RECEX

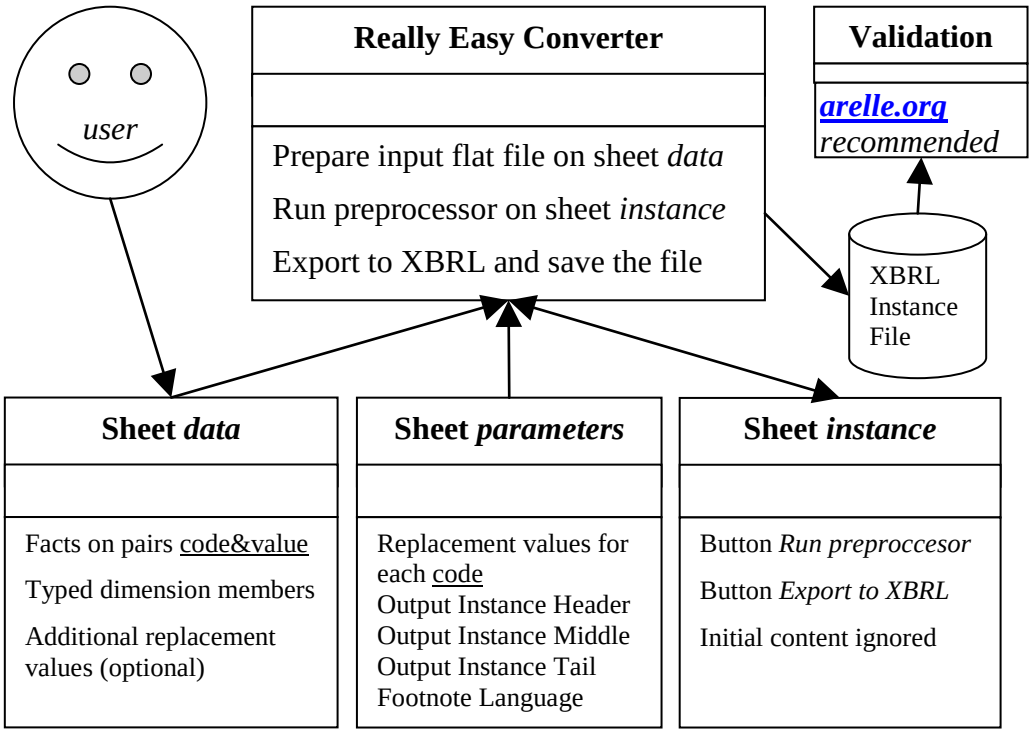
Really Easy Converter Excel to XBRL

Version: 2011-12-05

Authors: Ignacio Boixo, Javi Mora.
Contributor: Herm Fischer
Acknowledgment: Allyson Ugarte, Colm Ó hÁonghusa, Ignacio Amelivia.
In collaboration with XBRL Spain

GOAL:
Proof of Concept, generic converter from flat files to XBRL instance documents.

Description:
This is a Excel Visual Basic program, creating XBRL instance documents in two steps: Preprocessing data and generating XBRL file.



The preprocessor prepares the sheet *instance* using the flat file (composed of pairs of code-value) in sheet *data* and the replacement commands defined in the sheets *parameters* and (optionally) *data.*

Only *parameters* sheet must be customized for each different XBRL taxonomy. *Parameters* sheet contains a "distillation" of the way in which the XBRL instance is created for the specific XBRL taxonomy. An easy procedure is take an XBRL instance provided usually as example, and then create the *parameters* sheet using that example as pattern. Usually, it is even not need open the XBRL taxonomy: with the information in the XBRL instance provided as example is usually enough for testing purposes.

The converter do not validate at all. In fact, the converter do not generate any error message. To validate the XBRL instance generated it is recommended the Open Source XBRL tool Arelle, available in arelle.org for free in different idioms.

The source core of the converter is generic and taxonomy independent. The source code is designed as Proof of Concept, and therefore it do no capture all the richness and possibilities of the XBRL Language. But with only 500 lines of code, is more than enough for training purposes J

Excel Visual Basic has been selected to avoid security problems in enterprise environments. The empirical evidence shows than Visual Basic is near the only easy computer language able to be executed in personal computers without complex security restrictions.

