The XBRL Book

Simple, Precise, Technical.

by Ghislain Fourny

January 6, 2017

The XBRL Book by Ghislain Fourny

http://ghislainfourny.github.io/the-xbrl-book/

Revision history for the First Edition
2017-01-11 First Release

ISBN 978-1535117746

Trademarks:

XBRL is a registered trademark of XBRL International, Inc. XML, XML Schema, XLink and XML Base are recommendations by the World Wide Web Consortium (W3C). All other trademarks are the properties of their respective owners.

Example data:

A lot of the examples of this book are largely inspired by the samples by Charles Hoffman, CPA, which he kindly provides as public domain. Some other examples are taken from fiscal filings publicly submitted to the SEC via the EDGAR system.

Intellectual Property Status of the XBRL specifications:

XBRL International requires educational content that explains XBRL to include the following paragraph referring to the XBRL specifications.

Contents

Chapter 1

Introduction

1.1 Why this book?

XBRL is currently gaining increasing acceptance worldwide, as reporting authorities encourage or require companies or banks to submit their reports in this format. There is a reason for this: data submitted as XBRL filings, that is, where each value or piece of information is tagged with its meaning and context, can be processed automatically by machines.

As a result, an ecosystem of tools and vendors has emerged, and people from all backgrounds — IT, Business, Finance, ... — are working together to further establish and grow this technology.

When I began to dive into the XBRL world, a few years ago, I searched for books on the matter. I quickly realized that, as a technical person with a scientific background, I could not find any that would explain XBRL at the appropriate level of abstraction: almost all books — for excellent reasons, it has to be said — focus on the business aspects of XBRL, and how to use it best within a company. They thus stick to high-level technical explanations. I ended up spending most of my time reading the XBRL specifica-

tions directly. I could also exchange emails with Charlie Hoffman, to whom I am extremely grateful for his insights, vision and patience.

Later, I did finally find a few more books that go into deeper technical details, however they do so at the level of the XML and XML Schema syntaxes rather than at the higher semantic level of XBRL itself. Of course, this is very useful for XML-savvy people to get started, but after a few years working with this technology, I became more and more conscious that understanding XBRL in terms of XML is a bit like learning C++ or Java in terms of the assembly code they produce.

Put simply, this book is the simple and precise starting point I wish I had had when I first encountered XBRL. Most of these pages do not require any XML or XML Schema knowledge, except for the one section in each chapter that goes into details about the syntax — but the latter can easily be skipped on a first read.

1.2 Business reporting today

Among the various software technologies that have been designed in the era of computers, there is one in particular that enjoys undisputed popularity among business users, from financial analysts to executives, through consultants and business analysts: spreadsheets.

While spreadsheets support some automation with the popular VBA scripts, they lack the ability to express their content in a standardized way that computers could understand and act on without human intervention.

XBRL solves this problem. Fiscal reports can be submitted to, say, the SEC in a uniform and standardized way, enabling financial analysts to create their reports on the fly and automatically. Moreover, it is easier than ever to validate the data filed to SEC and eliminate accounting mistakes, leading to reports of higher and

higher quality every year. As of October 2015, Charlie Hoffman reported that two XBRL editors already reached the 90% correctness threshold — meaning that 90% of their XBRL filings showed no error for the validation criteria used, such as the fact that assets must match liability and equity.

1.3 Instances and taxonomies

The data in XBRL is submitted in a so-called XBRL instance. An XBRL instance contains facts. Each fact is a value that is tagged with a context that describes what it is about.

Instances are never submitted alone. They are submitted along with a taxonomy that gives meta-information about the contexts used in the reported facts. This part of XBRL is also known as a DTS (Discoverable Taxonomy Set).

Taxonomies can be created by reporting authorities so that companies can create their reports in a homogeneous way, but taxonomies can also be extended by them to fit their individual needs.

Taxonomies include formulas and validation rules that check that the submitted data is sound. They also organize the metadata in a structured way, just like a fiscal report on paper is structured and presented for being read by an investor.

1.4 Misconceptions about XBRL

We will start our journey through XBRL shortly. But first, let us go through some statements about XBRL that are commonly assumed, but I think are not right.

