XBRL Generation, as easy as a database.

Christelle BERNHARD - christelle.bernhard@addactis.com
Consultant & Deputy Product Manager of ADDACTIS Pillar3®

INTRODUCTION

The need for transparency and control of financial institutions is constantly increasing. This is particularly true for Solvency II regulations in which reporting to the authorities and to the market plays a central role. Moreover, collection of data is intended to help compare financial institutions and prevent systemic risks. Consequently, reporting under Solvency II is facing several difficult challenges:

- First, the number of facts to be reported is significant and could even grow in the years to come.
- Second, the aggregation of all facts (data) collected requires a high level of standardization of the description of each and every data.
- Third, the facts to be reported are multi-dimensional and each fact is assigned several characteristics.
- Last but not least, facts are inserted in a web of reciprocal relationships and have to be consistent with one another.

Meeting all these requirements is only possible through the use of a common language to describe the facts reported. The chosen language for insurance, as well as for banks, is XBRL.

Based on a common taxonomy it aims at describing as accurately as possible the facts in their full complexity (currency, type, year, line of business…). Based on these descriptions XBRL generators will change the chaos of internal data of the company into a standardized identical reportable format for the regulator.

How does this generation happen? And more importantly why are we convinced the generator should be based on a database technology? These are questions we will try to address in this paper.

1 HOW XBRL WORKS?

The first question that arises is:

What is actually XBRL?

XBRL stands for eXtensible Business Reporting Language. It is a kind of XML language. Basically it is a succession of tags in which some information is given.

A company that reports under XBRL has to send a document called instance which contains the business report facts along with several characteristics of each fact. For a company, a fact corresponds to a reported value.

Each characteristic is described in one or several taxonomies and generally in a Data Point Model (DPM). The set of taxonomies is called a DTS (Discoverable Taxonomy Set).

When we talk about XBRL, we need therefore to distinguish between the instance and the DTS/DPM. Companies have to send the instance to their regulator and do not have to worry about the DTS/DPM. The DTS and the DPM are built by the authorities.

How big can an instance be? The idea of facts to report.

The following table gives the number of different facts defined in the test XBRL instances provided by EIOPA's Tool for Undertakings:

Reporting kind	Annual Solo	Annual Group	Quarterly Solo	Quarterly Group
Number of facts	30 269	15 533	670	450

As most companies have several solo entities, as well as a lot of lines of business, some ring-fenced funds, etc. …, the number of "facts" to report is huge.

Moreover the reporting framework is complex. There are lots of hierarchies and rules that govern the data.

Because of the complexity of the Pillar 3 exercise, it is necessary to map a data codification and build a Data Point Model.

Data codification: data centric vs form centric

The usual way of seeing data is as a table, in a two-dimensional universe. This way is called form centric.
For example, if I say "in the column 3 and the row number 10, I have the value 90", I adopt a form centric point of view.

	Column 1	Column 2	...
Row 1			
Row 2			
...			

Fig.1: The form centric view: a two-dimensional representation.

The main disadvantage of the form centric view is that it cannot support more than two dimensions. This is not enough for business reporting because a data point has usually more than two characteristics. If I want for example to report that the Solvency II value for Intangible assets in my balance sheet is 30 000€, I need at least 4 qualifications of the value 30 000:

- The value is reported in Euro
- The fact that it is under Solvency II
- The fact that it is about intangible assets
- The fact that it is for solo reporting

Obviously, some data points have less than 4 characteristics and other more than 4. The data model has to be flexible enough to qualify in an unlimited way each single data.

In the form centric view, we define for example data in the first column, then in the second column, etc. … Once this model set, it will be hard to change. If the authorities want for example to add a column between the first and the second one, what will be the column number? We realize with this example, that the form centric view is not flexible enough. Business reporters need a model that can be easily enriched.

This is why data modelers have imagined a model that is data centric. Data centric means that each data point (or fact) will be the intersection of several characteristics. In XBRL, a fact is indeed defined with an intersection of several dimensions and one metric. There will be always one single metric but the number of dimensions is not predefined. This data codification type is therefore very flexible.