Excel is also very flexible preparing the input flat file. A number of solutions would be easily customized.

For production environments, please recode in a production language (C, Java, .Net and so no). As a courtesy, if your development (using this deliverable or not) is Open Source, please notify to info@openfiling.info for further dissemination. Thanks.

STEP 1: RUN PREPROCESOR

The replacement process would be used to transform each row of a flat file of the type Code-Fact into a row with all the information required for a Fact in an XBRL instance. The information required for this transformation is usually fixed, and therefore would be parameterized. In this way, the user only needs obtain the Facts corresponding to the Codes. The preprocessor transforms and completes the required information with the parameters.

How preprocessor works:

In the sheets *data* and *parameters*, any row starting with "/* " is a comment and it is ignored.

In the sheets *data* and *parameters*, any row starting with "/**" is a command to be processed.

Each row of sheet *data* (except if starting with "/*") is copied to a consecutive row in the sheet *instance*.

With the end of sheet *data*, an additional row with empty cells is added to sheet *instance*, as end of file mark. Therefore, the subsequent rows in *instance* sheet are ignored.

Each row in the *instance* sheet is explored as target row for replace.

At the end of the process, each row in the *instance* sheet should have all the required information to generate the fact in the XBRL instance.

/Replace**

The command "/**Replace" inserts predefined values in positional cells of the target row (sheet *data*).
The syntax is "/**Replace,"<keyword>{","<value>|","}

If the value of a cell in the target row (except RowID, FootNote and Fact cells) match with a keyword in a command "/**Replace,<keyword>", the command is triggered, and then:

If the keyword starts with the symbol “&”, it is a single cell replacement:

- The matching target row cell is made empty
- The value of the first non empty cell in the command row “/**Replace” (except the cells RowID and FootNote) is copied from the command row cell to the target row cell with the keyword as value (the cell triggering the command).
- The target row cells are re-explored again looking for further matches in the target row cells (recursive process).

If the keyword do NOT starts with the symbol “&”, it is a full row replacement.

- The matching target row cell is made empty
- For each not empty cell in the command row (except the cells RowID and FootNote), if the same cell is empty in the target row, then the value is copied from the command row cell to the target row cell.
- The target row cells are re-explored again looking for further matches in the target row cells (recursive process).

In case of several commands "/**Replace" with the same keyword, it will be used the immediately prior in the sheet *data* or, if not found, the last one in sheet *parameters* or, if not found, the first one in sheet *data*.

Other commands:

Command "/**OutputHeader,"<text> inserts the content of <text> in the output file, tipically the header the generated XBRL Instance document.
Syntax: "/**OutputHeader,"<text>

Command "/**OutputMiddle,"<text> inserts the content of <text> in the output file, tipically the between the contexts and the facts of the generated XBRL Instance document.
Syntax: "/**OutputMiddle,"<text>

Command "/**OutputTail,"<text> inserts the content of <text> in the output file, tipically at the end of the generated XBRL Instance document.
Syntax: "/**OutputTail,"<text>

Command "/**FootnoteLang,"<lang> uses <lang> as the language to be inserted in the footnotes (typically 2 characters), in the field xml:lang="en". The default (if not specified) is "en". Only the first occurrence of "/**FootnoteLang" will be considered. Therefore, all the footnotes will be in the same language (this is a restriction of this program).
Syntax: "/**FootnoteLang,"<lang>

Command "/**Null" It may be used in any cell, and forces a null (empty) value in the cell it appears in the
Rationale: when the target cell is not empty, it is not replaced. /**Null is used to force that a specified target cell, that must be empty in the sheet *data*, cannot be replaced.

STEP 2: EXPORT TO XBRL.

A file containing an XBRL instance document will be generated.

Each row in the *instance* sheet will generate a fact in the XBRL instance file.

Each row in the *instance* sheet should contain all the information required to generate a fact in an XBRL instance.