1.4.1 XBRL is XML

Yes, and no. Actually, no.

XBRL does use the XML syntax, which you will see if you open it with a text editor. But the same applies to an Excel (.xlsx) file: rename it as .zip and open it, and you will find XML. Yet people do not think of Excel as XML. The same goes for XBRL.

XML describes semi-structured documents that look like trees. XBRL describes facts organized in a tabular and structured way. For the record, XBRL does support hierarchies for organizing the metadata, but these hierarchies are not expressed using natural XML hierarchies: they are expressed using linkbases (XLink) as flat lists of edges.

So, XBRL is not XML and one could imagine having alternate syntaxes such as JSON or YAML or, for all we know, Markdown.

1.4.2 XBRL is complicated

Certainly, if you open an XBRL instance and taxonomy in a text editor and view the XML syntax, you will find it to be very complicated. If you attempt to read the original specifications, you will find very precise and intricate technical language.

Yet, the burden of this complexity should be, and is increasingly carried by the designers of XBRL tools. Only they programmaticaly manipulate XBRL at the XML level, and only they (should) read the specifications to ensure they are compliant.

From a user's perspective, XBRL is not very complicated to understand, and the best XBRL tools are the ones that manage to shield their users from the machinery. In the end, creating or reading XBRL filings should not be more complicated than using spreadsheet software.

1.4.3 XBRL is only for business reporting

Business reporting: this is what the B and the R in XBRL stand for. Yet XBRL is much more general that this and pretty much anything can be stored in the XBRL format.

1.4.4 XBRL does not scale up

Most vendors are able to validate XBRL data at the level of a filing. It is a challenge to make XBRL scale up in the sense that you can process the data over hundreds, thousands of filings. But not impossible. Actually, because XBRL cuts data into small atoms — facts — that all look alike from far away, it is all set for scaling up elegantly.

1.4.5 XBRL lacks of standardization

XBRL *is* a standard. It standardizes business reporting like it has never been done before. This is even the primary reason why the use of XBRL should be encouraged.

Of course, it still gives a lot of freedom to users: there are lots of different ways to express the same data in XBRL. There can still be some heterogeneity in the way companies build their metadata hierarchies, or in the terminology they use. Yet also this can be standardized, XBRL lays the groundwork for further standardization on a higher level. This is no different from languages such as C++ being very powerful, and companies enforcing styleguides to restrict their usage to a simpler subset.

In XBRL, some styleguides, also called application profiles, are already establishing themselves, for example many reporting authorities are taking over most SEC practices, and another practice establishing itself, for example in EBA filings (COREP, FINREP), is called the DPM (Data Point Model). The DPM is to XBRL a bit what REST is to HTTP: REST is "HTTP done right," and one could say the DPM — or at least, its core idea because its design goes beyond this — is "XBRL done right," in that it re-expresses the semantics of identifying and slicing and dicing facts in XBRL in a clean way. To be fair to other practices, all of them do in fact adhere to the DPM's core principles.

1.5 Samples

Some samples of XBRL instances, taxonomy schemas and linkbases are available online at `http://ghislainfourny.github.io/the-xbrl-book/samples.html`. More will be added.

These samples are provided as a complement to the material covered in this book. They can be opened in a text or XML editor for those who are interested in understanding the XML syntax, but they can also be opened by XBRL processors to be viewed on a higher level and without worrying about syntax.

They are grouped by chapter, and were designed in such a way that they only use the material covered so far. A lot of credits are to be given to Charles Hoffman, as they were largely borrowed from the samples that he designed to support and showcase his work on commonly used patterns.

Chapter 2

Facts

The R in XBRL stands for reporting. If XBRL could be summarized in one single definition, it would be this: XBRL is about reporting facts. In this chapter, we introduce the notion of a fact, analyze in details what it is made of, to finally arrive at the raw syntax in which this fact is reported in an XBRL instance.

2.1 Atoms of data

The XBRL paradigm is based on the idea that data can be broken into very small chunks, in such a way that each chunk makes sense all by itself, while being irreducible to anything smaller, at least in a meaningful way. Hence, each of these chunks can be seen as an atom of data. These chunks are called facts.

