Understanding the Basic Mechanics of a Digital Financial Report

by

Charles Hoffman, CPA (charleshoffman@olywa.net) and

Timothy Randle (timothy@clientserversllc.com)

February 6, 2015

ABSTRACT:

Digital financial reports contain thousands and sometimes many thousands of details. These details can be grouped into objective mechanical aspects which can be automated using machines and subjective aspects which require the professional judgment of a skilled accountant. IT professionals creating software for business professionals need to be aware of the mechanical things and relations between things which make up a financial report in order to create software useful to business professionals. With useful software the mechanical aspects can be handled by software freeing accounting professionals to use their skills in area which are impossible to automate, areas which require the professional judgment of a skilled human.

The primary purpose of this paper is to describe the mechanical aspects of a representational model of an XBRL-based digital financial report which was gleaned by reverse-engineering publically available public company XBRL-based financial filings submitted to the U.S. Securities and Exchange Commission (SEC).

A secondary purpose of this paper is to propose that the representational model of these XBRL-based digital financial reports is a good representational model to follow because the representation model allows for the system to be decidable meaning that a conclusion can be reached as to the consistency of a digital financial report to this representational model description expected by the system.

While it is the case that every aspect of every XBRL-based financial report provided to the SEC by public companies are not 100% consistent with the representational model; enough aspects of enough of these reports point to what each report needs

to look like in order for XBRL-based digital financial reporting to provide useful functionality.

This paper is intended to be understandable to both accounting professionals who need specific functionality from an XBRL-based digital financial report and information technology professionals who know the best ways to technically deliver such functionality.

CONTRIBUTORS:

The ideas in this document are not my own, but rather an accumulation and organization of many, many other ideas gleaned from IT professionals and accounting professionals over 15 years of trying to create XBRL-based digital financial reporting. I see myself as merely a custodian of this important information, nurturing it along for the benefit of all, condensing countless discussions into something hopefully useful for the common good. I would like to specifically thank the following individuals who contributed directly or indirectly to this effort:

Walter Hamscher, PhD, Geoff Shuetrim, PhD; David vun Kannon; Rene van Egmond; Thomas Egan, CPA; Josef Macdonald, CA; Jim Richards; Roger Debreceny; Jeff Naumann, CPA; David Prather, Alan Teixeira, CA; Hugh Wallis; Allyson Ugarte; Colm O hAonghusa; Giancarlo Pellizzari; Yossef Newman, CPA; Rob Blake; Mark Creemers; Marc van Hilvoorde; Herman Fischer; Ignacio Hernandez-Ros; Dean Ritz; Timothy Randle; Cliff Binstock; David Scott Stokes; Masatomo Goto; Paul Warren; Mark Goodhand; Campbell Pryde, CPA; Michele Romanelli; Maciej Piechocki, PhD; Victor Morilla; Mike Rowling; Joe Ryba, CPA; Matthias Brantner; Dennis Knochenwefel; Ghislain Fourny, PhD; Daniel Taylor, Chris Taylor, CPA, Thomas McKinney, CPA; Eric Cohen, CPA; Mike Willis, CPA; Louis Matherne, CPA.

There are others which I probably left off and for this I apologize. I acknowledge and appreciate the efforts others contributed to this endeavor.

Introduction

An important objective when sharing or exchanging information, particularly within a distributed system, is to share or exchange that information without disputes as to the precise meaning of that shared information. A lack of discipline and rigor or a lack of formality in expressing precise meaning soon leads to arguments as to information meaning. Agreement as to meaning delivers consistent understanding.

One aspect of meaning is basic mechanics. For something like an XBRL-based digital financial report basic mechanics relates to the report itself. The world is made up of *things*. A thing is nothing more than something that exists in the real world. Real world things can be related to other real world things. In order to get a machine such as a computer to work with these real world things you have to explain the things and explain the relations between those things.

By basic mechanics of a financial report we simply mean the basic important real world things that make up a financial report and the basic important relations between those real world things, the essence of a financial report. This is done so that we can then explain how a financial report works to a machine, such as a computer, so that the machine can help us create and make use of the information contained within financial reports.

This is as contrast to aspects of a financial report which are subjective in nature and therefore require the professional judgment of a skilled accountant.

These mechanical aspects are distinct from the subjective aspects which require judgment. The mechanical aspects are objective and require no judgment. The mechanical aspects relate more to logic, common sense, and mathematics. These subjective and therefore judgmental aspects have to do with *which* things exist in the financial report, some aspects of the *values* of those things, how the values are *measured*, and so forth.

This document focuses on the mechanical aspects which are ruled by logic, sensibility, basic mathematics, and common sense. Subjective and therefore aspects which require the professional judgment of a skilled accountant are not addressed because they are not in scope.

How to achieve a meaningful exchange of information is well understood by information technology professionals who specialize in that area of information technology. Just because someone is an IT professional, however, does not necessarily make them an expert in global standard approaches to the effective exchange of information.

IT professionals who understand information exchange understand this fundamental premise: The only way a meaningful exchange of information can occur is the prior existence of agreed upon technical syntax rules, domain semantics rules, and workflow/process rules.

It is the rules which provide the coordination and therefore the control which makes the meaningful exchange of information work effectively. It is the rules which turn what could be a guessing game into a reliable, predictable, repeatable and therefore useful tool. It is the rules that form the basis for the agreement in a distributed environment such as financial reporting. These rules are a tool which are used to provide the cooperation necessary to make such a system work appropriately.

Part of that meaning relates to the types of things and of relations between things in the problem domain. Ideally we would capture all the types of entities in our domain and all the relations between these entities. As expressive power increases, however, computational complexity increases also.

As expressive power increases and therefore computational complexity increases; reasoning problems can result in unforeseen complexity-caused blowups. Expressive power must be useful-yet-harmless.

The goal is to properly balance the system with carefully chosen things and relations between thing such that typical applications with a requirement for reliable, predictable, repeatable and efficient support for information description constraints and information quality control constraints.

The best balance between expressiveness and complexity depends on the intended application. Systems which achieve this balance are both extremely useful because the information is both very rich with meaning but also of high quality because the information is consistent with expectations of what the information should look like.

It is an important goal, therefore, to find the sweet spot where we have maximum expressivity (that goes furthest towards meeting our data-collection needs) without sacrificing the quality of something which is referred to as *decidability*.

Understanding the critical importance of decidability

In order to understand critical aspects that make a system work we need to take a brief but important fork in this discussion to make the reader conscious of the

notion of decidability. To understand the notion of decidability, we must also discuss the closed world assumption. We do that here.

There are two perspectives which can be adopted when evaluating information in a system: open world assumption and closed world assumption.

In the *open world assumption* a statement cannot be assumed true on the basis of a failure to prove the statement. On a World Wide Web scale this is a useful assumption; however a consequence of this is that an inability to reach a conclusion (i.e. not decidable).

In the *closed world assumption* the opposite stance is taken: a statement is true when its negation cannot be proven; a consequence of this is that it is always decidable. In other applications this is the most appropriate approach. So each application can choose to make the open world assumption or the closed world assumption based on its needs.

Because it is important that a conclusion as to the correct mechanics of a financial report is required because consistent and correct mechanics are necessary to making effective use of the information contained within a financial report; the system used to process a financial report must make the closed world assumption.

This assumption is not new to business professionals because business professionals make use of information from many, many relational databases and relational databases make the closed world assumption when working with data.

Essentially what this means is that if the information is not within the set of information that you are directly working with, the information is assumed not to exist.

Decidability means that a conclusion can be reached. Specifically in our case, decidability means that a conclusion must always be reachable as to the correctness or incorrectness of the mechanical aspects of a financial report

Decidable means that no interpretations that are not satisfied (unsatisfied or inconsistent) by at least one interpretation of the information in the system exists. If a representation of information is not decidable then the represented information is ambiguous because you cannot determine if the information is inconsistent or simply unsatisfied which means that a conclusion cannot be reached.