If data centric is more flexible and accurate, form centric is more common for data reporting in companies. An XBRL generator has therefore to transform the form centric into a data centric structure.

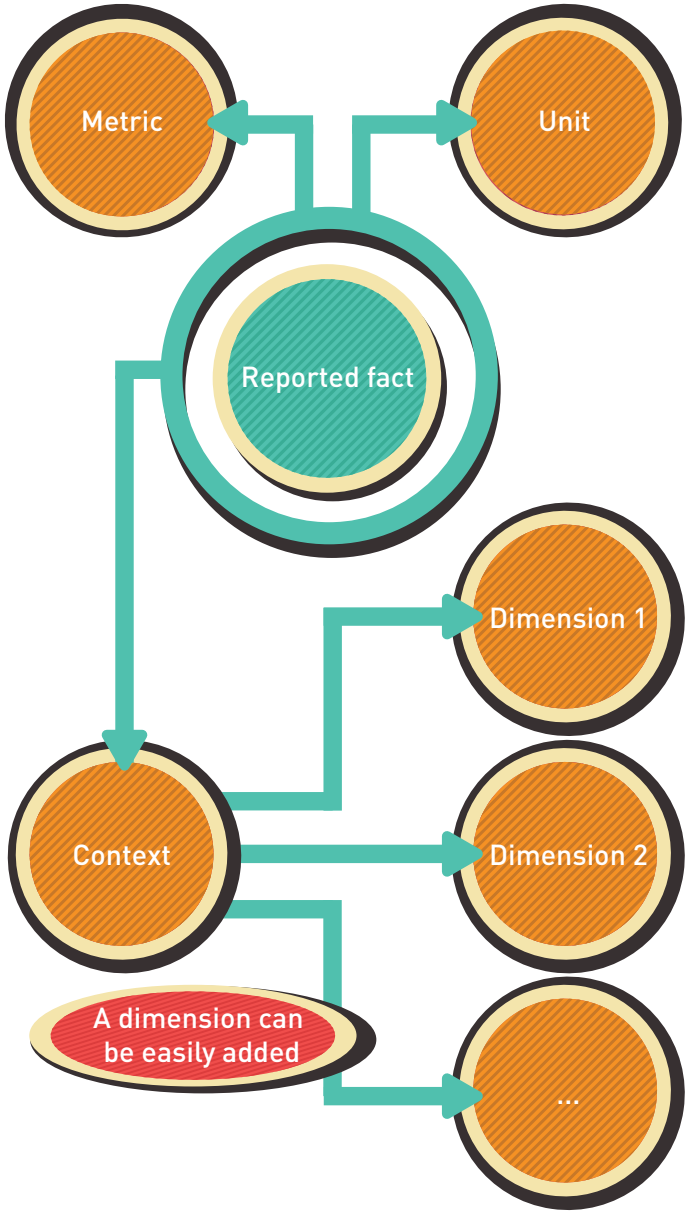


Fig.2: The data centric approach: the number of dimensions adapted to each reported fact.

Because of this complexity, EIOPA developed the Data Point Model that describes the modeling of the data (dimensions, domains …).
This DPM is captured in three formats: the DTS which contains the DPM, an Excel file and a SQLite database.

All the information needed to construct an XBRL instance is written in that DPM. One needs therefore not to read the entire DTS to write or read an instance.

2 WRITING AN INSTANCE

An instance is composed of:

- A fixed part:
  - The root tag <xbrl> with attributes and namespaces.
  - The Schemaref element, that gives the entry point into the DTS.
- A part depending on the global content of the instance:
  - The list of FilingIndicators, that indicates which reports are filled in the instance.
- The part describing every data which is a succession of:
  - Unit definitions, which are a list of the units used in the report.
  - Context definitions, which are a list of the dimensions relevant for a reported value
  - Facts expressed with the right metric, the reference to a context and (if needed) a unit and a precision.