An example of fact is that Coca Cola had $91,016,000,000 of assets as of April 3, 2015. Another example of fact is that the π constant is 3.1415 with a precision of 4 decimals, at all times.

A fact is a value that carries a context. In the first example, the value is 91,016,000,000, while the context specifies that these are the assets owned by Coca Cola on April 3, 2015, in U.S. dollars.

The context is crucial: the value alone would be useless, and it is the context that makes the fact self-explaining. If you simply give the value 91,016,000,000, and only this value, to somebody else, this person will not be able to do much, except maybe look for its mathematical properties. If however you give this value together with its context, that is, the entire fact, to another person, they will have all they need to understand it, and reuse it in a different environment, for example to generate a fiscal report for the company at hand, or a comparison across Dow 30 companies. As such, a fact is not only data, it is information.

2.2 Aspects

Let us now look more carefully to contexts. One of the requirements of XBRL is that, even though a fact can be understood by a human, it can also be processed by a machine. This implies that a context cannot be an informal description of what the value is about: it must have some structure.

Indeed, a context is made of a list of characteristics that qualify the fact. In our first example, the context associated with the value 91,016,000,000 has the following characteristics:

- These are assets;
- They belong to Coca Cola;
- The value is true as of April 3, 2015;
- They are expressed in U.S. dollars.

Looking closer, it can be seen that each characteristic is made of what is called an aspect, and of a value associated to this aspect. We can rewrite the above context as follows:

- What: Assets;

- Who: Coca Cola;
- When : April 3, 2015;
- Of what: U.S. dollars.

In this simple example, the aspects used are all standard XBRL aspects, in that they are specifically defined in the XBRL specifications because of their universality: The aspect describing what a value is about is called Concept. The aspect describing about whom this value is is called Entity. The aspect describing when this value holds is called Period, and the aspect describing the unit of the value is called Unit. So, a form of the context that is now very close to the way a fact is reported in XBRL[1] is:

- Concept: Assets;
- Entity: Coca Cola;
- Period: April 3, 2015;
- Unit: U.S. dollars.

Many more aspects can be created and used to describe the context of a fact, for example geographical aspects such as countries, or company subdivisions, what-if scenarios, the time at which the fact was reported or updated, and so forth.

[1] At that point, the reader familiar with the XBRL specification may point out that XBRL excludes concepts and units, as well as languages, from the context associated with a fact. However, this is more a technical detail than something that is semantically relevant to XBRL, and concepts as well as units are still considered aspects in other XBRL specifications. For pedagogical purposes, it is much easier to consider that they are part of the context as well, which we do here. Also from a data model perspective, this is the right thing to do.

Aspect	**Characteristic**
Concept	Assets
Entity	Coca Cola
Period	April 3, 2015
Unit	U.S. Dollars
Fact value	91,016,000,000

Table 2.1: Our example fact in tabular form, one characteristic per row.

Concept	**Entity**	**Period**	**Unit**	**Value**
Assets	Coca Cola	April 3, 2015	U.S. Dollars	91,016,000,000
Assets	Coca Cola	December 31, 2014	U.S. Dollars	92,023,000,000
Assets, Current	Coca Cola	April 3, 2015	U.S. Dollars	32,119,000,000
Assets, Current	Coca Cola	December 31, 2014	U.S. Dollars	32,986,000,000
Other Assets, Noncurrent	Coca Cola	April 3, 2015	U.S. Dollars	4,602,000,000
Other Assets, Noncurrent	Coca Cola	December 31, 2014	U.S. Dollars	4,407,000,000

Table 2.2: A fact table, displaying several monetary facts in structured form, one fact per row.

2.3 The tabular model

With this last description of the context, it should by now have become apparent that XBRL, and facts in general, are of a tabular nature. A single fact can be displayed as shown on Table 5.1.

Because of this structure, several facts can be displayed in a table form: this is called a fact table. In a fact table, each fact appears in a row, and each column corresponds to an aspect, plus one column for the value. For example, one can include further facts from the same fiscal report, as shown on Table 2.2.