The columns contain:

- RowID: Optional. To be included as "id=*RowID*" in the Fact.
- Footnote: Optional. To be included in the Fact as an XBRL footnote.
- PrimaryItem: Mandatory. To be included in the Fact.
- Decimals: Optional. To be included in the Fact.
- UnitRef: Optional. To be included in the Fact.
- Fact: Mandatory. To be included as the value of the Fact.
- Identifier: Mandatory. To be included in the context of the Fact.
- IdentifierSchema: Mandatory. To be included in the context of the Fact.
- PeriodInitDuration: Optional. For period of type "duration" as "startDate". If omitted, then the period is of type "instant". To be included in the context of the Fact.
- PeriodEndInstant: Mandatory. To be used as "endDate" (type "duration") or as value (type "instant"). To be included in the context of the Fact.

- A number of columns has been reserved for optional Dimensions or Tuples

The converter generates not repeated contexts

Dimension/Tuples
A row may have Dimensions or Tuples, but not both simultaneously.
Dimensions and Tuples information is left aligned, occupying a predefined set of consecutive columns, without empty cells in the middle.
Define always Dimensions and Tuples in the same order.

Tuples:
Each tuple uses two columns, the command "/**Tuple" and the name of the Tuple.
A PrimaryItem that can be repeated inside a Tuple when having the command "/**Unbounded" in the last column

A tuple is closed when the next row:

(1) Has NOT the same pair "/**Tuple,<tuplename>" in the same cells.
i.e.
PrimaryItemA,,,,/**Tuple,TupleZ
PrimaryItemB,,,,/**Tuple,TupleY
(Tuple Z is closed and then TupleY is opened)

(2) Has the same list of pairs "/**Tuple,<tuplename>" in the same cells, and a fact with the same PrimaryItem has already generated.
i.e.
PrimaryItemA,,,,/**Tuple,TupleZ
PrimaryItemB,,,,/**Tuple,TupleZ (Comment: A & B are both in TupleZ)
PrimaryItemA,,,,/**Tuple,TupleZ (Comment: A is used again. TupleZ is closed and reopened, for a new TupleZ occurence)

Exception: two consecutive PrimaryItems ending with the command "/**Unbounded"
i.e.
PrimaryItemA,,,,/**Tuple,TupleZ
PrimaryItemC,,,,/**Tuple,TupleZ,/**Unbounded
PrimaryItemC,,,,/**Tuple,TupleZ,/**Unbounded (Comment: Two occurrences of C will be generated in the same TupleZ occurrence)

Dimensions
Each *explicit Dimension* uses two columns: Dimension and Dimension Member

Each *typed Dimension* uses four columns: Dimension, Dimension Member, Command "/**Typed" and the Dimension Member Type

Scenario/segment

By default, the dimensions will be placed as "<scenario>"

The command "/**Scenario,/**Segment" forces that all the dimensions (if any) at the left of "/**Scenario" are to be placed as "<scenario>", and all dimensions (if any) at the right of "/**Segment" are to be placed as ""

The command "/**Scenario,/**Segment" is double and therefore must be placed in two consecutive cells. Only the first occurrence of "/**Scenario,/**Segment" is processed; the following occurrences are ignored.

