Putting the Expertise into an XBRL-based Knowledge Based System for Creating Financial Reports

November 16, 2017

Charles Hoffman, CPA (Charles.Hoffman@me.com)

Hamed Mousavi (hamedmousavi@yahoo.com)

Applications that are generally used to create external financial reports today, such as Microsoft Word, understand nothing about a financial report. Word understands things like paragraphs, table columns, table rows, etc. The term "balance sheet" means nothing to Microsoft Word.

What if it were possible to get a software application to understand financial reports? What if the software application you used to create financial reports could understand what a balance sheet is, what goes on a balance sheet, if certain specific line items are reported on the balance sheet that specific disclosures are also required to be reported, and so forth?

Well, it is possible for software applications to understand financial reports. A knowledge-based system[1] is a set of computer programs that use machine-readable knowledge that are built to mimic human behavior and knowledge. Another term used to describe such a system is "expert system"[2].

XBRL is a technology that lets you structure financial reports in a form that is readable by computer software applications, such as knowledge based systems[3]. You can address each and every piece of a report explicitly because each and every piece is identified. These structured digital financial reports[4] make possible things that have never been possible before because the information contained in financial reports of the past have been unstructured and therefore not readable by machine-based processes.

[1] Wikipedia, *Knowledge-based Systems*, retrieved June 23, 2017, https://en.wikipedia.org/wiki/Knowledge-based_systems

[2] Charles Hoffman, *Comprehensive Introduction to Expert Systems*, http://xbrlsite.azurewebsites.net/2017/IntelligentDigitalFinancialReporting/Part01_Chapter02.6_ComprehensiveIntroductionToExpertSystems.pdf

[3] YouTube, *How XBRL Works*, retrieved June 23, 2017, https://www.youtube.com/watch?v=nATJBPOiTxM

[4] Charles Hoffman, *Conceptual Overview of an XBRL-based Structured Digital Financial Report*, http://xbrlsite.azurewebsites.net/2017/IntelligentDigitalFinancialReporting/Part01_Chapter02.1_ConceptualOverviewOfDigitalFinancialReporting.pdf

So how do you get a knowledge-based system to understand a financial report and help professional accountants create these reports better, faster, and/or cheaper than the old-school processes that have been used to create reports[5] in the past?

This document describes a commercial quality working proof of concept that we call *Pesseract*[6] that was created to answer that specific question. For such a system to work, that system must be able to perform specific tasks and be approachable by the professional accountants using the application. In creating the application we started with the tasks that needed to be performed to successfully create a financial report and then worked backwards to design the nuts and bolts that make the system actually work.

Our approach to doing this is explained in this document in terms that a professional accountant can understand and relate to. This working application itself proves the feasibility of such a product. You can see this proof of concept in action in videos[7] or you can download[8] and use the application yourself and see how it works.

Knowledge based systems

Expert systems[9] is a branch of *artificial intelligence*. Expert systems, also called knowledge-based systems or logic systems or simply knowledge systems, are computer systems. Knowledge systems are **computer programs that are built to mimic human behavior and knowledge**. A computer application that performs some task that would otherwise be performed by a human expert is an expert system. In his book, *Systematic Introduction to Expert Systems*[10], by Frank Puppe points out that there are three general categories of expert systems (we will use the term knowledge based system):

- **Classification or diagnosis type**: helps users of the system select from a set of given alternatives.
- **Construction type**: helps users of the system assemble something from given primitive components.
- **Simulation type**: helps users of the system understand how some model reacts to certain inputs.

Knowledge based systems are for reconstructing the expertise and reasoning capabilities of qualified subject matter experts within some specific, very limited, narrow domain of knowledge in machine-readable form. A model of the expertise of the domain of knowledge of the best practitioners is

---

[5] Charles Hoffman, *Changing Old School Financial Report Creation Processes*, http://xbrl.squarespace.com/journal/2017/2/14/changing-old-school-financial-report-creation-processes.html

[6] Pesseract, http://pesseract.azurewebsites.net

[7] YouTube.com, *Understanding Pesseract*, https://www.youtube.com/playlist?list=PLFjXOvKVNbOrQvn_HBeyAN-hCVRZZMFJh

[8] Pesseract, http://pesseract.azurewebsites.net

[9] *Understanding Expert Systems Applicability to Financial Reporting*, http://xbrl.squarespace.com/journal/2015/7/15/understanding-expert-systems-applicability-to-financial-repo.html