Table 2.3 shows another fact table that contains textual facts. It does not have any Unit aspect, but a Language aspect is present. Table 2.4 shows a fact table merging the first two. It has empty cells because not all aspects apply for all facts.

Concept	Entity	Period	Language	Value
Name	USA	Januar 1, 2016	English	United States of America
Name	USA	Januar 1, 2016	German	États-Unis d'Amérique
Name	USA	Januar 1, 2016	French	Vereinigte Staaten von Amerika

Table 2.3: A fact table, displaying several textual facts in structured form, one fact per row.

Concept	Entity	Period	Unit	Language	Value
Assets	Coca Cola	April 3, 2015	U.S. Dollars		91,016,000,000
Assets	Coca Cola	December 31, 2014	U.S. Dollars		92,023,000,000
Assets, Current	Coca Cola	April 3, 2015	U.S. Dollars		32,119,000,000
Assets, Current	Coca Cola	December 31, 2014	U.S. Dollars		32,986,000,000
Other Assets, Noncurrent	Coca Cola	April 3, 2015	U.S. Dollars		4,602,000,000
Other Assets, Noncurrent	Coca Cola	December 31, 2014	U.S. Dollars		4,407,000,000
Name	USA	Januar 1, 2016		English	United States of America
Name	USA	Januar 1, 2016		German	États-Unis d'Amérique
Name	USA	Januar 1, 2016		French	Vereinigte Staaten von Amerika

Table 2.4: A fact table, displaying several textual and monetary facts in structured form, one fact per row.

2.4 Instances

When facts are reported, they are batched and reported in what is called an XBRL instance. Often, it can also be called a report or a filing. An XBRL instance can simply be seen as a flat list of facts. For example, the quarterly report of Coca Cola submitted to the SEC for the fiscal period Q1 2015 is an XBRL instance.

Taking the example of the yearly and quarterly fiscal reports submitted to the SEC, an XBRL instance typically reports between 500 and 2000 facts.

2.5 Collisions

So, an XBRL instance can be seen as a bag of facts, with each fact having a value and a context against which this value makes sense. A question that arises naturally is: what happens if several

facts are reported against the same context, in other words, if facts collide with each other? Can it happen at all?

The answer to the latter question is yes. Not only can it happen, because the XBRL specification does not specifically forbid it, but it also does happen in practice. There are several approaches to the question of colliding facts.

2.5.1 Why it is a good thing to allow them

Intuitively, it would seem like a good idea to simply forbid fact collisions. But it would be infeasible in practice. At the scale of a single XBRL instance of course, it is an easy task to check for collisions, because there are not so many facts. Actually, good XBRL software should probably warn you if you are attempting to generate an instance with colliding facts.

However, on much bigger scales, such as all instances submitted to an authority, or even all XBRL instances worldwide, this is simply unrealistic, because we are talking billions and billions of facts, distributed across millions of machines. Of course, it could happen some day, if XBRL becomes mainstream and establishes itself, that some standardized mechanism and infrastructure allows for a worldwide collision detection, but there is nothing like this yet as of 2015.

Furthermore, it is not necessarily a bad thing to allow for collisions to happen. The recent experience in the database world showed that one can handle much vaster quantities of data if one gives up, or at least makes compromises on, consistency. Concretely, if collisions are allowed, it makes it much simpler and more efficient to scale up the production, exchange and storage of XBRL facts.

2.5.2 Detect a collision

Collisions can be detected by comparing contexts: if two facts have the exact same aspects, and each one of these aspects is associated with the same value for both facts, then these facts are duplicates and collide. The XBRL specification provides more involved technical machinery to describe this[2], but this is the essence of it.

2.5.3 Amendments

What to do when a collision is detected depends on the values of the colliding facts: if these values are identical, the facts are consistent with each other. If however the values diverge, it requires more care in the semantics of this divergence.

A very common use case found in practice is that of amendments. In the United States, companies that report to the SEC have the possibility to resubmit facts with updated values, either in a special amended report, or in the next period. For example, a fact reported in a Q1 report (a 10-Q report) may be updated in the next Q2 report (also 10-Q), or in a Q1 amendment report (a 10-Q/A report).