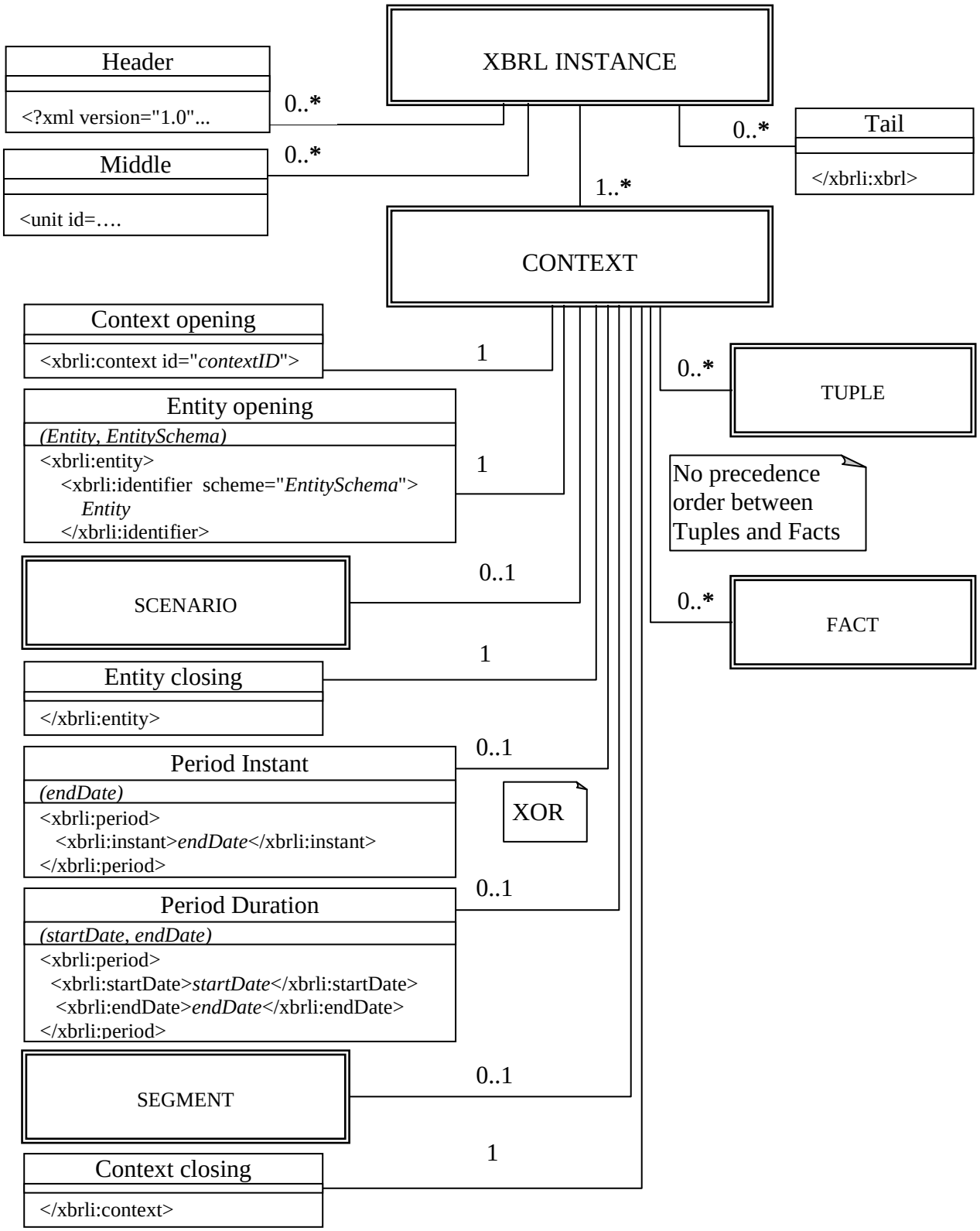
Bibliography:

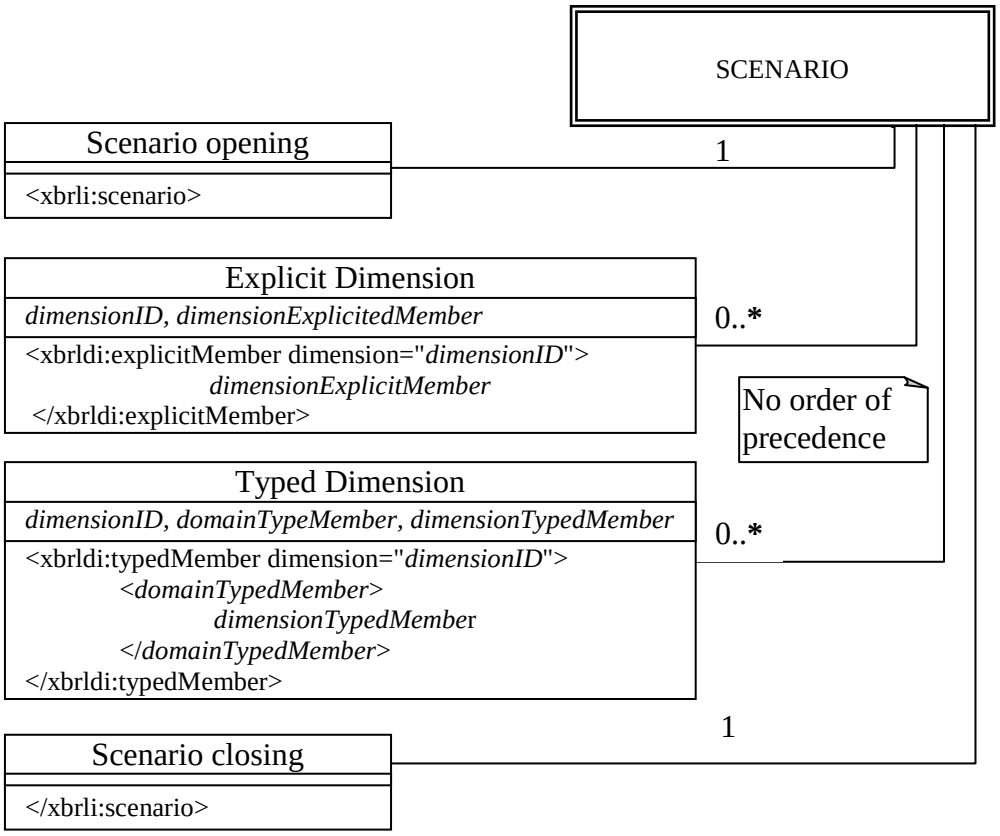
Extensible Business Reporting Language (XBRL) 2.1 (specification)
http://www.xbrl.org/Specification/XBRL-RECOMMENDATION-2003-12-31+Corrected-Errata-2008-07-02.htm

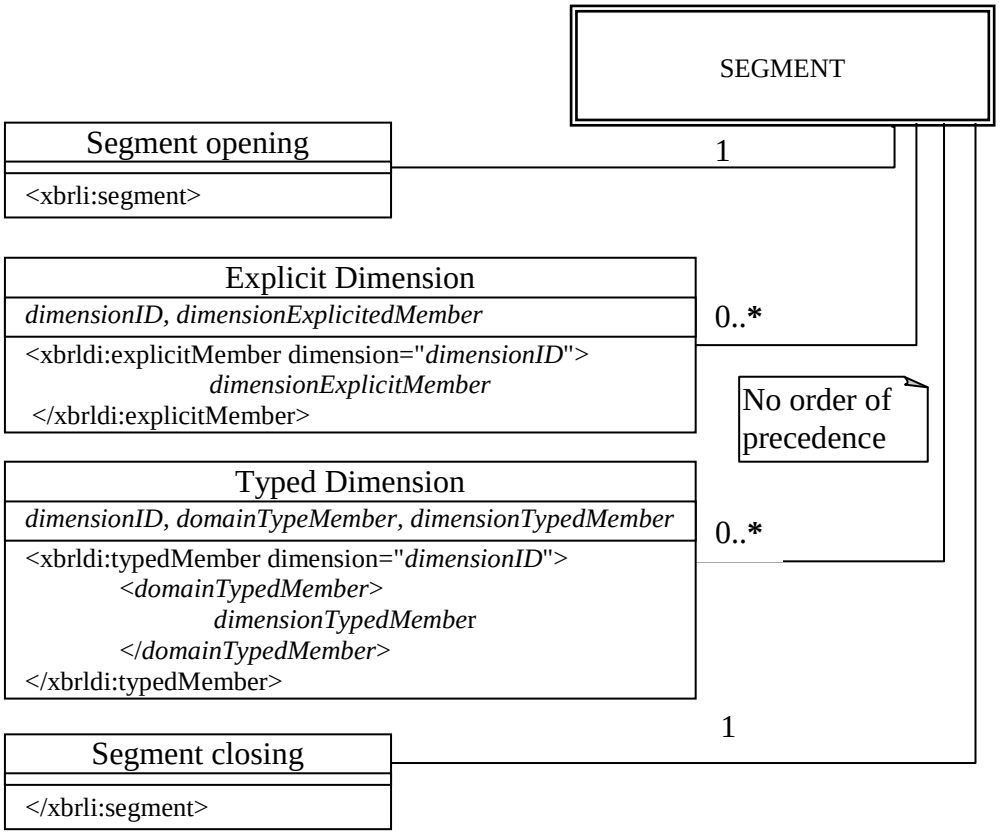
XBRL Dimensions 1.0 (specification)
http://www.xbrl.org/Specification/XDT-REC-2006-09-18.htm