At the risk of being redundant we point out again the critical distinction between the mechanical aspects of a financial report and the subjective aspects which require or judgmental by a skilled accountant. A conclusion about the correctness

or incorrectness of the mechanical aspects in no way suggests or implies that a computer will ever be able to determine the overall appropriateness of a financial report. Such determination always involves professional judgment and therefore always involves a skilled professional accountant.

Understanding the notion of things and relations between things

Different people use different terms to describe the same ideas. The Semantic Web uses the term "thing". Others use the term "entity" to describe the same idea. Different terms used by different people in the IT profession can make it hard for business professionals to understand a very important notion in the digital age.

You may have heard terms such as "entity relationship diagram" or "conceptual model" or "UML model" or "OWL ontology" or "XBRL taxonomy" or perhaps even some other term. Essentially, they are all trying to describe the very similar if not exactly the same ideas. Different approaches have different forms of expressiveness. Different approaches use different technical syntaxes to make what is created readable by machines such as computers. Essentially though, you can think of these using three simple ideas:

- **Thing**: A thing is some category of object that exists in the real world.
- **Individuals**: An individual is the actual object that exists[1].
- **Relation between things**: One thing can be related to another thing.

For example, the notion of an *accounting entity* is a **thing**. Microsoft is an **individual** which is in the set of things known as an *accounting entity*. A *financial report* is a **thing**. The fiscal year 2014 financial statement of Microsoft is an **individual**. The idea that an *accounting entity* creates a *financial report* is a **relation between things**.

A digital financial report is a finite set of structural things. The things that make up a digital financial report are related to other specific things and have no relationship what-so-ever with other things. Again, bear in mind that we are referring to the mechanical things right now.

Global standard languages, such as XBRL, can be used to describe things and relations between things. XBRL[2] is not the only global standard which can be

[1] Another term used to describe individuals is "instance", an instance of some class. Another term is "member", a member of the set of things. We are standardizing on the term individual.

used. OWL 2 DL[3] is also a global standard which can describe these things and relations between things. Other languages exist, many are machine readable.

No matter what the technical syntax, if the same real world thing is being described you would expect that the description would be the same no matter what the syntax. Intuitively, this is an easy idea to grasp. However, if you consider that technologies created by humans were created independently of each other in many cases and created differently for slightly different purposes and therefore provide different sorts of functionality; it becomes clearer why this can be confusing to a business professional or even IT professional. Trying to figure out which is the best technical syntax to use or even what the pros and cons of using one technical syntax as contrast to some other syntax can be a daunting task.

There are many global standard technical languages for describing problem domain things and the relations between things. There are two pieces that all of these global standards have: some *syntax* for expressing such information and some set of *semantics* which provides the expressive power[4] of the global standard language.

We raise this point not to evaluate one technical syntax against some other technical syntax. We raise this point simply to explain the moving pieces of the puzzle that we are trying to solve.

Understanding the importance and limitations of first-order logic

First-order logic might seem hard to understand but in reality it is very a straight forward idea. First-order logic is simply an approach or language for describing things and relations between things. Again, different languages have different syntaxes, different levels of expressive power, they are good for some things and not as good as other things. Description logics[5] are a family of representational languages. *SROIQ* Description Logic[6] is one such language which is based on a fragment of first-order logic. There are two reasons *SROIQ* Description Logic is important: (a) it is decidable, (b) OWL 2 DL and *SROIQ* Description Logic have consciously equivalent expressive power. Meaning, they were consciously and

---

[2] XBRL, see http://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html

[3] OWL 2 DL, see http://www.w3.org/TR/2012/REC-owl2-primer-20121211/

[4] See this image to help understand relative expressive power, http://www.xbrlsite.com/2014/Library/KnowledgeRepresentationLanguageExpressiveness.jpg

[5] Description Logics, see http://en.wikipedia.org/wiki/Description_logic

[6] *A Description Logic Primer* describes the importance of *SROIQ*, http://arxiv.org/pdf/1201.4089.pdf

specifically built to be equivalent for a reason and use the same fragment of first-order logic. While different syntaxes, semantically they are equivalent. While OWL and Description Logic were initially created independently, wise people realized that there are significant advantages to making them equivalent and with specific functionality[7]. The result was OWL 2 DL and *SROIQ* Description Logic which have different syntaxes but equivalent semantics.

Remember the discussion about decidability earlier? Both OWL 2 DL and *SROIQ* Description Logic where consciously created to be decidable. What is relevant here is not OWL 2 DL or *SROIQ* Description Logic. The two relevant pieces are that someone went through a lot of trouble to make these two tools equivalent and to use specific fragments of first-order logic which are decidable.

A theory describes the world and tries to describe the principles by which the world operates. A theory is simply a system of ideas which is intended to explain something, for example the things and the relations between those things. A theory is generally explained using first-order logic.

A theory is a communications tool. A theory explains, using first-order logic, a theory explains real world things and relations between those things. A theory can be right or wrong, but it is characteristic by its intent: the discovery of essence. The purpose of a theory is to correctly describe the essence of some real world problem domain.

For example, *Financial Report Semantics and Dynamics Theory*[8] is a theory that explains the mechanics of how a financial report works.

First-order logic is very powerful and can be used to express a theory which fully and categorically describes structures of a finite domain (problem domain). This is achieved by specifying the things of the problem domain and the relations between those things.

No first-order theory has the strength to describe an infinite domain. Essentially what this means is that the things and the relations between things which make up a problem domain must have distinct boundaries. They must be made finite.

This is not to say that such a system cannot be flexible. For example, a form is not flexible. A financial report is not a form. This is not to say, however, that a financial report cannot be finite.

---

[7] *From SHIQ and RDF to OWL: The Making of a Web Ontology Language* helps you understand important ideas and concepts, see http://www.cs.man.ac.uk/~horrocks/Publications/download/2003/HoPH03a.pdf

[8] *Financial Report Semantics and Dynamics Theory*, http://xbrl.squarespace.com/fin-report-sem-dyn-theory/

Understanding the notion of patterns

A pattern is a representation model or set of rules which are used to guide. Patterns are important to IT professionals. Computers can leverage patterns. Patterns are both a communications tool that can help business professionals and IT professionals communicate, functionality templates which can be leveraged to make software easier to create, and a specimen that exemplifies the ideal qualities of something.

Basically, patterns describe.

Understanding the notion of slot or opening

While a form is finite but inflexible, a financial report is finite and flexible. The difference between the two can be described using the notion of a "slot" or "opening". A form has no slots or openings. A form only has cells into which information may be placed.

A **slot** is simply the idea of an allotted place in an arrangement where something can be logically and sensibly placed.

For example, suppose you wanted to add something to a roll up of property, plant and equipment as shown below:

Property, Plant and Equipment, by Component [Line Items]	Period [Axis]	
	2010-12-31	2009-12-31
Property, Plant and Equipment, by Component [Roll Up]		
Land	1,000,000	1,000,000
Machinery and equipment, gross	2,000,000	2,000,000
Furniture and fixtures, gross	6,000,000	6,000,000
Accumulated depreciation	(1,000,000)	(1,000,000)
Property, plant and equipment, net	8,000,000	8,000,000

You cannot add a second total to a roll up as a roll up only has one total. It would not make logical sense to add a second total to a roll up. Other terms used to describe this are illogical or irrational. What makes sense is to add another line item to the total of the roll up, somewhere in the list of existing line items. One

slot is adding a line item between *Land* and *Machinery and equipment, gross*. Another slot is adding a line item before the first item *Land*.

Further, *what* you add to the list is also constrained. For example, what you add needs to be a number as a roll up involves showing how some list of numbers rolls up. You would not add text. And it cannot be just any number, it needs to be an "as of" type number (as contrast to a "for the period" number from, say, the income statement). Why? Because all of the other numbers in the list are "as of" some balance sheet date, not "for the period" of some income statement or cash flow statement period.