In this case, the collision is easy to solve: the latest reported value has to be taken. A more involved solution would involve adding an aspect with the time at which the fact was reported (database people call this "transaction time", which removes the collision completely).

2.6 Precision and decimals

In an ideal world, for example in mathematics, values are exact. Any physicist will however tell you that, in practice, values are always given with a margin of error, or at a certain precision.

[2]Equality predicates such as structure equality, value equality, parent equality, context equality, unit equality and XPath equality, defined recursively

XBRL supports annotating a fact value with information about its precision.

For example, let us consider the following value, $\frac{22}{7}$, which is an approximation of π:

$$3.142857$$

The first few digits are correct, however at some point, it deviates from the actual value of π. Let us mark the digits that are accurate in green, and those that are not in red:

$$3.142857$$

XBRL provides two different frameworks for expressing exactly this:

- *Precision of 3*: The first 3 significant digits (that is, not including any zeros in the front: 3, 1 and 4) are correct;
- *2 Decimals*: The value is correct up to 2 digits after the decimal period (1 and 4).

Let us take another example: the Earth-Sun distance (astronomical unit), of which we consider that the trailing zeros are imprecise:

$$149,597,870,700$$

This leads to a negative value of the Decimals property, because this time the imprecision happens before the period:

- *Precision of 10*: The first 10 significant digits (that is, not including any zeros in the front: 3, 1 and 4) are correct;
- *-2 Decimals*: The value is correct up to 2 digits *before* the decimal period.

The two ways are equivalent, in that, given the precision, it is possible to infer the decimals, and given the decimals, it is possible to infer the precision.

For example, let us go back to the assets of Coca Cola on April 3, 2015. Typically, values in balance sheets are given up -3 or -6 decimals after the period (the last 3 or 6 digits before the period are not considered precise). In the present case, Coca Cola reported them with -6 decimals:

$$91,016,000,000$$

Since this number has 11 digits before the period, we can equivalently say that the value has a precision of $11 + (-6)$, that is, 5 significant digits.

There is a special value for infinity, used for exact representations: if the value reported is exact, it has infinite Precision and an infinite number of Decimals.

When an XBRL instance is produced, exactly one of the two properties, precision or decimals, has to be provided. The other property also exists in any case, but it will be inferred automatically.

2.7 Basic aspects

Facts have a context made of characteristics, and each characteristic is a value associated to an aspect. The number of aspects in facts is virtually only limited by imagination. However, there are a few ones that are standard and common and that we describe here: concept, entity, period and unit. All facts have a concept, entity and period, but not necessarily a unit.

Further chapters will introduce how new aspects (called *dimensions*) can be created and used.

For ease of language and for the sake of a smoother read, we will use the terminology "concept of a fact" to mean "the value

associated with the concept aspect in the context of the fact", and likewise for the other aspects ("period of a fact", "entity of a fact", "unit of a fact").

2.7.1 Concept

The concept aspect describes *what* the value is. For example, it can be assets, or it can be income, or it can be the total of a bill.

Concepts are the most important aspect in XBRL, and are described in greater detail in chapter 3. In particular, the concept of a fact has an implication on what value, period and unit is allowed.

Internally, they are identified with qualified names, which are described in Section 3.3.

2.7.2 Entity

The entity aspect describes about *whom* the fact is. In the very widespread case of fiscal reports, it is the company that is reporting their own fundamentals.

Entities are identified in a way that can change from country to country, from stock exchange to stock exchange, or even from regulator to regulator. In the USA, the SEC assigns so-called CIKs (Central Index Keys) to all companies, for example Coca Cola has CIK 21344. XBRL allows for any scheme, but of course requires that you specify which scheme you use every time an entity is identified.

2.7.3 Period

The period aspect describes *when* the fact is valid (database experts will know this as *valid time*, as opposed to *transaction time*).

XBRL allows for two kinds of periods:

- Instant periods, which can be a single point in time, such as April 3, 2015 or November 11, 2011 at 11:11am.

- Duration periods, such as January 1st, 2014 thru June 30th, 2014. A special kind of duration period is the forever period, which means that the fact is valid at all times.

