type, or the end of the reference period (for metrics of instant type). The variations from this reference period in certain data points are expressed with the combination of "comparison period" (RPC) and "reference period" (RPP) dimensions. This approach has been introduced in order to overcome the difficulty of defining time constraints for multiple periods in the table and definition linkbases.

Comparison period (RPC) dimension is used to enable entry of values for past periods like last year, equivalent quarter of last year etc. This is used to resemble current visualization of reporting requirements with different comparison periods identified on the columns of the tables.

Reference period (RPP) dimension is used to explain relative time reference for particular data point, e.g. "trailing year".

The local name of base items is composed of three parts:

- A letter that represents the data type in lower case (see data types table below)

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:ILS)
Percent	num:percentItemType	p	xbrli:pure
Decimal	xbrli:decimalItemType	r	xbrli:pure
Integer	xbrli:integerItemType	i	xbrli:pure
Date	xbrli:dateItemType	d	No unit
Boolean (true/false or 0/1)	xbrli:booleanItemType	b	No unit
Text	xbrli:stringItemType	s	No unit
Explicit domain	xbrli:qnameItemType	e	No unit
Typed domain	Domain corresponding data type, codification letter and reporting unit		

- A letter that represents the period type (i: instant, d: duration).
- A number that corresponds to the numeric code in the model (no zero padding or predetermined length).

In the case of domain based data types, an additional attribute (*model:domain*) is included to identify the qualified name of the domain (explicit or typed). 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:

<xs:element name="ei58" type="xbrli:QNameItemType" substitutionGroup="xbrli:item" id="boi_ei58" xbrli:periodType="instant" nillable="true" model:domain="boi_exp:BI" model:hierarchy="http://www.boi.org.il/xbrl/role/dict/dom/BI/1" />

In the above example, element has a restriction on reported values to the members of hierarchy 1 from the BI domain.

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

{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. Some examples follow:

Data type	Period type	Code	Name	Id	Prefix
Monetary	Instant	5	mi5	boi_mi5	boi_met
Date	Instant	177	di177	boi_di177	boi_met
Integer	Duration	102	id102	boi_id102	boi_met

Dimensions

Dimension items are represented in XBRL as XDT dimensions. The local name of each dimension corresponds to its code in the model: a short sequence of capital case letters (usually three, but it is not limited to three letters, where first two letters represents code of the domain used).

The id of the element (necessary for XLink locators) is composed like base items:

{opre}_{name}

Where {opre} represents the prefix of the base namespace of the owner of the dimension and {name} represents the name described above. Some examples follow:

Code	Name	Id	Prefix
MCC	MCC	boi_MCC	boi_dim
SEC	SEC	boi_SEC	boi_dim
RTT	RTT	boi_RTT	boi_dim

Dimension schemas include a reference to a definition linkbase whose file name is "dim-def.xml" and is placed in the same folder as the schema file. This linkbase includes the following information about explicit dimensions:

- Reference to the domain associated to the dimension by means of a dimension-domain relationship (with xbrldt:usable attribute equal to false).
- Reference to the default member of that dimension by means of a dimension-default relationship. Note that though the model defines default members at domain level, the dimensions XBRL specification establishes this relationship at dimension level. Thus, each dimension using a domain with a default member must include this relationship.

These relationships are defined in an extended role that is the standard one (http://www.xbrl.org/2003/role/link)

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")[2].

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 not limited to two letters). The id of the element (necessary for XLink locators) is composed like base items:

{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 follow:

Code	Element name	Type	Id	Namespace	Prefix
GA	GA	Explicit	boi_GA	http://www.boi.org.il/xbrl/dict/exp	boi_exp
TD	TD	Typed	boi_TD	http://www.boi.org.il/xbrl/dict/typ	boi_typ
TI	TI	Explicit	boi_TI	http://www.boi.org.il/xbrl/dict/exp	boi_exp

[2] 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. As some tools in the market do not support a single file with two different extended links, these items have been split into two different schemas

Though the namespace of explicit and typed domains is different, different local names should be used to avoid any confusion.