XBRL Abstract Model draft
http://xbrl.org/Specification/abstractmodel-primary/PWD-2011-10-19/abstractmodel-primary-PWD-2011-10-19.html

Annex: UML description

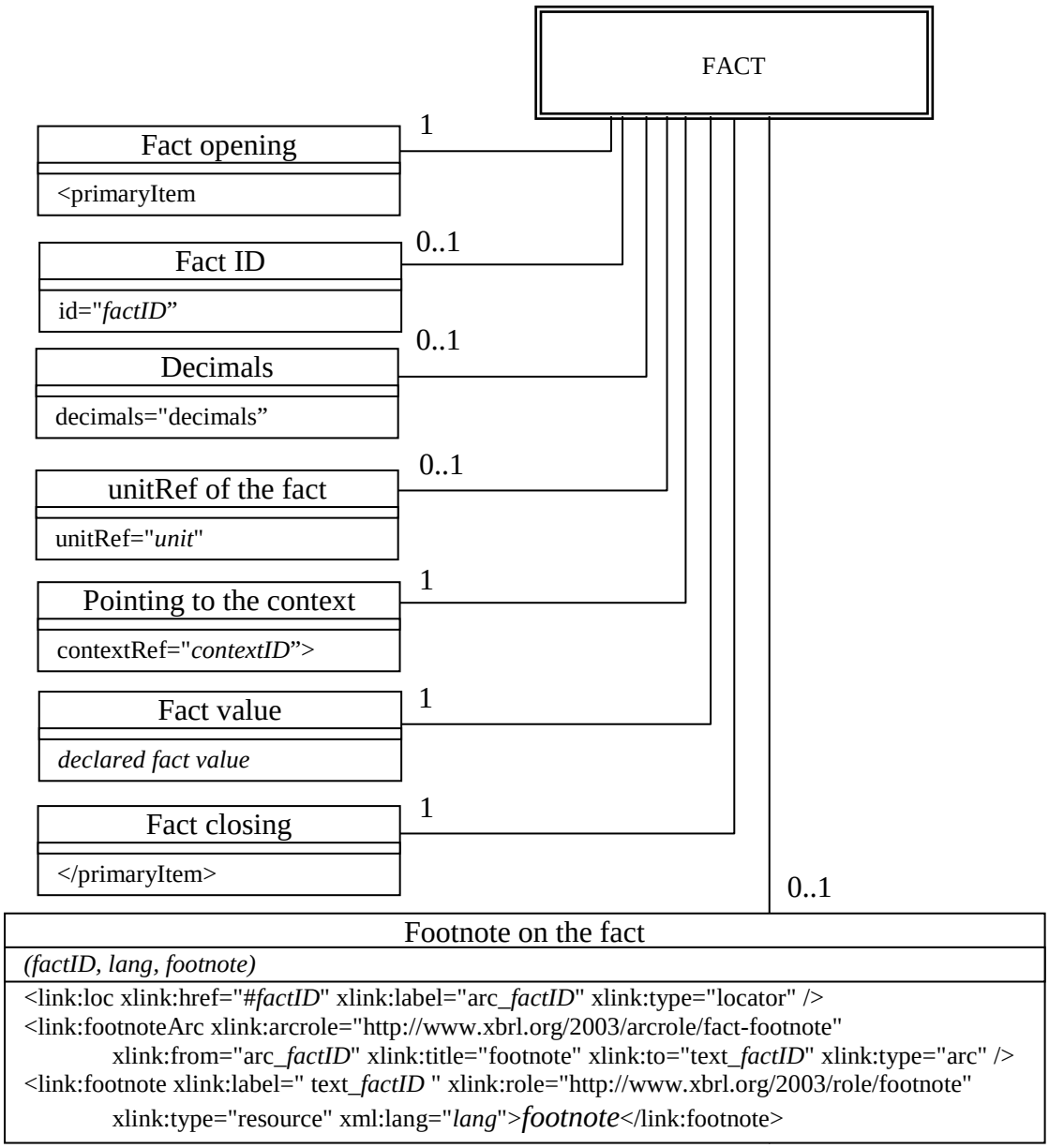


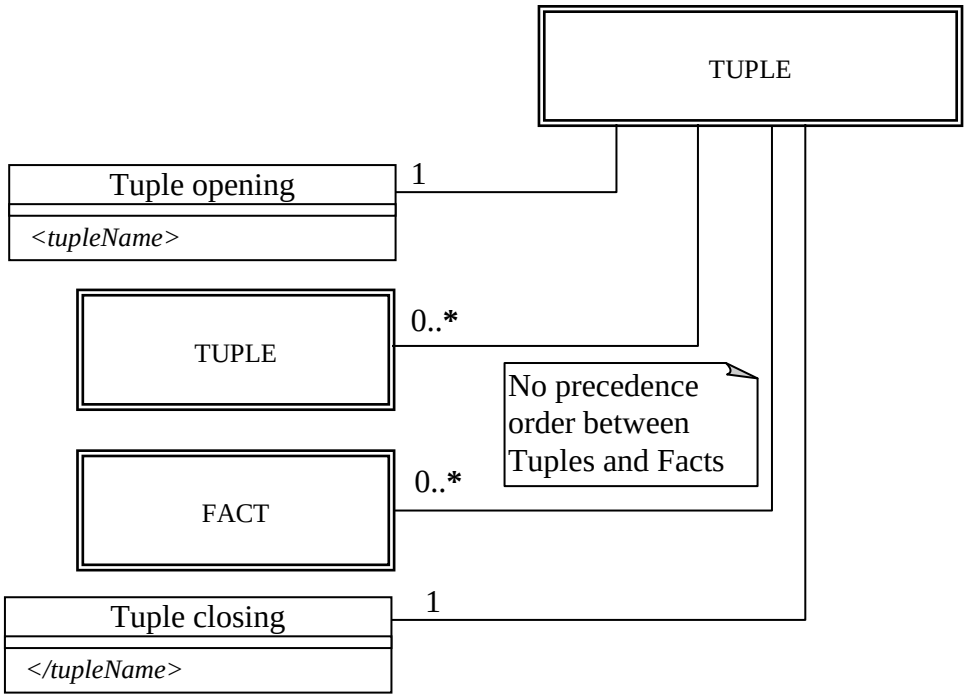



SEGMENT
Segment opening

1
Explicit Dimension
dimensionID, dimensionExplicitedMember
<xbrldi:explicitMember dimension="dimensionID">
dimensionExplicitMember
</xbrldi:explicitMember>
0..*
No order of precedence
Typed Dimension
dimensionID, domainTypeMember, dimensionTypedMember
<xbrldi:typedMember dimension="dimensionID">
<domainTypedMember>
dimensionTypedMember
</domainTypedMember>
</xbrldi:typedMember>
0..*
1
Segment closing

1


FACT
Fact opening
<primaryItem
1
Fact ID
id="factID"
0..1
Decimals
decimals="decimals"
0..1
unitRef of the fact
unitRef="unit"
0..1
Pointing to the context
contextRef="contextID">
1
Fact value
declared fact value
1
Fact closing
</primaryItem>
1
0..1
Footnote on the fact
(factID, lang, footnote)
<link:loc xlink:href="#factID" xlink:label="arc_factID" xlink:type="locator" />
<link:footnoteArc xlink:arcrole="http://www.xbrl.org/2003/arcrole/fact-footnote"
xlink:from="arc_factID" xlink:title="footnote" xlink:to="text_factID" xlink:type="arc" />
<link:footnote xlink:label=" text_factID " xlink:role="http://www.xbrl.org/2003/role/footnote"
xlink:type="resource" xml:lang="lang">footnote</link:footnote>



XBRL instance document: structure of the generated file

<?xml version="1.0" encoding….	*(/**Header)*
<xbrli:context id=….	*(Contexts generated)*
<unit id=….	*(/**Middle)*
<*primaryItem*……	*(Facts generated)*
<link:loc xlink:href……	*(Footnotes generated)*
</xbrli:xbrl>……	*(/**Tail)*