There is another slot which makes sense in the information above. You can see that there are two periods. Adding information for a third or even more periods makes sense. It could also make sense to add an entirely new characteristic such as Geographic Area [Axis] and break down the information by that dimension.

Basically, it makes no sense to simply add information randomly or arbitrarily to the roll up. While every slot or opening where it makes sense to add information to the existing information above has not been pointed out, the set of examples provide should help you understand the notion of a slot.

Another way to think of a slot is as a tool. Consider the definitions of arbitrary and standard:

- **Arbitrary**: based on random choice or personal whim, rather than any reason or system; depending on individual discretion (as of a judge) and not fixed by law
- **Standard**: used or accepted as normal; something established by authority, custom, or general consent as a model or example

Understanding the notion of class

A class[9] is a set or *category of things* having some property or properties in common and differentiated from other things. For example, *Assets* is one thing. *Revenue* is another thing. Something cannot be both an asset and revenue; they are different classes of things. However, at another level, Assets is a concept and Revenues is a concept. And so from the perspective of a concept, they are both of

[9] For more information on classes, see http://www.xbrlsite.com/2015/fro/us-gaap/html/Classes/

the class concept[10]. While a comprehensive discussion of the notion of a class is beyond the scope of this document, it is important for business professionals to understand the notion of a class. IT professionals should realize that the term class is being used differently than how object oriented programming (OOP) uses this term.

A financial report problem domain is made up of classes of things. In fact, a financial report is finite in the sense that it is made up of exactly the following structural pieces or things which can be grouped into the following classes:

- Economic or accounting **entity** which creates a report
- **Report** which is created by an entity which contain a set of components
- **Component** which contains or groups a sets of facts
- **Characteristics** which describe and distinguish facts contained within a component from other facts
- **Facts** which are reported and can be organized into components and described by characteristics
- **Blocks** which is a part[11] of a component, a component is made up of one to many blocks
- **Relations pattern** which can be either a "whole-part[12]" type relation, an "is-a" type relation, a concept arrangement pattern, or a business rule which describes relations; Concept characteristic-type relations pattern (called concept arrangement patterns also a type of whole-part or is-a relation) which can be a "roll up", a "roll forward", an "adjustment", or a "hierarchy"
- **Properties** of an economic/accounting entity, report, component, block, fact, characteristic, or relation pattern

[10] Just like one person may call something “data” and another may call it “metadata”, assigning classes can be subject to the perspective of the user of the class.

[11] A block is a sub-set of a component. For example, the disclosure Funding Status of Defined Benefit Plans is made up of two roll forwards, a roll up, and a hierarchy each of which is a block of the component, see http://www.xbrlsite.com/2013/ReportingTemplates/2013-05-15/Library/730000-003-FundingStatusOfDefinedBenefitPlans/Template.jpg

[12] Whole-part relations are a significant topic and beyond the scope of this document, for more information see http://xbrl.squarespace.com/journal/2015/1/20/toward-understanding-whole-part-relations.html

So the things that make up a financial report fall into that finite set of distinct and identifiable classes. Each of those classes of things has different slots or openings into which things can be added.

Individual systems can be implemented differently and so they could operate differently. Generally, a system could add additional classes. However, a system must have all of the classes in the list above. It is perhaps the case that a system might eliminate the notion of a block by specifying that a component may only contain one concept arrangement pattern.

We are looking at the U.S. Securities and Exchange EDGAR system and the XBRL-based financial filings which go into that specific system.

Realizing that creating a financial report is about creating subclasses or individuals and adding things into slots

As stated, the structural pieces or things which make up a financial report can be grouped into classes. No new classes can be added, you may only use existing classes[13]. Classes may never be redefined; you cannot arbitrarily change the meaning of a class. However, subclasses can be added and identified as being associated with one of those existing classes of things. But subclasses can only be added as specified by the system. Individuals can be created and specified as being a member of one class or another, you simple cannot create an individual which is associated with nothing or which is two things at the same time.

And so:

- **Adding new economic/accounting entities**: (Individual) An economic/accounting or reporting entity is created by creating a new instance of identifier. For example the CIK number of a public company which reports to the SEC.
- **Adding new report**: (Individual) A new report is created by creating a new report instance. For example, Microsoft submits a new financial report for fiscal year ended 2014.
- **Adding a new characteristic:** (Class and/or Individual) A new characteristic can be added but the characteristic MUST be distinguished as

[13] Individual systems can add whatever classes, relations between classes, and properties that they want. Here we are assuming the SEC EDGAR system and XBRL-based financial filings which go into that system only.

being either a "whole-part" or "is-a" type of relation or some existing subclass of existing relations (which must be one of those two). For example, Microsoft uses the existing characteristic "Legal Entity [Axis]" (which is a whole-part type relation) or Microsoft creates the characteristic "Tax Entity [Axis]" and distinguishes that characteristic as being a "whole-part" type of relation.

- **Adding new concept characteristic:** (Class and/or Individual) A new concept can be added but the concept MUST be distinguished as being a subclass of some existing concept or distinguished as being a new type of class (if that is allowed). For example, Microsoft might add a new concept to its balance sheet such as "Ultra-tangible asset"; however it MUST NOT break the rules of a "roll up" because a balance sheet is a roll up. Further, the added concept MUST be identified as a subclass of something that exists on a balance sheet which can contain ONLY assets, liabilities, or equity.

- **Adding new disclosure (component or block):** (Class and/or Individual) A disclosure is in essence a set of facts which must be disclosed. A set of facts is represented as a component and that component might have one or many blocks. To add a new disclosure, a reporting entity simply creates a new component and/or block individual. That individual of the class component MUST be (i.e. follow) the relations patterns of the existing component which the individual is a member of. For example, if Microsoft creates a "balance sheet" individual, it must associate that individual with the existing class "balance sheet" and therefore must follow the relation rules of a roll up because the existing component "balance sheet" is a roll up. Why? Because a balance sheet is a roll up, it is not ever a roll forward. Now, a reporting entity could also, if they desired, create a new subclass of "balance sheet" called "my balance sheet" and associate it with the class "balance sheet". Or, a company could create an entirely new disclosure such as "cash and cash equivalents by county", associate that disclosure not with some existing disclosure but rather with the root class "component" and then provide a completely new disclosure. However, what the reporting entity may NOT do is create some new relations pattern, it must use existing relations patterns (i.e. no new relation patterns can be added). Basically, any individual MUST follow the rules as must any new class.

- **Adding facts**: (Individual) A fact is always an individual. Facts are put into blocks which go into components. Facts are never "free floating in space". Every fact has distinguishing aspects to make them identifiable from other

reported facts. Facts are described by characteristics, exist within a report, and are reported by an economic/accounting entity. For example, the accounting entity Microsoft might report the fact 1,000,000 which relates to the consolidated entity, to the current balance sheet date of December 31, 2014, be reported in US Dollars, and report the balance sheet line item Cash and cash equivalents. That fact might be in the component balance sheet and has a relation between the concept Current assets in that it rolls up to that total.

- **Adding new properties**: New properties MUST NEVER be added, XBRL-based financial filings to the SEC does not allow the addition of new properties, there is no "slot" available where new properties may be added.

Different systems can have different rules for allowing new classes, subclasses, relations between classes, or properties. System boundaries can be extended by adding new relation patterns. New relation patterns must be consciously and formally added in a controlled and coordinated manner only by system implementers before any new pattern is allowed to be used. System boundaries can be extended by adding new classes or properties. New classes and new properties must be consciously and formally added in a controlled and coordinated manner only.

Understanding why adding new patterns is both rare and not a significant constraint

Adding new patterns is both rare and not a significant constraint. While this notion might seem absurd or unintuitive, it is important to look at empirical evidence to understand why this is the case.