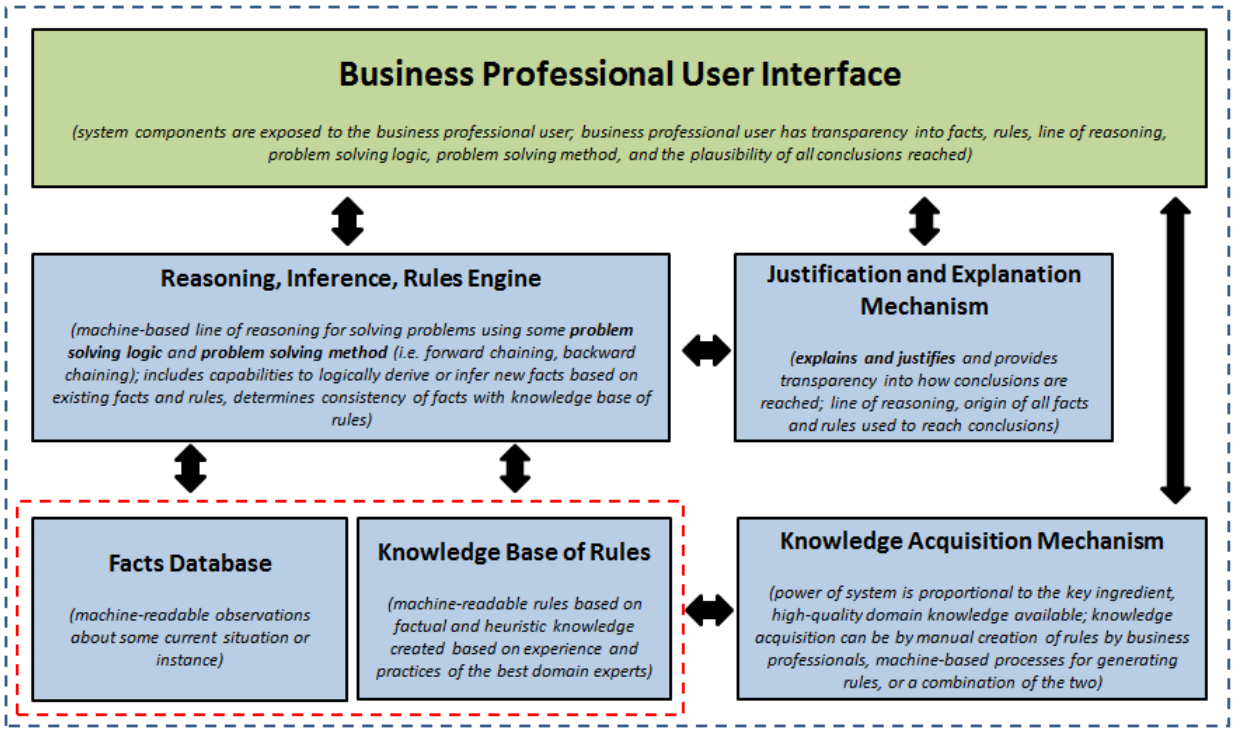
[10] Frank Puppe, *Systematic Introduction to Expert Systems*, http://www.amazon.com/Systematic-Introduction-Expert-Systems-Representations/dp/3642779735/

formally represented in machine-readable form and the knowledge based system uses that information to reach conclusions or take actions based on that information when trying to solve some problem. *A knowledge based system augments the capabilities of a human by enabling software applications to help a human similar to how a calculator helps a human do math*.

It is above and beyond the scope of this document to explain the detailed inner workings of a knowledge based system and exactly how such systems work[11] or how to build such a system. But here we will provide this succinct description of a knowledge based system in order to help you understand the general idea of what a knowledge based system is and what it does:

> Simply put, a **knowledge based system** is a system that draws upon the knowledge of human experts related to the business logic and related business rules used to solve some business problem that has been represented in machine-readable form and stored in a **fact database** and **knowledge base**. The system applies **problem solving logic** using a **problem solving method** and a **line of reasoning** to solve problems that normally would require human effort and thought to solve. The knowledge based system supplies an **explanation and justification mechanism** to support **conclusions reached** by the knowledge base system and presents that information to the business professional using the system.