The third part is the one that is the most interesting because it contains all the information to describe each single data. We will now focus on this part and explain how it works.

How can I report to the authorities that technical reserves for Health SLT in balance sheet 2014 on solo basis is 538 711.61€?

Step 1: Find the main characteristics of the value to report

- The reporting date is 31/12/2014
- The identifier for the entity is 123456789 (for this example)

Step 2: Find the characteristics around the data point

This can be found in the annotated templates. The annotated templates are presented in spreadsheets. 'Annotated' points to the fact that the dimensions, domains and metrics are specified for each data point.

Extract of the annotated templates:

Here we can see:

- In blue, the metric of the data point, which is "Metric: Monetary | BC/Liability | LB/Gross technical provisions [other than local GAAP specific]"
- In black, the dimensions which are:
  - o BL/Health non-SLT
  - o VG/Solvency II
- At the bottom left of the spreadsheet, we see that there is a dimension on the Z Axis which is : CS/Solo
  The Z Axis contains dimensions that are defined for one entire table.

Step 3: Find the correspondence in XBRL to the characteristics found in the step 2. This information is in the DPM Dictionary.

Step 3.1: Find the metric
XBRL code corresponding to the metric is in the spreadsheet named “met MD”:

The code is mi363 and the datatype is xbrli:monetaryItemType. This means that we need to give the unit (here Euro) and the precision (here 2 decimals) of the value.

Step 3.2: Find the dimension BL/Health non-SLT
First, find the applicable domain for the dimension BL in the spreadsheet ‘Dimensions’:

55	TG	Types of guarantee	CG	s2c
56	CB	Insurance classes	LB	s2c
57	BL	Line of business [general]	LB	s2c
58	RB	Related line of business	LB	s2c
59	TB	Insurance/reinsurance business	LB	s2c

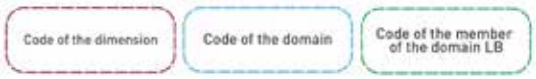
BL corresponds to Line of business and has the applicable domain code LB.

Open the spreadsheet named LB and find the code corresponding to Health non-SLT:

Health SLT	x45
8: Line of business [general]	
Total/NA	x0
Non-life	x75
Health non-SLT	x43
Life	x60
Health SLT	x45

The code for the Line of Business Health non-SLT is x43.

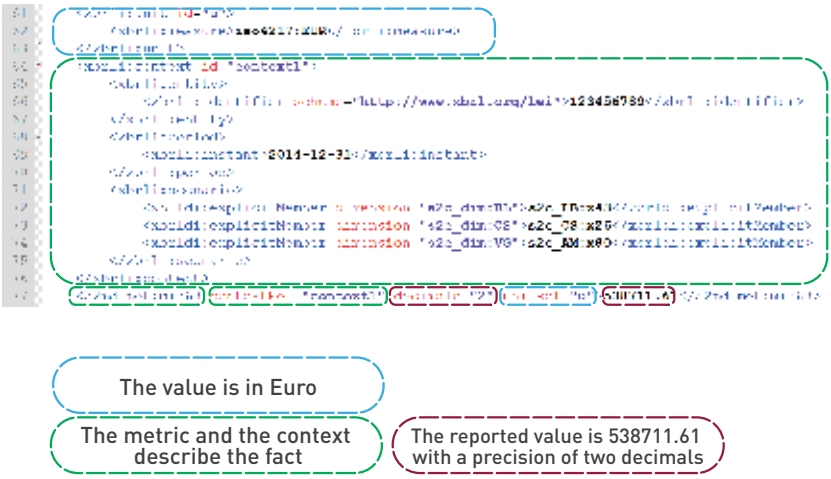
The XBRL syntax of this dimension will therefore be:
<xbrldi:explicitMember dimension=»s2c_dim:BL»>s2c_LB:x43</xbrldi:explicitMember>



Step 3.3: Find the dimension VG/Solvency II
First, find the applicable domain for the dimension VG in the spreadsheet 'Dimensions': applicable domain code is AM. Open the spreadsheet named 'AM' and find the code corresponding to Solvency II. The code is x80.