Whenever a time is not specified but only a date is given, the time is implicitly assumed to be midnight at the end of the day, that is, 24:00[3].

2.7.4 Unit

The unit aspect describes what the value designates, in other terms, whether what is being counted is apples or pears.

Units can be simple, such as currencies (U.S. Dollar, Swiss Franc, Euro, British Pound, Japanese Yen and so on) but also as complex as physical units with products and possibly a ratio: m, km/h, $N.m^2/kg^2$ or also, say, for dividends, dollars per shares, and so on.

XBRL defines standard units such as *pure* and *share*, and many more in unit registries.

There are some constraints on the usage of units. For example, if the fact value is not a number, the fact cannot have a unit aspect at all (not even the pure unit). Or if the fact value is expressed in a currency, then the unit must be a currency code following the ISO 4217 standard.

Units will be described in more details in Section 3.4. Internally, they are identified with qualified names, which are described in Section 3.3.

2.7.5 Language

The language aspect describes in what language the value of the fact is (English, German, Japanese...), in case it is textual. The

[3]Which is equivalent to midnight at the beginning of the following day, the latter being used in the specification for technical reasons.

value of this aspect must be a language code, as described in Section 4.3.

This aspect can only appear on facts that report a string value.

Note that the XBRL specifications diverge regarding whether the language of a fact value should be an aspect or not. The core XBRL specification will consider facts in multiple language as duplicates, hence not considering language an aspect. The same goes in an official document about duplicates in XBRL, which leaves it to the filers to have or not facts in multiple languages.

However, the newer Open Information Model working draft considers languages an aspect, and we think that this is the right way to go from a data modelling perspective.

2.8 XML syntax of an XBRL instance

Hopefully, upon reading this chapter, the reader should have forgotten that XBRL is XML, and if so, then it means that the author's point came across that XBRL really *is not* XML, even if it *uses* the XML syntax, and could use any other semi-structured syntax such as JSON or YAML.

Since XML is the standard syntax used to store and transfer XBRL facts, though, this book would not be complete if it did not also give a few hints on how this syntax looks like. In this section, we describe how XBRL facts are encoded in XML.

The first important point is that, in the XML syntax, a fact, that is, its value and its context, is split into three parts:

- The context, but without the concept and the unit;
- The unit;
- The concept and the value of the fact.

In the raw XBRL specification, contexts exclude concepts and units. To avoid any ambiguity, we will continue to use the word

context including them, but refer to contexts that exclude them as *syntactic contexts*, or use a teletype font such as `xbrli:context`.

The main motivation is reusability: syntactic contexts and units can be shared and reused across several facts, which saves a lot of space.

2.8.1 XML content ahead

The readers who do not intend to use or look at the XML syntax of XBRL can safely skip the remainder of this section, as well as all sections on the XML syntax of XBRL in each chapter. This is typically the case for users and consumers of an XBRL processor, who have the ability and luxury to work on a more abstract level than the syntax.

From the sake of simplicity and modularity, we assume from now on that the reader is already familiar with the XML and XML Schema technologies. In particular for this section, we assume that the reader knows about XML concepts such as elements, attributes, text, comments, QNames and namespaces. If such is not the case, we kindly refer to books on the matter, as knowing XML and XML Schema well is strongly advisable for any engineer who needs to directly produce, read or update XBRL syntax. This is typically the case for developers of XBRL processors.

We will introduce the XML syntax of XBRL through examples rather than schemas or detailed descriptions, in order to convey the overall taste of it. For reference or specific clarifications, the XBRL specification remains the ideal place to look up.

2.8.2 Overall structure

So, how does an XBRL instance look like (at least the part that contains the facts)? It is made of a root `xbrli:xbrl` element, of which the children are `xbrli:context`, `xbrli:unit` and fact elements (the actual facts), which reference contexts and units.