This is a diagram of the components of a knowledge based system:



[11] Charles Hoffman, *Understanding Knowledge Based Systems*, http://xbrl.squarespace.com/journal/2017/5/3/understanding-knowledge-based-systems.html

The knowledge based system we are building is a construction-type system. It is for constructing a financial report. The "primitives" that we are using are **disclosures**. We leverage other primitives such as topics to organize disclosures, **templates** which are used to help create disclosures, **exemplars** which are other instantiations of the same disclosure, and **prototypes** which are best practice examples of a specific disclosure. All this is explained in our conceptual model[12].

An observation that you might have is that the objects which we defined are not mentioned in the US GAAP XBRL Taxonomy, the XBRL technical specification, or the SEC's Edgar Filer Manual. Why and how do we use these and other objects? The answer is leverage.

Understanding Patterns and Cluster

A pattern can be defined as idea that has been useful in one practical context and will likely also be useful in other contexts. Think of patterns as a way of putting building blocks into context; for example, to describe a re-usable solution to a problem. Building blocks are what you use; whereas patterns can tell you how you use them, when, why, and what trade-offs you have to make in doing so.

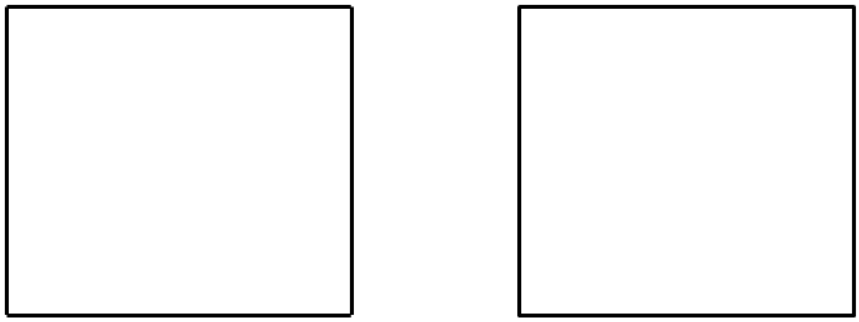
Also, consider the notion of clusters. Malcolm Gladwell explains clusters nicely in his Ted Talk, *Choice, happiness, and spaghetti sauce*[13]. He helps one understand variability and how grouping things into clusters can be used to better understand which the best pickle is and what the best spaghetti sauce is. The answer is that there is no best pickle or spaghetti sauce. But there are best clusters of pickles and spaghetti sauces.

Using patterns, clusters, and other ideas you begin to understand that you are not building software for an individual user's perceived preferences. If software is too inflexible, it will not be usable to perform the tasks that a use needs to perform. If software is too flexible, the software can very likely meet the needs of everyone but no one will use it because the software is too hard to use. But striking an appropriate equilibrium, leveraging the notions of patterns, clusters, and some other ideas; then software can be constructed that a user will be very happy with, will find the software easy to use, and the software will meet the needs of a large portion of the market, and if the minority of others do not have their needs met then they probably need to find a software application that has a better fit.

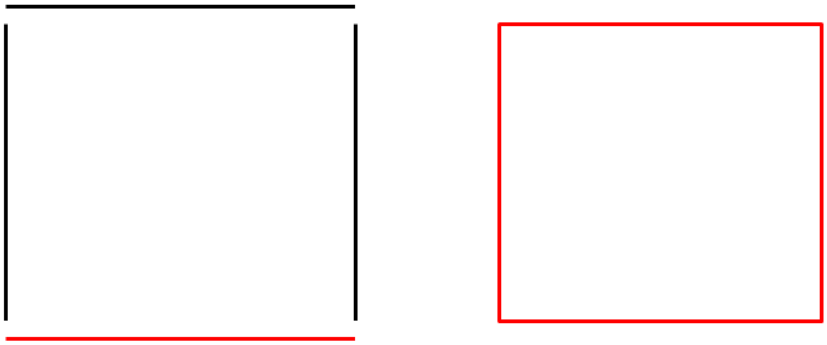
Leveraging Patterns, Compound Objects, Composite Objects

Patterns, compound objects, and composite objects provide leverage and make software easier to use. Look at these two objects below. These objects look like the same things, but they are not the same. On the LEFT are four lines; on the RIGHT is a square.