If one were to observe XBRL-based financial filings, one would realize that 98% or more of public company financial reports contain [Line Items] which contain concepts and abstracts which follow these concept arrangement patterns:

- **Text block**
  - Level 1 Note Level Text Block
  - Level 2 Policy Level Text Block
  - Level 3 Disclosure Level Text Block
- **Roll Up**: Concept A + Concept B + Concept *N* = Total
- **Roll Forward**: Beginning balance + Additions – Subtractions – Ending balance

- **Hierarchy**: No mathematical relationships
- **Adjustment**: Originally stated balance + Adjustments = Restated balance
- **Roll Forward Info**: Beginning balance info + Additions info – Subtractions info – Ending balance info (there are no mathematical relations, but information for the beginning and ending balances must be distinguished)

Similarly, each [Axis] falls into one of two categories and describes the [Member]s of that [Axis] as being one of the following two member arrangement patterns:

- **Whole-part**: Characteristic describes something composed exactly of their parts and nothing else or more where the parts add up to the whole
- **Is-a**: Characteristic describes some list but the list does not add up mathematically

Consider the following theory: A combination of those *concept arrangement patterns* and *member arrangement patterns* describes every component of every report of every reporting entity which submits XBRL-based financial information to the SEC.

That theory is speculated to be true for 98% of the components of public company financial reports. Being conservative, we leave room for 2% of report components which might deviate from these rules because they are not structural patterns described in this document. Basically, the following spectrum delineates all possible alternatives:

1. A reporting entity report component follows (**is consistent with**) existing concept arrangement patterns and existing member arrangement patterns.
2. A reporting entity component DOES NOT FOLLOW, however SHOULD FOLLOW (**is inconsistent with**) existing concept arrangement patterns and existing member arrangement patterns. HOWEVER, after the inconsistency is corrected within the report, the reporting entity report component follows (**is consistent with**) existing patterns.
3. A reporting entity component DOES NOT FOLLOW, but either a concept arrangement pattern or member arrangement pattern IS MISSING from the list of allowed patterns. The pattern is logical, rational and sensible and would NOT RENDER the system NOT DECIDABLE. THEREFORE, the pattern should be added (**is consistent with**).
4. A reporting entity component DOES NOT FOLLOW, but either a concept arrangement pattern or member arrangement pattern IS MISSING from the list of allowed patterns. The pattern is logical, rational and sensible HOWEVER; the pattern (a) can be reduced down to a less complex pattern

and (b) if added it WOULD RENDER THE SYSTEM NOT DECIDABLE. THEREFORE, the pattern should NOT BE ADDED. Rather, the reporting entity should change how they report information to keep the overall system safe (**is consistent with**).

5. A reporting entity component follows (**is consistent with**) the existing [Hierarchy] concept arrangement patterns and an existing member arrangement pattern; HOWEVER the pattern is in reality not a [Hierarchy] but rather some other unsupported mathematical relation or some other unsupported member arrangement pattern. While not optimal because specific information which could be verified to be consistent is not being verified, this is still on par with current practices. Currently, a [Roll Forward] is a known and a commonly used pattern. The pattern is identifiable, but has no computation articulated.

And so either a filer is already consistent with the existing system (#1), or should be consistent with the existing system (#2). It is possible that a reporting entity is using a logical and sensible concept arrangement pattern or member arrangement pattern that is missing (#3); and if so, that pattern should be added to the system. It is possible that a reporting entity is using a logical and sensible concept arrangement pattern or member arrangement pattern; however, (a) that pattern can be broken down into a simpler, less complex pattern an (b) if the pattern were added to the system it would make the system not decidable and therefore should not be added to the system.

And, as discussed in the next section, there is always a fallback position (#5). Everything can be represented as a [Hierarchy] concept arrangement pattern. Other concept arrangement patterns simply add additional rules, generally mathematical computations. This allows new patterns to evolve. This is explained in more detail in the next section.

Understanding that pattern maintenance is an evolutionary process

Every concept arrangement pattern is some [Hierarchy][14] of concepts. Other non-[Hierarchy] concept arrangement patterns add some sort of mathematical computation. For example,

[14] I really don't like the name [Hierarchy] because everything is a hierarchy. A better term might be [Set] or some other term.

- A [Roll Up] is simply a [Hierarchy] with the addition of XBRL calculation relations which articulate the information about how the concepts roll up.
- A [Roll Forward] is simply a [Hierarchy] with the addition of a preferred label role to differentiate the beginning and ending instant concept.
- An [Adjustment] is simply a [Hierarchy] with the addition of a preferred label role to differentiate the originally stated and restated balances plus a member arrangement pattern to distinguish the Report Date [Axis].
- A [Text Block] is a [Hierarchy] which has only one concept which is of a specific data type.

Basically, any information can be represented as a [Hierarchy]. The down side of representing information in this manner if it really is some other pattern is that you do not provide metadata which software can use to assure that what is represented is consistent with reality. The information might be consistent with the knowledgebase of information, but that is only because the rules are not included in the knowledgebase. What that means is that the information needs to be verified using manual processes because consistency cannot be determined using automated processes because there are no machine-readable business rules.

This situation is not optimal, but it is also not the end of the world either. As was stated above, this situation is on par with current XBRL-based public company financial filings in that [Roll Forward] concept arrangement patterns in existing SEC filings do not provide business rules for the [Roll Forward].

What this means is that there is already a process to allow patterns to evolve.

Understanding that patterns are finite (i.e. not infinite)

To understand that it is not an overwhelming task to inventory all patterns and add new patterns to the system, consider the notion of report frame patterns[15]. If you look closely at the report frame patterns, this is what you observe:

- Every public company can be grouped into one of 95 report frames.
- Of the approximately 8,000 reporting entities in scope (funds and trusts are excluded as they follow other patterns which are not of interest); 90% of all public companies fall into one of 13 report frames. The remaining 10% of reporting entities use the other 82 report frames.

[15] For more information on report frame patterns, see http://www.xbrlsite.com/2015/fro/us-gaap/html/ReportFrames/

- Some of the report frames which are used are likely illegal. For example, why would a commercial and industrial company report using an unclassified balance sheet? Meaning, some existing report frames need to be removed.
- It is highly-likely that some report frames will have only 1 reporting entity, for example JPMorgan seems to fall into that category. Nothing wrong with that.
- It is highly-likely that there are between perhaps 100 to maybe even 250 additional report frames. It is of no consequence to have 100 or even 250 additional report frames.

Every other class works precisely the same way. Some finite list of subclasses can exist. And so, the system is finite, the system has boundaries, but the system is flexible but only where specific flexibility is exposed.

Understanding technical syntax rules and workflow/process rules

There has not really been much emphasis on technical syntax rules and workflow/process rules, the primary focus is on business domain semantic rules.

The reason for less effort in explaining technical syntax rules is because of the following:

- XBRL technical syntax rules were created and interoperability between software is excellent due to a publically available conformance suite provided by XBRL International.
- Because of the first point; XBRL-based digital financial reports provided to the SEC by public companies are 99.9% consistent with the XBRL technical syntax rules.
- Business professionals should never be exposed to technical syntax; software should hide all aspects of technical syntax from business professionals.

Basic workflow/process rules are worth covering a little because that would yield important useful information. However, there has not been a lot of focus on workflow/process rules so we really don't know the full extent of what workflow/process rules are necessary.

However, we do understand the basic, fundamental rules which are necessary for any system to work with a digital financial report.

Consider a simple query of two concepts: *Assets* and *Liabilities and Equity*. In order to extract that information from any XBRL-based financial filing using a machine-based process the following process needs to be followed:

1. Software MUST locate each report you want to query.
2. The report MUST be valid XBRL technical syntax. If the technical syntax is invalid, you may or may not get the correct results.
3. Software MUST locate the appropriate reporting units (currency). In the case of public company financial reports, 99% of entities report using US Dollars. However, 1% use other currencies as the reporting units.
4. Software MUST appropriately identify the root reporting entity in the report. Generally, this is the consolidated entity but it could be a parent holding company or some other accounting entity.
5. Software MUST appropriately locate the current balance sheet date. Generally you want information about the current balance sheet data and not the prior balance sheet.
6. Software MUST find the appropriate US GAAP concept used to express *Assets* which is us-gaap:Assets.
7. Software MUST find appropriate US GAAP concept for *Liabilities and Equity*. This is a little harder because there are multiple possible concepts: us-gaap:LiabilitiesAndStockholdersEquity or us-gaap:LiabilitiesAndPartnersCapital.
8. Software MUST check the returned information to assure that it is consistent with what is expected, the business domain rule that "Assets = Liabilities and Equity".