Explicit domain members and hierarchies

Explicit domain members are represented using XBRL abstract items of domain item type ("*domainItemType*" is defined in the non-numeric set of types of XII's type registry). The default domain member of a domain (usually the one with code 0) is marked with an attribute: *model:isDefaultMember = "true"*.

The local name of each explicit domain member corresponds to its numeric code in the model preceded by a lower case "x"[3]. The id of explicit domain members follows the general rule:

{opre}_{name}

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":

Domain code	Domain members schema	Namespace	Prefix
MC	http://www.boi.org.il/fr/xbrl/dict/dom/MC/mem.xsd	http://www.boi.org.il/xbrl/dict/dom/MC	boi_MC
SE	http://www.boi.org.il/fr/xbrl/dict/dom/SE/mem.xsd	http://www.boi.org.il/xbrl/dict/dom/SE	boi_SE

Hierarchies are represented using XBRL extended link roles whose role is built following this 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. The id of these roles is composed following the pattern:

{opre}_{code}

Domain code	Hierarchy code	Role	Id
MC	1	http://www.boi.org.il/xbrl/role/dict/dom/MC/1	boi_1
SE	13	http://www.boi.org.il/xbrl/role/dict/dom/SE/13	boi_13

The schema file that represents hierarchies is placed in the same folder as members and it is called "hier.xsd". In addition to labels, these schemas include 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.
- A definition linkbase (hier-def.xml), which enables the inclusion of the members of a hierarchy in dimensional combinations using domain-member relationships.

[3] Local names are XML schema tokens and thus, are not allowed to start with a numeric character

- 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.

These arc roles are defined in the model schema:

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

These arcs (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 schema that supports the model. The root member of the definition and presentation relationship networks is the domain item defined in the schema.

Reporting requirements layer

Frameworks, taxonomies, tables, modules and other concepts constitute the layer of the model where actual reporting requirements are specified with the support of 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 concept represented.

Frameworks

Frameworks are public elements represented using XBRL abstract items of framework type (“*model:frameworkType*”) in the schema file “fws.xsd”. The local name of each framework element corresponds to its code in the model and its id follows the general pattern.

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}

BoI is modularized in a way that every of the reporting directives constitutes separate framework container, e.g.

Schema property	Value
Official location	http://www.boi.org.il/fr/xbrl/fws/fws.xsd
Target namespace	http://www.boi.org.il/xbrl/fws
Target namespace prefix	boi_fws
Element local name	d806; d87; d810D; d98 …
Element id	boi_d806; boi_d87; boi_d810D; boi_d98 …

Each framework has a folder where the files of its taxonomies are placed. This folder has the name of its code in the model:

Description	Framework folder
Directive 98	http://www.boi.org.il/fr/xbrl/fws/d98
Directive 810D	http://www.boi.org.il/fr/xbrl/fws/d810d

Taxonomies

Taxonomies are public elements represented using XBRL abstract items of 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 and another subfolder with the date of its version, using the ISO 8601 codification.

Thus, the file "tax.xsd" includes a single element. Its local name corresponds to its code in the model and its id uses the general pattern:

Schema property	Value
Official location	{oloc}/fws/{framework}/{normative}/{pub-date}/tax.xsd
Target namespace	{ons}/fws/{framework}/{normative}/{pub-date}
Target namespace prefix	{opre}_tax
Element local name	{taxonomy}
Element id	{opre}_{taxonomy}

Since every directive is stored in a separate framework, there will be only one normative code per framework, however, there can be multiple versions basing on a publication date only.

Description	Version	Taxonomy folder
Directive 98	1.0	http://www.boi.org.il/fr/xbrl/fws/d98/d98/2015-07-15
Directive 98	2.0	http://www.boi.org.il/fr/xbrl/fws/d98/d98/2015-12-31
Directive 814	1.0	http://www.boi.org.il/fr/xbrl/fws/d814/d814/2015-07-15
Directive 814	2.0	http://www.boi.org.il/fr/xbrl/fws/d814/d814/2015-09-30

The folder of a taxonomy includes three folders for tables (*tab*), modules (*mod*) and validations (*val*).