[12] *Conceptual Model of a Digital Financial Report*, http://xbrl.squarespace.com/conceptual-model/
[13] TED, Malcom Gladwell, *Choice, happiness, and spaghetti sauce*, https://www.ted.com/talks/malcolm_gladwell_on_spaghetti_sauce



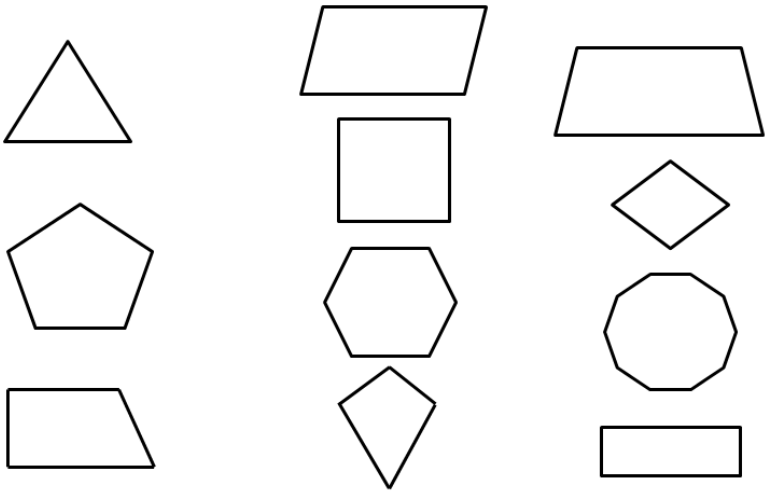
Below is another view of those same two objects. Note that one is really four lines and the other is a square:



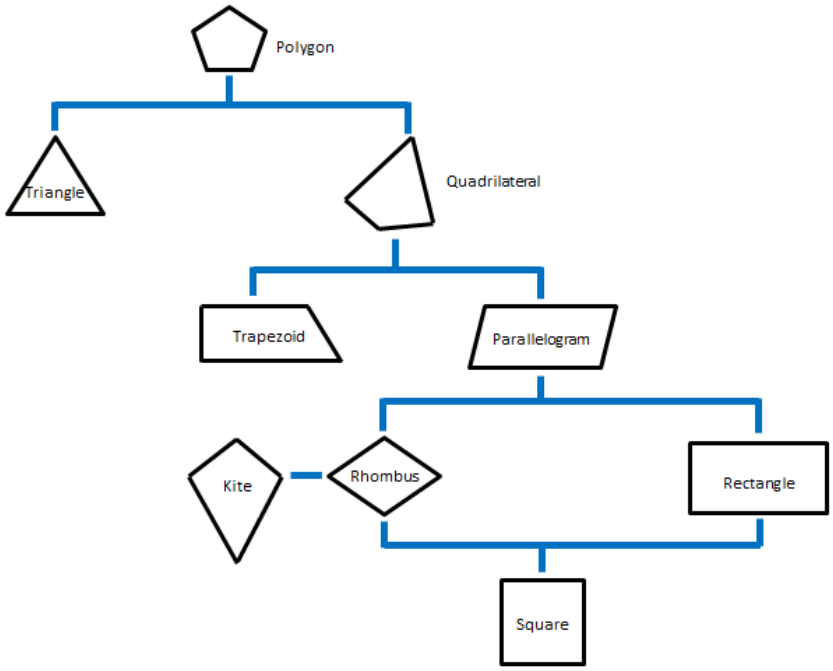
Imagine having to work with low-level objects in an application such as PowerPoint. To make your life easier, PowerPoint provides high-level objects that you can use to perform work rather than requiring you to create your presentations using only low-level objects. These higher-level objects are "compound objects" or "composite objects". These higher level objects are easier to use.

Categories

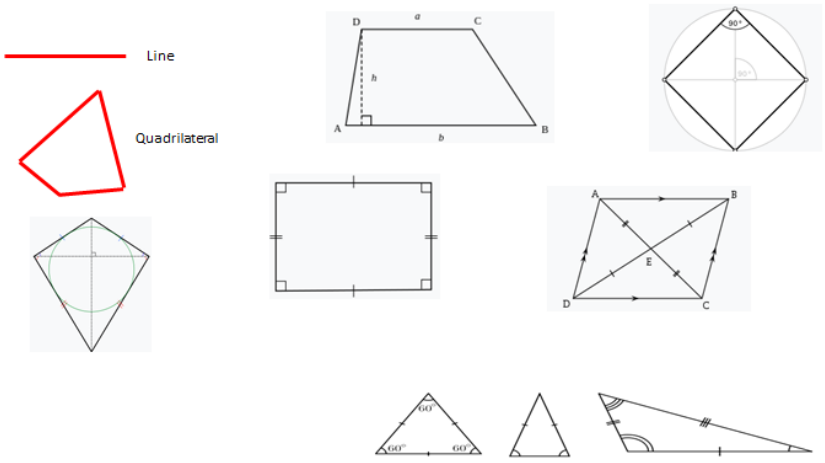
Here are a bunch of different compound or composite objects, all of which are made out of lines:



These objects can be organized and **categorized** in useful ways leveraging patterns that can be identified within the objects. For example, one pattern is the *number of lines* the object contains. Another pattern is the angle of the lines relative to one another. Another is the relative length of opposite sides of an object. Here is a simple diagram where various shapes that are made out of lines are categorized relative to each other:

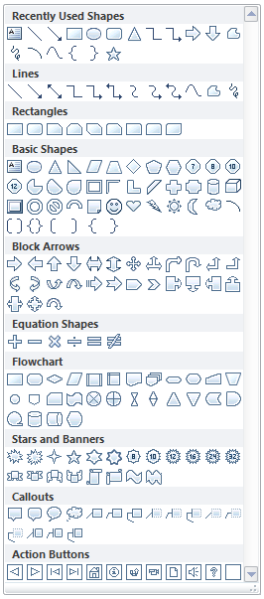


Objects can be described and identified by their **properties**. Note that lower-level objects like a line have fewer properties. Note that a quadrilateral has no properties other than that it is made up of four lines. But other objects are rich with properties. Object properties are **universal business rules** that can be embedded in software applications, leveraged by software engineers, business users never have to deal with violating the rules because software can be created that will not let them violate the rules. Business users never have to manage these universal business rules because the rules never change.



Other rules can change and therefore business professionals need to be provided with a mechanism for adjusting rules.

Once objects are identified, named, their properties identified, and objects classified they can be organized within software applications. Below is an example from Microsoft PowerPoint. The interface helps you work with categories of higher-level objects; rather than lower-level pieces. You can always group lower level objects to create new higher-level objects. To do this, the objects do need to be defined, the properties articulated, the categories created. And doing so has advantages as can be exemplified by PowerPoint. The point is: you use higher-level objects that have been provided to you, not by creating your own higher-level objects using lines.