That is an overview of the workflow/process to obtain a basic set of information from the knowledgebase of XBRL-based public company financial filings. And here are the results of that query for every financial report in that data set:

xbrl:Entity	Legal Entity	Fiscal Period	Fiscal Year	Assets	Liabilities and Equity	Units	Difference in Value
All CIK numbers	Root economic entity	FY	2001	280	280	iso4217:USD	0
All CIK numbers	Root economic entity	FY	2009	31,586,555,000	31,586,555,000	iso4217:USD	0
All CIK numbers	Root economic entity	FY	2010	23,061,516,000	23,061,516,000	iso4217:CAD	0
All CIK numbers	Root economic entity	FY	2010	8,833,200,000	8,833,200,000	iso4217:GBP	0
All CIK numbers	Root economic entity	FY	2010	33,205,444,569,755	33,235,543,477,631	iso4217:USD	30,098,907,876
All CIK numbers	Root economic entity	FY	2011	45,216,467	45,216,467	iso4217:AUD	0
All CIK numbers	Root economic entity	FY	2011	110,885,000	110,885,000	iso4217:BRL	0
All CIK numbers	Root economic entity	FY	2011	28,708,716,218	28,708,716,218	iso4217:CAD	0
All CIK numbers	Root economic entity	FY	2011	1,226,733,000	1,226,733,000	iso4217:EUR	0
All CIK numbers	Root economic entity	FY	2011	7,938,800,000	7,938,800,000	iso4217:GBP	0
All CIK numbers	Root economic entity	FY	2011	1,565,000	1,565,000	iso4217:ILS	0
All CIK numbers	Root economic entity	FY	2011	46,395,324,314,234	46,165,763,878,111	iso4217:USD	(229,560,436,123)
All CIK numbers	Root economic entity	FY	2012	49,066,850	49,066,850	iso4217:AUD	0
All CIK numbers	Root economic entity	FY	2012	32,470,161,238	32,470,161,238	iso4217:CAD	0
All CIK numbers	Root economic entity	FY	2012	1,303,349,000	1,303,349,000	iso4217:EUR	0
All CIK numbers	Root economic entity	FY	2012	10,504,300,000	10,504,300,000	iso4217:GBP	0
All CIK numbers	Root economic entity	FY	2012	47,493,211,088,244	47,307,285,874,940	iso4217:USD	(185,925,213,304)
All CIK numbers	Root economic entity	FY	2013	54,642,443	54,642,443	iso4217:AUD	0
All CIK numbers	Root economic entity	FY	2013	39,919,462,935	39,919,385,738	iso4217:CAD	(77,197)
All CIK numbers	Root economic entity	FY	2013	13,120,000	13,120,000	iso4217:EUR	0
All CIK numbers	Root economic entity	FY	2013	48,909,115,040,682	48,735,740,980,605	iso4217:USD	(173,374,060,077)
All CIK numbers	Root economic entity	FY	2014	342,493,649,881	342,493,649,881	iso4217:USD	0
				176,531,415,952,227	175,972,655,073,402		(558,760,878,825)
							-0.3%

The results[16] show that most of the balance sheets balance, Assets = Liabilities and Equity. Some are inconsistent with what you would expect. The total inconsistency is .3% which is not too bad. However, the information needs to be 100% consistent in order to not get humans involved to figure out what is causing the inconsistencies.

Proving the structural mechanics using XBRL-based public company financial filings

How can you tell if the mechanics that this paper describes is correct? It is actually rather easy: look at publically available XBRL-based financial filings which public companies report to the SEC. First though, you need to reconcile the mechanical representation with an implementation of the mechanical representation in software[17]. While it is beyond the scope to do a detailed reconciliation between the semantics use in this document, the terms use by software and the US GAAP XBRL Taxonomy and SEC, and the XBRL technical syntax specification; it is necessary to

[16] Query and results provided by SECXBRL.info which is a commercial software application, see http://app.secxbrl.info/

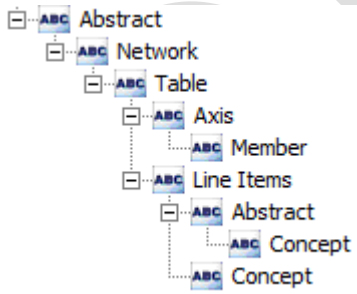
[17] Reconciliation of Financial Report Semantics and Dynamics Theory, to US GAAP XBRL Taxonomy Architecture and SEC implementation, to XBRL technical syntax, see http://www.xbrlsite.com/2012/Library/SemanticObjectIReconciliation.pdf

provide an overview because we need to shift terminology slightly. This is that overview which reconciles terminology:

Term used in this document	**Term used by software**
Economic or accounting entity	Reporting Entity CIK (XBRL context entity identifier)
Report	XBRL instance document + XBRL taxonomy
Component	XBRL **Network** + [**Table**]
Characteristic (other than concept)	[**Axis**] + [**Member**]
Characteristic (concept)	[**Line Items**] + **Concept**
Fact	Fact
Block	XBRL Network + [**Table**] + [**Abstract**]
Relations pattern	NOT IN SCOPE
Properties	NOT IN SCOPE

That is a rough explanation of the terms we use to describe the mechanics of a financial report and terms use by software applications, SEC filings, and the US GAAP XBRL Taxonomy. A complete reconciliation of terminology is beyond the scope of this document and would cause more confusion and complexity that most business professionals would tolerate.

To keep this simple, the implementation of the mechanics can be distilled down to the following classes of report elements: Network, Table, Axis, Member, Line Items, Abstract and Concept. They are roughly related as follows:



Software was used to query the mechanical structure of 6,674 XBRL-based public company 10-K filings for primarily fiscal year 2013 and the following results were obtained:

		Parent						
		Network 477,041	Table 232,230	Axis 386,912	Member 1,216,391	LineItems 232,690	Abstract 732,409	Concept 3,165,249
Child	Network	0	0	0	0	0	0	0
	Table	1,261	1	0	0	45	230,899	24
	Axis	1	386,888	0	0	3	20	0
	Member	3	0	450,091	766,221	4	72	0
	LineItems	183	232,181	0	0	107	217	2
	Abstract	474,310	22	0	1	113,059	144,471	546
	Concept	46	26	11	137	1,222,427	1,929,257	13,346

In the columns are the mechanical class of pieces which serve as the parent for some child mechanical class of piece: Network, Table, Axis, and so on. In the rows are the child mechinacal pieces: Network, Table, Axis, and so on. The cells show the number of relations which exist in the set of 6,674 digital financial reports.

This second graphic of the same information will help you to interpret and understand the results:

		Parent						
		Network 477,041	Table 232,230	Axis 386,912	Member 1,216,391	LineItems 232,690	Abstract 732,409	Concept 3,165,249
Child	Network	0	0	0	0	0	0	0
	Table	1,261	1	0	0	45	230,899	24
	Axis	1	386,888	0	0	3	20	0
	Member	3	0	450,091	766,221	4	72	0
	LineItems	183	232,181	0	0	107	217	2
	Abstract	474,310	22	0	1	113,059	144,471	546
	Concept	46	26	11	137	1,222,427	1,929,257	13,346

What the graphic says about the reltionships between the structural pieces of the digital financial reports is the following:

- Of the 386,912 [Axis] which exist in the report, there are ZERO occasions where a parent [Axis] has a child [Axis].
- Of the 232,690 [Line Items] which exist in the report, there are 1,222,427 occasions where the parent [Line Items] has a child which is a Concept.