Step 3.4: Find the dimension CS/Solo
First, find the applicable domain for the dimension CS in the spreadsheet 'Dimensions': the applicable domain code is 'CS'. Open the spreadsheet named 'CS' and find the code corresponding to solo. The code is x26.

Step 4: Write all information gathered previously in the XBRL instance:



3 BEST TECHNICAL WAY TO GENERATE AN XBRL INSTANCE?

We have previously seen how we can write one XBRL fact with the DPM as an input. Now, the question that arises is: can I use the same procedure and automate it for my reporting? May I generate the instance using the DPM under the spreadsheets format?

We also mentioned that the DPM can be found in two other formats: the DTS and a Database. What is the best format to use for XBRL generation?

Generating an instance with spreadsheets as an input?

Based on the previous example, it can be derived that instances could be generated directly from the spreadsheet format. And rightly so. If we want to use the annotated templates only, we “simply” have to automate the steps followed in the previous section:

- Identify for each data point the metric and the dimensions part of the context:
  - For this we need the annotated templates
  - We will obtain a list of strings
- Find the metric in the DPM dictionary
- For each dimension, find:
  - The dimension code (eg VG)
  - The applicable domain code (eg AM)
  - The code of the member (eg x46)

This has several disadvantages:

- If the regulator decides to add a column in the report the first step of the algorithm will have to be changed accordingly.
- Comparing strings is always a tricky exercise: if there is only one character missing, then there is a risk that the program won’t find the string.
- It is a very slow process.
- Spreadsheets do not have internal restrictions: a potential mistake is undetectable.

This solution, if theoretically possible, isn't reliable in a project as ambitious as insurance reporting. It could fit a reporting limited in size to some hundreds figures. When it comes to 30.000 facts, it just doesn't sound credible and/or possible.

Moreover, the annotated templates and the DPM dictionary have inter and intra dependencies. Indeed, the annotations of each fact in the annotated templates refer to dimensions or metrics in the DPM dictionary. Then, each dimension has an applicable domain that is the key to find the code of the dimension. Writing a fact is therefore following a process in which each milestone contains a relation to the next step. Indeed, the process presented in section 2 could be summarized as follows:

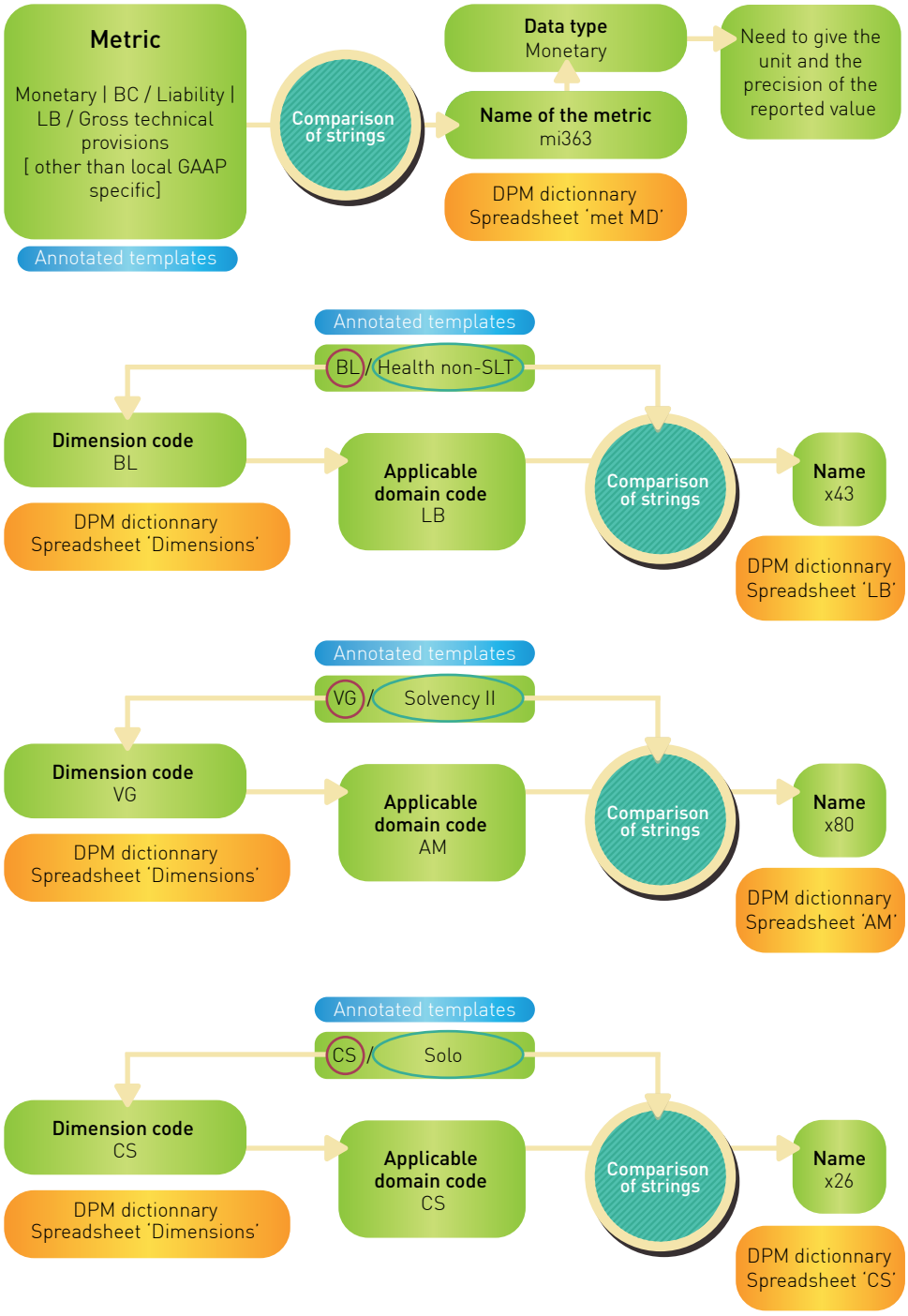


Fig.3: Writing a fact with the DPM in spreadsheets format is a complex process.

A lot of relations have to be explored in order to be able to write a fact. Spreadsheets are obviously not suitable for following easily and quickly so many relations.

A XBRL instance generator should therefore be based on a technology that is more reliable and that can manage relations and multidimensional data.

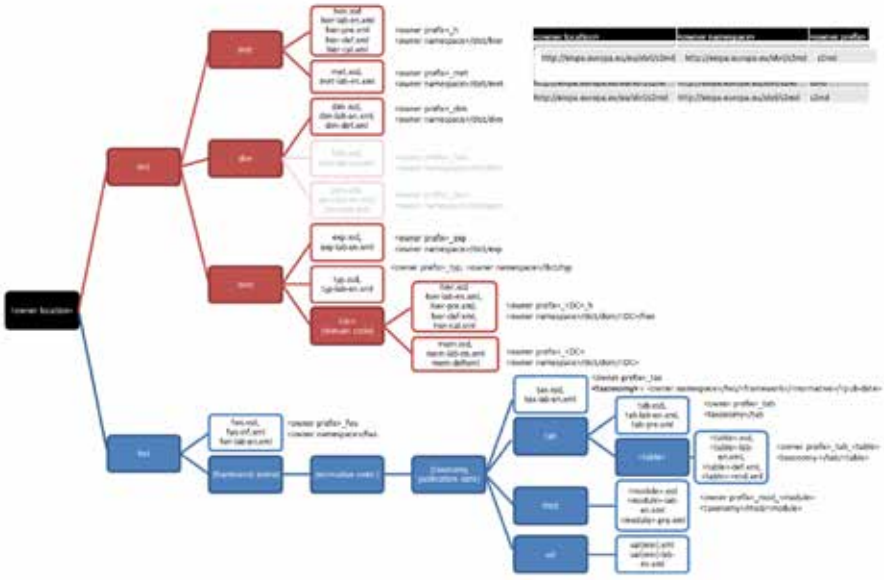
Generating an instance with the DTS as an input?