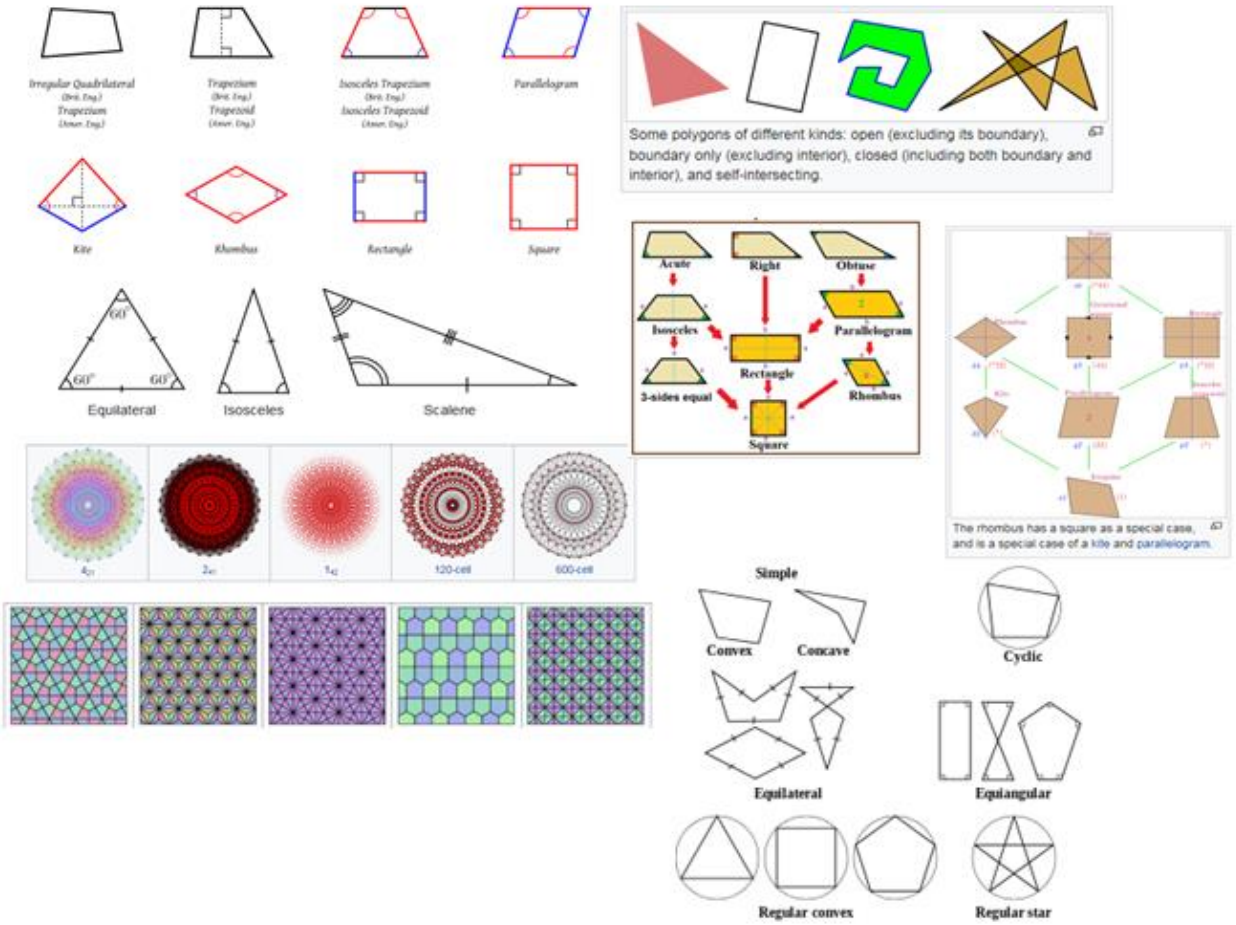


Flexibility

Flexibility should be a well-thought-out and conscious choice. There are literally an infinite number of possible polygons which can each have different properties. But all squares, which are a specialization of a polygon, have very similar properties. By determining the length of *one line* of a square, you

determine everything there is to know about a square. Why? Because of the rules of a square: *all sides are equal and every angle between lines is always equal to 90 degrees*.

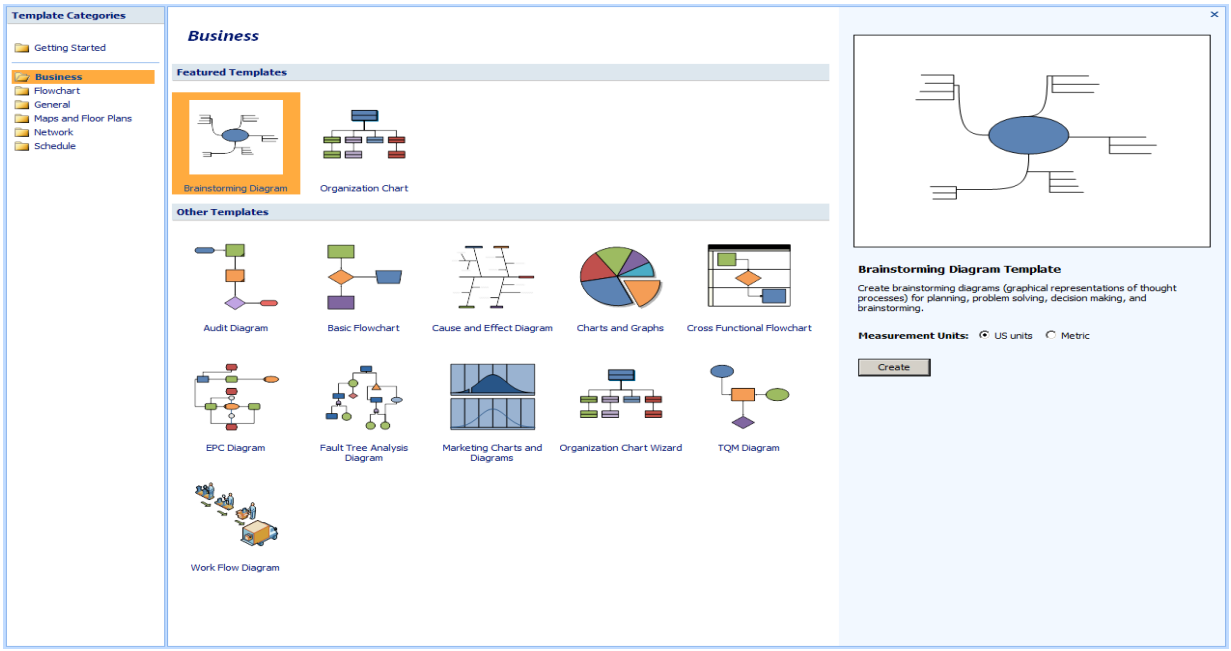
If you need more flexibility than what a square offers, create a new category of object and give it a name and defines that new object's properties as you may deem appropriate. Make things flexible where they need to be flexible. Don't just create general flexibility to play it safe…that makes working with objects unnecessarily harder.