Without going into a lot of detail, the following graphic shows what the above graphic means: the allowed and disallowed relations between the mechanical building blocks: Network, Table, Axis, Member, LineItems, Abstract and Concept.

		Parent						
		Network	Table	Axis	Member	LineItems	Abstract	Concept
Child	Network	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL
	Table	OK	Disallowed	Disallowed	Disallowed	Disallowed	OK	Disallowed
	Axis	Disallowed	OK	Disallowed	Disallowed	Disallowed	Disallowed	Disallowed
	Member	Disallowed	Disallowed	OK	OK	Disallowed	Disallowed	Disallowed
	LineItems	Disallowed	OK	Disallowed	Disallowed	Disallowed	Disallowed	Disallowed
	Abstract	OK	Disallowed	Disallowed	Disallowed	OK	OK	Not advised
	Concept	Not advised	Disallowed	Disallowed	Disallowed	OK	OK	Not advised

The point here is not to have a debate about what should be allowed and what should not be allowed. While that debate and perhaps even a theoretical or philosophical discussion about the merits of allowing or disallowing relations could prove useful, that is not the point.

The point is this: First, if a profound majority of XBRL-based financial reports are represented in a certain way, it is very difficult to say that the approach is wrong. Not impossible because the majority could be incorrect in certain occasions.

But second, and most importantly, if rules can be created and enforced by software and it is possible to have 100% agreement then why is that not done?

Look at the graphic again. Notice that there are ZERO occasions where a Network is a child of any other mechanical structure. Why is that? The reason that there are ZERO is that the XBRL technical specification states that such relations are not allowed, and the XBRL consistency suite tests to make sure software does not make this mechanical mistake.

And so an obvious question is this: why are not other mechanical aspects not enforced in this manner?

Proving other mechanics using XBRL-based public company financial filings

The following is a summary of the consistency of other mechanics of XBRL-based public company 10-K financial filings from the same set of 6,674 filings for FY 2013, an earlier set of similar 10-K financial reports for FY 2012, and for another similar set of 10-Q and 10-K financial filings for FY 2014[18].

#	Goal or Desired State	Process tests	FY 2014	FY 2013	FY 2012
1	Consistent XBRL technical syntax	Automated XBRL technical syntax error checks	99.9%	99.9%	99.9%
2	Consistent EDGAR Filer Manual (EFM) syntax/semantics	Automated EFM syntax and semantics error checks	98.0%	97.9%	80.5%
3	Consistent report level structure	Automated model structure error checks	97.6%	95.8%	97.9%
4	Detectable economic entity or accounting entity or "root reporting entity" or "entity of focus"	Successful and unambiguous identification of the "entity of focus"	99.3%	99.2%	98.8%
5	Detectable and unambiguous current period balance sheet and income statement period dates	Successful and unambiguous identification of the current balance sheet date and income statement period	98.4%	99.3%	99.8%
6	Detectable and unambiguous set of fundamental reported facts and intact relations between those fundamental facts which prove trustworthy nature of information	Automated verification checks to be sure fundamental accounting concepts are distinguishable/decipherable and the relations between those fundamental concepts are intact/sound	97.9%	97.8%	97.9%
7	Detectable basic primary financial statement roll up computations are intact which prove trustworthy nature of information	Automated verification checks for existence of business rules which articulate these basic primary financial statement relations and successful passing of these business rules	90.7%	90.1%	84.9%

The primary point here is that if you look at the columns on the right for FY 2014, FY 2013, and FY 2012 you notice that testing against what we would expect yielded

[18] Not all FY 2014 financial filings have been submitted to the SEC as of the date of this document, so the latest 10-Q was used if the 10-K was not available.

a very high number of XBRL-based public company financial reports that are consistent with those expectations.

Distinguishing between a component and a block

Because distinguishing a component and a block can be a little tricky, we wanted to provide some additional detail and examples which help make this idea more understandable. Consider the following financial report disclosure represented using XBRL:

Property, Plant and Equipment, by Component [Line Items]	Period [Axis]	
	2010-12-31	2009-12-31
Property, Plant and Equipment, by Component [Roll Up]		
Land	1,000,000	1,000,000
Machinery and equipment, gross	2,000,000	2,000,000
Furniture and fixtures, gross	6,000,000	6,000,000
Accumulated depreciation	(1,000,000)	(1,000,000)
Property, plant and equipment, net	8,000,000	8,000,000

That disclosure is a roll up of the components that make up property, plant, and equipment, net. Basically there is a one-to-one correlation between the concept arrangement pattern (i.e. roll up) and the component.

Similarly, the following component contains one disclosure in one component:

Schedule of Accrued Liabilities [Line Items]	Period [Axis]		
	2013-01-01 - 2013-12-31	2012-01-01 - 2012-12-31	2011-01-01 - 2011-12-31
Balance at beginning of period	26,987,000	12,742,000	8,972,000
Acquisition			3,151,000
Deferral of new extended warranty revenue	20,191,000	22,344,000	8,659,000
Recognition of extended warranty deferred revenue	(12,789,000)	(8,099,000)	(8,040,000)
Balance at end of period	34,389,000	26,987,000	12,742,000

Again, there is a one-to-one correlation between the component and the concept arrangement pattern (i.e. this time a roll forward).

But now consider the component below. In that component you see one component but you see two concept arrangement patterns: a roll forward and then a roll up:

Restructuring Cost and Reserve [Line Items]	Period [Axis]					
	2010-01-01 - 2010-12-31			2009-01-01 - 2009-12-31		
	Restructuring Type [Axis]			Restructuring Type [Axis]		
	Facility Closing [Member]	Severance [Member]	All Restructuring Types [Domain]	Facility Closing [Member]	Severance [Member]	All Restructuring Types [Domain]
Restructuring Reserve [Roll Forward]						
Restructuring reserve, beginning balance	97,000,000	204,000,000	301,000,000	94,000,000	200,000,000	294,000,000
Restructuring charge	(1,000,000)	0	(1,000,000)	(4,000,000)	(4,000,000)	(8,000,000)
Cash payments	(4,000,000)	(4,000,000)	(8,000,000)	(6,000,000)	(6,000,000)	(12,000,000)
Accrual adjustment	0	(1,000,000)	(1,000,000)	(1,000,000)	0	(1,000,000)
Translation adjustment	30,000,000	5,000,000	35,000,000	14,000,000	14,000,000	28,000,000
Restructuring reserve, ending balance	122,000,000	204,000,000	326,000,000	97,000,000	204,000,000	301,000,000
Restructuring Reserve [Roll Up]						
Current portion of restructuring reserve	96,000,000	204,000,000	300,000,000	96,000,000	204,000,000	300,000,000
Long-term portion of restructuring reserve	26,000,000	0	26,000,000	1,000,000	0	1,000,000
Restructuring reserve	122,000,000	204,000,000	326,000,000	97,000,000	204,000,000	301,000,000

In order to maintain a one-to-one correlation between a piece of the report and the concept arrangement pattern used to represent the piece of the report, the notion of the *block* is used.

By thinking of the one component as two blocks, each with a one-to-one relation between the represented information and the concept arrangement pattern, software can help business professionals using and creating the information in many ways.

Accountants have the option of combining information in different ways when they want to present their disclosures. But they have far fewer options when it comes to representing the information in logical, sensible, and mathematically correct ways.

Not understanding the information makes it harder to create and harder to use the information.

Consider the component taken from an XBRL-based public company financial filing submitted to the SEC below. The component contains six different blocks of information: one hierarchy and five roll ups. But it is harder to understand the information because the pieces are not separated.

Software can create the separations for business users making use of the information within a component. Different disclosures can be identified by their structural signatures. A roll up always has (or always should have) XBRL calculation relations expressed. A roll forward always has an XBRL preferred label role for the start date and another for the end date of the roll forward. These structural signatures can be used by software to help business users making use of reported information. The more creators of information help the software, the better the experience software can provide to business users.