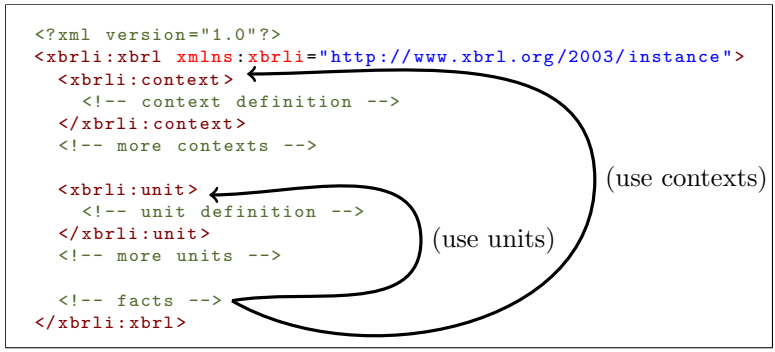

Figure 2.1: The skeleton of an XBRL instance: first contexts, then units, then facts. Each fact references a context and a unit.

Namespaces

The `xbrli:xbrl`, `xbrli:context`, `xbrli:unit` elements are all in the XBRL instance namespace `http://www.xbrl.org/2003/instance`, which is typically associated with the prefix `xbrli`. Often, it is made the default namespace for less verbosity, but for pedagogical reasons, we will use the prefix `xbrli` throughout this book.

So, the skeleton of an XBRL instance, would be as shown on figure 2.1, with at least one context and one fact mandatory.

We will now go into more details about each of the three kinds of elements.

2.8.3 Context

The syntactic `xbrli:context` describes the context of the fact, that is, all aspects except concept, unit and language. Since for now, we have only seen entity and period, we will only give the syntax for these two and come back to it later with new aspects.

```
<xbrli:context
      xmlns:xbrli="http://www.xbrl.org/2003/instance"
      id="cocacola-in-april">

  <xbrli:entity>
    <xbrli:identifier scheme="http://www.sec.gov/CIK">
      0000021344 <!-- This would be Coca Cola -->
    </xbrli:identifier>
  </xbrli:entity>

  <xbrli:period>
    <xbrli:instant>2015-04-03</xbrli:instant>
  </xbrli:period>

</xbrli:context>
```

Figure 2.2: A `context` element. It contains at least the entity and period – but not the concept and unit.

Figure 2.2 shows a context element that contains an entity and a period definition. A context must be identified with an `id` attribute, so that facts can refer to it. The semantics of `id` attributes is that defined by XML, in particular they must be unique within an XML document.

Entity

In a context, the entity is defined with an `xbrli:entity` element that has an `xbrli:identifier` child element.

The `xbrli:identifier` element is made of a `scheme` attribute URI and a value. The value is simply a string, and its format depends on the scheme used. In the example on Figure 2.2, the scheme used is the official CIK scheme of the Securities and Exchange Commission. The CIK of Coca Cola is 21344, but they are typically extended to a 10-digit string.

Period

The period is defined with a `xbrli:period` element. Its children depend on the kind of period. Figure 2.3 shows further examples of syntax for periods.

An instant period uses a single `xbrli:instant` element. The `xbrli:instant` has an XML Schema type of either `xs:date` or `xs:dateTime`, that is, a union of these types. Both formats use ISO 8601, with lexical values such as "2015-10-30" for an `xs:date` or "2015-10-30T08:00:00.000Z" for an `xs:dateTime`.

A duration period, which is a time interval, needs to specify two points in time instead of just one: a starting point and an ending point. Hence, it has two child elements: `xbrli:startDate` and `xbrli:endDate`. Both `xbrli:startDate` and `xbrli:endDate` are of type either `xs:date` or `xs:dateTime`, in the same way as instant periods.

An exception is the special duration period "forever", which is represented with simply an empty `xbrli:forever` element.

The elements `xbrli:instant`, `xbrli:startDate` and `xbrli:endDate` call all contain a value of the XML Schema Types date or a dateTime.

```
<xbrli:period xmlns:xbrli="http://www.xbrl.org/2003/instance">
  <xbrli:instant>2015-04-03T12:00:00</xbrli:instant>
</xbrli:period>
```

(a) An instant period, in this case an `xs:dateTime`

```
<xbrli:period xmlns:xbrli="http://www.xbrl.org/2003/instance">
  <xbrli:startDate>2015-04-03</xbrli:startDate>
  <xbrli:endDate>2015-10-03</xbrli:endDate>
</xbrli:period>
```

(b) A duration period (except forever), in this case `xs:date`s

```
<xbrli:period xmlns:xbrli="http://www.xbrl.org/2003/instance">
  <xbrli:forever/>
</xbrli:period>
```

(c) A forever period

Figure 2.3: The XML syntax for each kind of period – forever, even though it is a duration, has a different syntax.