Tables

The table folder includes a schema file (tab.xsd), 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 definition of table groups (if any), which are represented using XBRL abstract items of table group type

("*model:tableGroupType*"). Its name is composed by adding the prefix "tg" to the code in the model. The linkbase with the hierarchy of tables is not referenced in schema; otherwise, all the modules defined in a taxonomy would include indirect links to all the tables in the taxonomy.

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}
Element id	{opre}_{local-name}

Arcs with role "group-table" are used to establish the link between a table group and other table groups or tables in the presentation linkbase. This arc role is defined in the schema that supports the model.

Table groups are used to link numerous tables resulting from normalization of templates or if an original templates is composed by two or more physical tables. In other words, table groups represent those templates that consist of more than one table. If a business template was normalized (e.g. table 98-1), then resulting sub-tables contain additional suffix to its names (98-1.a; 98-1.b; 98-1.c; …). General approach of application these suffixes is that part .a is the beginning of a table (row 1 and/or column 1) of a normalized table, and consequent parts are first moving horizontally, and later vertically to the last columns and rows.

One of the other results of normalization of templates is removing redundant rows and/or columns. As a consequence, it can happen that in the sub-table of a normalized table, some rows or columns can be skipped.

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:

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

In addition to label linkbases, this schema includes a table linkbase ({table}-rend.xml) and a definition linkbase ({table}-def.xml).

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

{ons}/role/fws/{framework}/{normative}/{pub-date}/tab/{table}

In this linkbase, the different components of tables are represented using resources. The "id" of these resources is based on the code of the model plus a prefix to obtain a unique code in the context of the linkbase file:

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}

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

Each extended link role contains a set of primary items and a single hypercube[4]. In 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 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.

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}/{pub-date}/tab/{table}/1

{ons}/role/fws/{framework}/{normative}/{pub-date}/tab/{table}/2

The label linkbase file for a table contains labels for Table Linkbase nodes. In addition to the standard label, a table:table node, also contains a documentation and filing indicator label which defines a code to be used on filing indicators (see next section of this document).

The link between table groups and individual tables is established in the tab-pre.xml linkbase file as well as in linkbase files of modules (as described below).

Modules

Modules are represented using XBRL abstract items of module type ("*model:moduleType*"). Each module is stored in a different schema file whose name module file is the same as the code of the module in the model plus the extension ".xsd". These schema files imports the schemas of all the tables imported by that module:

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

[4] The model schema includes a hypercube element to be used. There is no need to define hypercube elements in each table or taxonomy.

Element local name	mod_{module}
Element id	{opre}_mod_{module}

In addition to label linkbases, each module includes a presentation linkbase ("{module}-pre.xml") where the relationship between modules and tables / table groups is expressed using group-table arcs whose source is the module element and target is the table / group of tables element. Furthermore, table groups link to individual tables via a group-table relation.

The module schema also imports the formula linkbases and the linkbases with the preconditions on filing indicators.

Modules in the BoI taxonomy serve as entry points, defining the potential tables in each individual instance file that can be reported. Usually, there is one module (entry point) per directive gathering all templates, but there can be cases (e.g. Directive 865) where two or more different reporting scenarios gathering different set of templates are applicable (Annually and Quarterly reporting).

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 to:

- provide hints to applications using the taxonomy, when processing instance files, on which templates are included in the filing and, for example, shall be displayed to users,
- trigger execution of business rules (XBRL assertions) to be run on a filing to check its correctness depending on the reported scope of data.

In technical terms, filing indicators are facts included as part of an instance document where the filer provides information about the reported templates (within the scope defined by a module that the filing is defined against, see previous section on Modules).

The elements and attributes used to communicate filing information 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. Throughout this document, the prefix "*find*" will be used to make reference to this schema namespace.

Each reported template is represented as an instance fact of the item *find:filingIndicator* under the *find:fIndicators* tuple element. If there is no filing indicator for a template included in a module, it is assumed that a filing contains no information on this template. In some rare cases however, it may be necessary that filers explicitly identify unreported templates, usually with a reason explaining this situation/choice. To cater for this situation, a find:filingIndicator fact relating to the template identification can have a *find:filed* attribute set to boolean "false".