Commitments (Details) (USD $) In Millions, unless otherwise specified	12 Months Ended		
	Oct. 31, 2012	Oct. 31, 2011	Oct. 31, 2010
Commitments			
Rent expense	$ 1,012	$ 1,042	$ 1,062
Sublease rental income	37	38	46
Property under capital lease	882	577	
Accumulated depreciation on property under capital lease	453	454	
Minimum lease payments, sublease rental income			
Minimum lease payments, 2013	780		
Minimum lease payments, 2014	665		
Minimum lease payments, 2015	517		
Minimum lease payments, 2016	351		
Minimum lease payments, 2017	218		
Minimum lease payments, thereafter	805		
Minimum lease payments, total	3,336		
Less: Sublease rental income, 2013	(28)		
Less: Sublease rental income, 2014	(23)		
Less: Sublease rental income, 2015	(18)		
Less: Sublease rental income, 2016	(9)		
Less: Sublease rental income, 2017	(4)		
Less: Sublease rental income, thereafter	(12)		
Sublease rental income, total	(94)		
Minimum lease payments net of sublease rental income, 2013	752		
Minimum lease payments net of sublease rental income, 2014	642		
Minimum lease payments net of sublease rental income, 2015	499		
Minimum lease payments net of sublease rental income, 2016	342		
Minimum lease payments net of sublease rental income, 2017	214		
Minimum lease payments net of sublease rental income, thereafter	793		
Minimum lease payments net of sublease rental income, total	3,242		
Capital lease commitments			
Capital lease commitments, 2013	59		
Capital lease commitments, 2014	240		
Capital lease commitments, 2015	11		
Capital lease commitments, 2016	7		
Capital lease commitments, 2017	4		
Capital lease commitments, thereafter	33		
Capital lease commitments, total	354		
Less: Interest payments, 2013	(8)		
Less: Interest payments, 2014	(6)		
Less: Interest payments, 2015	(3)		
Less: Interest payments, 2016	(2)		
Less: Interest payments, 2017	(2)		
Less Interest payments, thereafter	(12)		
Interest payments, total	(33)		

Here is one final example. Below you see four blocks: the first two are [Roll Forward]s, the third a [Roll Up], and the fourth a [Hierarchy]. The two [Roll Forward]s are connected to the [Roll Up], the ending balances of the [Roll Forward]s are the items which are being rolled up in the [Roll Up]. Because the information is represented correctly and because the rendering engine which produced the renderings from the machine-readable representation, the information is easy to understand.

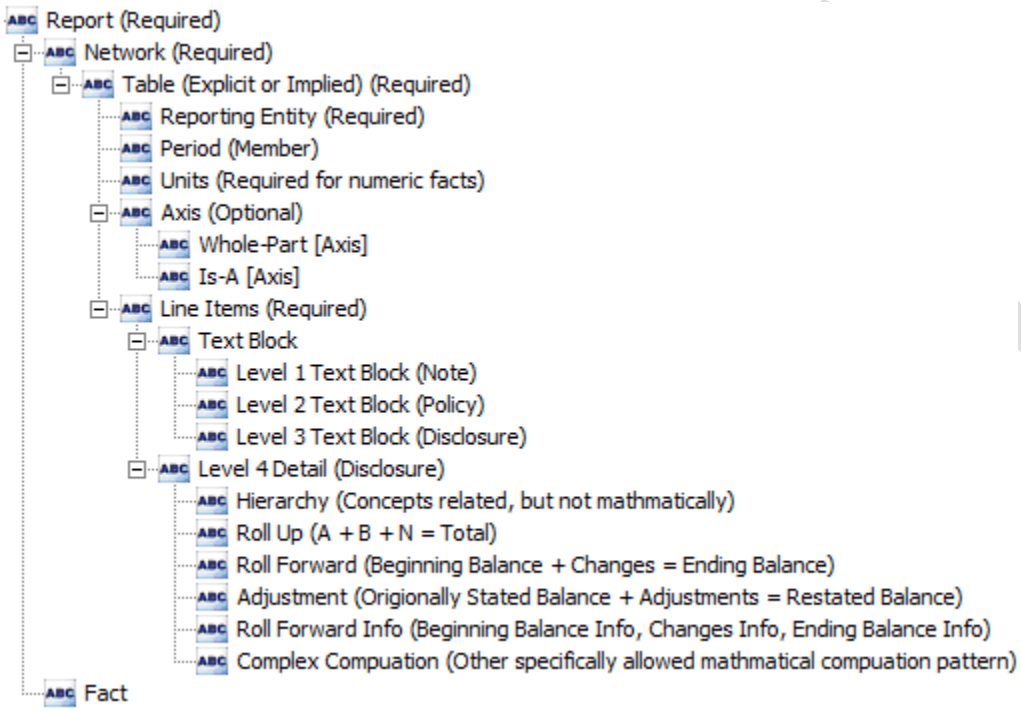
In addition to the concept arrangement patterns which show the organization of the [Line Items] (which are in the rows on the left of the rendering), the information is further distinguished using the *Defined Benefit Plan Category [Axis]*.

A block is a combination of a *concept arrangement pattern* and *member arrangement patterns* which work together to distinguish reported facts.

	Period [Axis]					
	2011-01-01 - 2011-12-31			2010-01-01 - 2010-12-31		
	Defined Benefit Plan Category [Axis]			Defined Benefit Plan Category [Axis]		
Defined Benefit Plan Disclosure [Line Items]	U.S. Pension Benefits [Member]	Non-U.S. Pension Benefits [Member]	Other Postretirement Benefits [Member]	U.S. Pension Benefits [Member]	Non-U.S. Pension Benefits [Member]	Other Postretirement Benefits [Member]
Change in benefit obligation [Roll Forward]						
Benefit obligation at beginning of year	444,000,000	593,000,000	166,000,000	375,000,000	327,000,000	157,000,000
Service cost	38,000,000	9,000,000	8,000,000	32,000,000	8,000,000	10,000,000
Interest cost	21,000,000	33,000,000	8,000,000	22,000,000	26,000,000	9,000,000
Actuarial loss	43,000,000	25,000,000	28,000,000	31,000,000	4,000,000	10,000,000
Benefits paid	(19,000,000)	(16,000,000)	(14,000,000)	(47,000,000)	(12,000,000)	(15,000,000)
Curtailment	0	(4,000,000)	0	0	(1,000,000)	0
Acquisitions of businesses	0	2,000,000	0	34,000,000	253,000,000	27,000,000
Plan amendments	0	0	0	0	0	(32,000,000)
Other changes	(3,000,000)	1,000,000	0	(3,000,000)	2,000,000	0
Exchange rate adjustments	0	0	0	0	(14,000,000)	0
Benefit obligation at end of year	524,000,000	643,000,000	196,000,000	444,000,000	593,000,000	166,000,000
Change in plan assets [Roll Forward]						
Fair value of plan assets at beginning of year	416,000,000	474,000,000	0	346,000,000	248,000,000	0
Actual return on plan assets	(5,000,000)	38,000,000	0	48,000,000	36,000,000	0
Employer contributions	43,000,000	28,000,000	14,000,000	72,000,000	52,000,000	15,000,000
Acquisitions of businesses	0	0	0	0	160,000,000	0
Administration expenses	(2,000,000)	1,000,000	0	(3,000,000)	1,000,000	0
Exchange rate adjustments	0	1,000,000	0	0	(11,000,000)	0
Fair value of plan assets at end of year	433,000,000	526,000,000	0	416,000,000	474,000,000	0
Funding Status [Roll Up]						
Fair value of plan assets	433,000,000	526,000,000	0	416,000,000	474,000,000	0
Benefit obligation	524,000,000	643,000,000	196,000,000	444,000,000	593,000,000	166,000,000
Funded status - underfunded at end of year	(91,000,000)	(117,000,000)	(196,000,000)	(28,000,000)	(119,000,000)	(166,000,000)
Accumulated Benefit Obligation [Hierarchy]						
Accumulated benefit obligation	491,000,000	616,000,000	196,000,000	421,000,000	553,000,000	166,000,000

Summary of the complete representation model and mechanics

To tie all of the pieces together, we provide this summary of the representation model and an overview of the mechanical aspects of a financial report. The graphic below shows each of the implementation pieces which can be different depending upon how a software application exposes the pieces of a digital financial report to its business users. This is a summary of the pieces of a financial report.