According to the XBRL 2.1 recommendation, a DTS is a set of several taxonomies that can be interconnected. “A taxonomy is comprised of an XML schema and all the linkbases contained in that schema or directly referenced by that schema”. The linkbases in taxonomy express relationships between concepts and relate those concepts to their documentation.

The DTS is a web of relationships that has to be discovered in order to be able to manipulate the dimensions, domains, etc … needed to construct an instance.

Concretely, the DTS is a compressed folder of a huge number of files and folders. The architecture of this set is complex as it is highlighted in the following schema:

VIII Annex 3. EIOPA Solvency II Preparatory XBRL Taxonomy: Owners, Folders, Files, Namespaces and Prefixes



Source: EIOPA Solvency II DPM and XBRL Preparatory Taxonomy Framework Architecture.pdf (in the version 1.5.2. of the Solvency II taxonomy package).

The number of files and folders is indeed significant:

Version number of the taxo-nomy package	2013 -06-30	1.0	1.2	1.4.1	1.5.2	1.5.2.b
Date of publica-tion	30 June 2013	November 2013	January 2014	May 2014	July 2014	December 2014
Nb of folders	623	905	911	492	490	491
Nb of files	2522	3667	4551	2499	2537	2540
Total nb of lines	432 819	1 428 590	1 590 249	1 010 705	1 182 666	1 212 444

The last line is the sum of the number of lines of each single file in the package. Each file contains information but also relations to other files.

Like spreadsheets, flat files are not suitable for expressing relations and multidimensional structures. Moreover, flat files are easily modifiable which can lead to undetectable mistakes. The best way of presenting an architecture that has a significant number of relations is in a database.

Generating an instance with the Database as an input?

The problem of structured data and relations is not new in the IT world. It gave birth in the past to the idea of relational Database: it is nowadays clear that when it comes to structured data and relations between concepts it is more natural to describe the model in terms of database. In fact, databases are made to organize data and relate different data structures in a fast and reliable way. A database is actually a set of related data organized in tables.

The solutions used by IT to solve the problem of huge sets of interconnected data are the key to simple management of XBRL formats. This is why EIOPA is providing a Database format for the XBRL reporting.

The database used by the EIOPA's Tool for Undertakings like the DTS and the spreadsheets contains all the information needed to write an instance. However because it is a database, this information is more easy to reach for a program and better organized (only 30 tables needed in the database when there are 2540 files in the DTS!).

Moreover, there are several tools (like primary key, foreign key constraint etc…) in databases to enforce an internal consistency.

This is why we feel that technology of reporting should recur to Database structure to ensure better stability, coherence over time as well as easier management of different versions (because there will be future versions…).

The insurers involved in the question of XBRL should therefore pay a lot of attention to technologies embedded by generators.

SUMMARY

Standardized insurance reporting is an ambitious challenge for regulators as well as for the insurance industry. Guaranteeing that all data of all European companies will be uniformly specified is not easy to achieve. Only the use of a standardized taxonomy will allow regulators, investors, law makers to compare companies and to identify systematic risks in the insurance industry. To get to this point the regulators have chosen the XBRL as a reporting language. No doubt that there will be a long way to the desired uniformity of reporting. Along this route, insurers have to tame the rules and specifications of XBRL. We hope we showed in this article and in the example of writing an instance that the main limit to a robust and good quality reporting lies into the capacity to address the issue of the multi-dimensional nature of business data. The only way to address this multi dimensionality of data is clearly to refer to a database structure of the DPM. Other solutions are not stable and reliable in the longer term.

Complexity of XBRL makes it improbable that insurers will develop themselves a generator for reporting.

They will obviously rely on providers. The main question they then have to face when outsourcing such an important activity is the organization of the XBRL generator of the provider. We believe they would be better going to providers like ADDACTIS Worldwide that based their generators on Database technology.