The following instance represents a filing with information about template with code 865-1 and no information (explicitly stated) on template 865-21:

```
<find:fIndicators>
        <find:filingIndicator contextRef="context">865-1</find:filingIndicator>
```

<find:filingIndicator contextRef="context" filed="false">865-21</find:filingIndicator>
</find:fIndicators>

Contexts to which facts representing *find:filingIndicator* element refer must identify the reporting entity and use the end date of the reporting period as the instant date.

Identification of templates on *find:filingIndicator* facts is made using codes. These codes are represented as label resources with the following role, as defined in the model schema:

http://www.eurofiling.info/xbrl/role/filing-indicator-code

These code labels are applied to a table:table resource.

Validation rules

In the current version of the BoI taxonomy, validation checks are based on the legacy business rules used in the ZPTM system. Since previously, no unique identifiers for validations were used, they were generated and applied in the BoI taxonomy for the first time.

Current set of validation rules do not apply generalization of filters, meaning that single assertion should result with a single evaluation (except situations with comparison period, where single formula can be applied to one or more comparison periods)

Assertion sets

Validations are grouped into assertion sets that correspond to the tables 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 are used in their corresponding assertion definitions.

The link between an assertion set and the table (or tables19) 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 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 to different assertion sets

Assertion example (textual description)	Assertion sets	Tables
$a > 0 (where $a represents data in table 1)	assertion set 1	table 1
$a > 0 (where $a represents data in tables 1, 2 and 3)	assertion set 1	table 1
	assertion set 2	table 2
	assertion set 3	table 3
$a = $b (where $a represents data in table 1 whereas $b represents data in table 2)	assertion set 4	table 1 table 2
$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 4	table 1 table 2
	assertion set 5	table 3 table 4

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

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 processing a certain filing would configure the processor to skip all those assertion sets that are linked to a table that is not reported.

However, currently, the XBRL specifications do not provide a standard API to pass this information to XBRL processors, neither 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.

Preconditions and filing indicator parameters

Each value assertion defined is associated to a precondition[5] 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). 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 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).

There is a filing indicators parameter defined for each table defined in the framework. These 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:fIndicator = '{template-code}'" as="xs:boolean" …/>

Where 'template-code' represents the code of the template

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 or expression is composed of a sequence of and expressions on the tables involved:

"$find:t{t98-21} and $find:t{t98-22} and …"

Some examples:

[5] Assertions might have additional preconditions as required by the logic of the assertion to be tested. But these additional preconditions do not depend on filing indicators.

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.

Notation

Assertions are being identified by a unique code, to enable 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. Additionally, there are two levels of severity of an error that evaluation may raise:

- fatal
- warning

Information about severity of particular business rule is stored within the name of the file where suffix "*_f*" representing fatal error or "*_w*" representing warning is added.

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:

Resource description	File location
Assertions file	{taxonomy-loc}/val/vr-v{ID}_{f/w}.xml
Assertion sets location that apply to a single table	{taxonomy-loc}/val/aset-{tab1}.xml
Assertion sets location that apply to multiple tables individually	{taxonomy-loc}/val/aset-{tab1}.xml {taxonomy-loc}/val/aset-{tab2}.xml
Assertion sets location that cross information in a set of tables	{taxonomy-loc}/val/aset-{tab1}_{tab2}.xml
Assertion sets that cross information in a multiple sets of tables	{taxonomy-loc}/val/aset-{tab1}_{tab2}.xml {taxonomy-loc}/val/aset-{tab3}_{tab4}.xml
Preconditions on filing indicators plus variable-set-precondition arcs	{taxonomy-loc}/mod/{module}-find-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 cases of assertions, an additional set of linkbases might be included for error messages expressed in different languages:

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

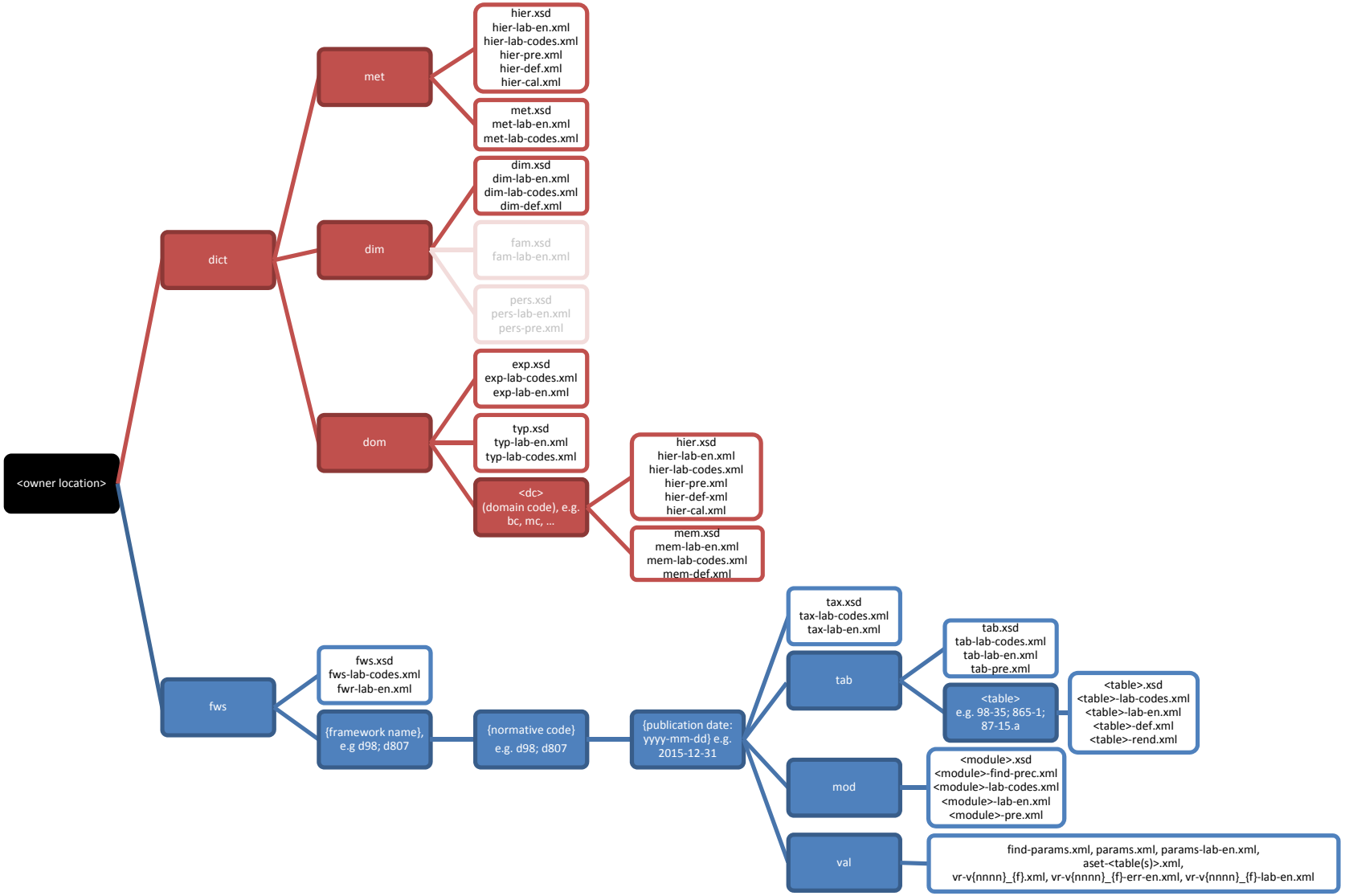
Where {assertions-file} corresponds to the name of the file with the assertions whose error message are described, without the extension.

Currently, for every assertion there is an error message containing information about table and cells involved in given validation check.

These files will be included by the modules defined in the taxonomy.

Hypercubes

It is important to remark that the XBRL hypercubes in the taxonomy are validation artefacts (essentially just indicating grey cells) and should not be used by external systems for the automatic creation of database structures. The hypercubes in the taxonomy are generated automatically by an algorithm, and do not obey to any kind of business criteria. These hypercubes might be dramatically modified with any future change to the reported information in a table, with the only consideration being the reduction of the final set of hypercubes and performing more efficiently with XBRL market tools.



Architecture file structure