The table below summarizes the pieces that exist in the 10-K financial information of 6,674 public companies who report to the SEC using the XBRL format. The class of report piece, a count of the individuals in those reports, an average for many of the pieces and a brief comment is provided:

Class	Count	Average per Report	Comment
Report	6,674	1	Facts required to exist in Report
Network	477,041	71	Part of Component
Table	232,230	35	Part of Component
Axis	386,912	58	Part of Characteristic
Member	1,216,391	181	Part of Characteristic
Line Items	232,690	35	Type of [Axis], subclass of Characteristic
Abstract	732,409	111	No meaning, only used for organization
Concept	3,165,249	474	Part of Characteristic
Properties	Not counted		Each class has different but finite properties

Fact	8,532,275	1,278	Described by Characteristic, Required to exist within Network, Required to exist within explicit or implied Table
Text Block	398,492	59	Counted facts with data type of nonnum:textBlockItemType
Roll Forward	48,960		Counted preferred label roles which had start date and end date (approximate)
Roll Forward Info	18,794		Counted preferred label roles which had start date and end date but data type was not monetary (approximate)
Roll Up	114,584		Counted XBRL calculation relation roots
Hierarchy			Counted Networks with no matching XBRL calculation and no start date/end date preferred label role (work in progress)
Whole-part			Count specific [Axis] types (work in progress)
Is-a			Count specific [Axis] types (work in progress)

Taking this one step further, this provides lists of the next level of the digital financial report, the classes of text blocks, disclosures, characteristics, etc:

Class	**Comment**
Axis (need to break this out by whole-part and is-a type relations)	http://www.xbrlsite.com/2015/fro/us-gaap/html/Classes/Axes_Tree.html
Level 1 Note Level Text Blocks	http://www.xbrlsite.com/2015/fro/us-gaap/html/Classes/Level1TextBlock_Tree.html
Level 2 Policy Level Text Blocks	http://www.xbrlsite.com/2015/fro/us-gaap/html/Classes/Level2TextBlock_Tree.html
Level 3 Disclosure Level Text Blocks	http://www.xbrlsite.com/2015/fro/us-gaap/html/Classes/Level3TextBlock_Tree.html
Hierarchy	http://www.xbrlsite.com/LinkedData/Exemplars/Disclosures.aspx?InformationModel=[Hierarchy]
Roll Up	http://www.xbrlsite.com/LinkedData/Exemplars/Disclosures.aspx?InformationModel=[Roll Up]
Roll Forward	http://www.xbrlsite.com/LinkedData/Exemplars/Disclosures.aspx?InformationModel=[Roll Forward]
Report	http://www.sec.gov/Archives/edgar/monthly/xbrlrss-2014-12.xml

Expanding base mechanics, advanced mechanics articulated by the Financial Report Ontology

In order to explore the idea of consistent mechanics of a digital financial report, we used a base subset of the things and relations between things that one would find in a financial report. The purpose of using this base is to both reduce complexity of trying to explain these mechanics and to avoid debates by focusing on easy to distinguish things and relations and where high percentages of XBRL-based public

company financial reports submitted to the SEC are consistent with those mechanics. If someone looks at the facts, these mechanical aspects are self-evident.

But these basic mechanical aspects of a financial report form only the base or foundation of a digital financial report.

The *Financial Report Ontology*[19] builds on that base.

The *Financial Report Ontology* is nothing more than a set of things and relations between things. It is basically a set of business rules which describe how a digital financial report works. The ontology is expressed in machine-readable terms.

Article 9 of *The Business Rules Manifesto*[20] states that business rules are: "Of, By, and For Business People, Not IT People". Article 9 further details what it means with the following three sub points:

- 9.1. Rules should arise from knowledgeable business people.
- 9.2. Business people should have tools available to help them formulate, validate, and manage rules.
- 9.3. Business people should have tools available to help them verify business rules against each other for consistency.

Business professionals understand their domains. Accounting professionals understand the domain of financial reporting. Business rules both *describe* the business domain rules, the semantics are IT professionals call them, of a business domain such as financial reporting and serve as the *quality control* mechanism that assures financial reports created are consistent with that description.

There is a direct relation between the description and quality control. In fact, description and quality control are two different sides of exactly the same coin. What we stated earlier in this document is worth repeating:

The only way a meaningful exchange of information can occur is the prior existence of agreed upon technical syntax rules, domain semantics rules, and workflow/process rules.

The *Financial Report Ontology* is simply additional helpful rules. The more business rules there are, the more software can do to help business and accounting professionals.

---

[19] Financial Report Ontology, http://xbrl.squarespace.com/financial-report-ontology/
[20] Business Rules Manifesto, http://www.businessrulesgroup.org/brmanifesto.htm

Conclusion

We explained the mechanics of a digital financial report and showed that extremely high levels XBRL-based public company financial reports filed with the U.S. Securities and Exchange Commission are consistent with these mechanics.

We point out that these basic mechanics are finite and provide the necessary boundaries to allow for the system to be completely described by these basic mechanics using a fragment of first-order logic which is decidable.

By sticking to these basic mechanics digital financial reports can achieve the important criteria of being able to conclude if the mechanical aspects of the digital financial report are consistent with the description of the mechanics of a financial report or inconsistent. The reason that this is necessary is to be able to write software to assure that the mechanics of such digital financial reports are consistent.

Digital financial reports contain thousands and sometimes many thousands of individual pieces or structures. These structures, commonly formatted in machine-readable form using XBRL, are used to represent the information contained in the digital financial report. There are two distinct aspects of these pieces or structures that are important to recognize:

- **objective aspects** which are mechanical and do not require judgment and therefore can be managed using automated machine-based processes.
- **subjective aspects** which require the professional judgment of a skilled accountant, therefore they must be managed by humans.

These objective mechanical aspects are distinct from the subjective aspects which require professional judgment. The mechanical aspects relate to the things and relations between the things in a digital financial report. These mechanical aspects are governed by logic, common sense, and the rules of math. These mechanical aspects are what make up the structure of a financial report.

IT professionals creating software for business professionals need to be aware of the mechanical things and relations between things which make up a financial report in order to create software useful to business professionals. With useful software the mechanical aspects can be handled by software freeing accounting professionals to use their skills in area which are impossible to automate, areas which require the professional judgment of a skilled human.

Is the purpose for each individual to dig their heels into the ground and insist that their arbitrary reality is the only reality? Or is the purpose to consciously create a coordinated, shared, commonly accepted, standard, useful view of reality to achieve a specific purpose: so that reality does appear to be objective and stable enough yet nuanced enough to be useful so that information can be used safely, reliably, predictably, repeatedly by both human and automated machine-based processes. The desired system state is one of balance or equilibrium; of consistency.

Prudence dictates that using financial information from a digital financial report not be a guessing game. It is only through conscious effort that the specific control mechanisms can be put in place to realize this intent.

It is only through conscious collaboration, cooperation and coordination by the participants of the financial reporting supply chain that that XBRL-based digital financial reporting will work safely, reliably, predictably, repeatedly, effectively, and efficiently. That is the goal.

Empowered by this goal and with the intension of achieving this goal; the intelligent and wise direction of those who brought OWL 2 DL and *SROIQ* Description Logic (fragment of first-order logic which is decidable) together should be emulated.

Conscious and skillful execution using this approach can create digital financial reporting which is simple and elegant; and yet a sophisticated and powerful tool.