2.8.4 Unit

Units are defined in a way similar to contexts, using the `xbrli:unit` element, and also require an `id` attribute so facts can refer to them. The most basic units are straightforward to define with a `xbrli:measure` child element. Unit values are QNames, so you need to bind any namespace you may need, such as the standard ISO 4217 namespace for all currencies. More details on standardized units are given in Section 3.4.

Figure 2.4 shows a few examples of unit elements.

You can build product units, like square feet or kWh, by appending more `xbrli:measure` elements.

Ratio units are slightly more complex, and require inserting a `xbrli:divide` element with two children, `xbrli:unitNumerator` and, as you may already have guessed, `xbrli:unitDenominator`. The `xbrli:measure` element can then be used in each of these two grandchildren elements.

```
<xbrli:unit xmlns:xbrli="http://www.xbrl.org/2003/instance"
            xmlns:ISO4217="http://www.xbrl.org/2003/iso4217"
            id="dollars">
  <xbrli:measure>ISO4217:USD</xbrli:measure>
</xbrli:unit>
```

(a) A dollar unit

```
<xbrli:unit xmlns:xbrli="http://www.xbrl.org/2003/instance"
            id="pure">
  <xbrli:measure>xbrli:pure</xbrli:measure>
</xbrli:unit>
```

(b) A pure (dimensionless) unit

```
<xbrli:unit xmlns:xbrli="http://www.xbrl.org/2003/instance"
            xmlns:ISO4217="http://www.xbrl.org/2003/iso4217"
            id="francs-per-share">
  <xbrli:divide>

    <xbrli:unitNumerator>
      <xbrli:measure>ISO4217:CHF</xbrli:measure>
    </xbrli:unitNumerator>

    <xbrli:unitDenominator>
      <xbrli:measure>xbrli:shares</xbrli:measure>
    </xbrli:unitDenominator>

  </xbrli:divide>
</xbrli:unit>
```

(c) A more complex unit, with a numerator and a denominator

Figure 2.4: Three examples of unit elements: an ISO 4217 currency, a standard unit, and a unit with a division.

2.8.5 Concept and value

Now that the machinery for defining syntactic contexts and units has been introduced, we can move to the most important piece: elements that define facts.

Unlike `xbrli:context` and `xbrli:unit` elements, facts are defined with a dynamic element name, and this name is that of the concept. You guessed correctly: this implies that concepts are actually syntactically QNames. We will see in later chapters that they can be associated with labels to be displayed so as not to confuse the end users.

Figure 2.5 shows the syntax that describes our running example in this chapter: the assets Coca Cola reported for April 3, 2015, in U.S. dollars. As we will see in Chapter 3, concept names are defined and grouped in namespaces. The concepts used with the SEC are part of the US GAAP (Generally Accepted Accounting Principles) taxonomy, and live in the corresponding namespace[4].

For each fact, two attributes are used to reference the context and unit: `contextRef` and `unitRef`. Either the precision or the decimals can be reported – but not both – with respectively the `precision` and `decimals` attributes. The `precision` attribute has to be a positive integer, and the `decimals` attribute has to be an integer. However, both can also have the special value "INF" for infinite precision.

Finally, the language aspect, if any, is syntactically represented with an `xml:lang` attribute according to the XML specification. Figure 2.6 shows how this is done. Note that, since languages only apply to string values, `xml:lang` will never appear with `unitRef`, `decimals` or `precision`. The `xml` prefix does not need to be bound to the XML namespace, because it is builtin in XML. Figure 2.6

[4]The namespace is actually updated every year to a new version. It makes comparisons across years more complicated, but still doable. More on this in Section 3.3.