Objects can be organized into useful categories or other grouping to present sets of objects to a business user that needs to make use of the object.

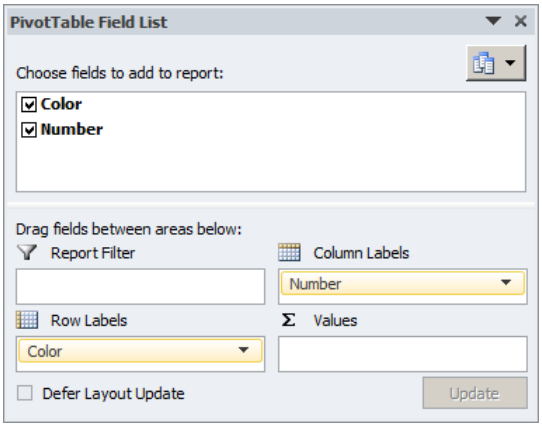
Templates

Below is one possible interface that is used to organize a set of templates. Templates organize categories of objects and help you use those objects within having to create them from scratch. There are many, many different possibilities for organizing and working with objects, the limit is only the ability of a business professional to express what they want or need and the cleverness of a software developer to create something new or find something that has already been created and use those ideas to solve the new problem you are trying to solve:



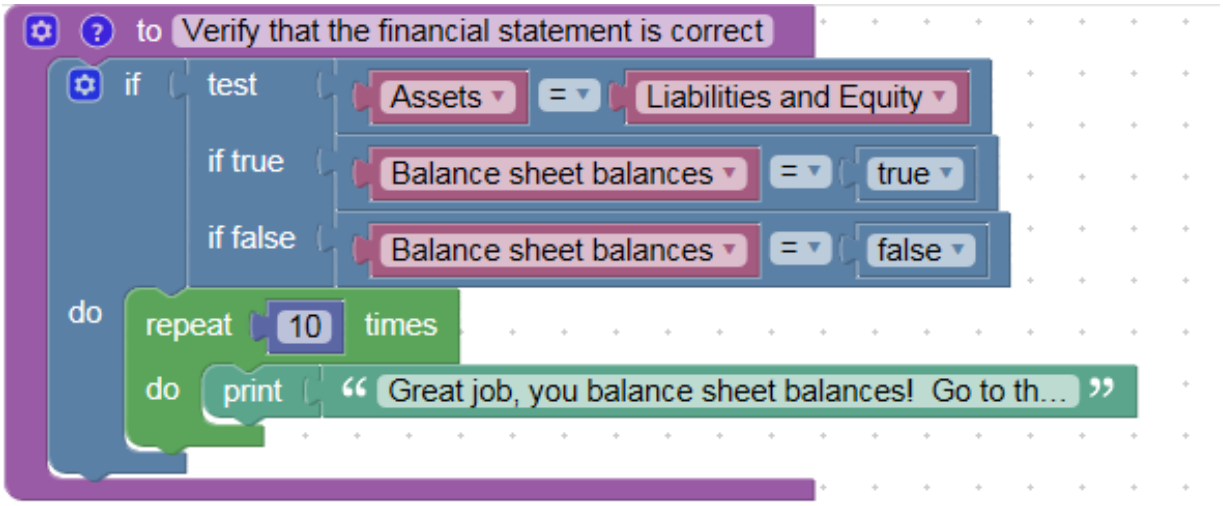
Snapping pieces together like Legos

Business professionals work with objects at the highest possible level, snapping pieces together like Legos and watched over the objects by the software application leveraging the patterns and business rules which helps make sure business professionals don't make mistakes where that task is possible for software to perform. So think of a pivot table. Most business professionals have used Excel pivot tables:

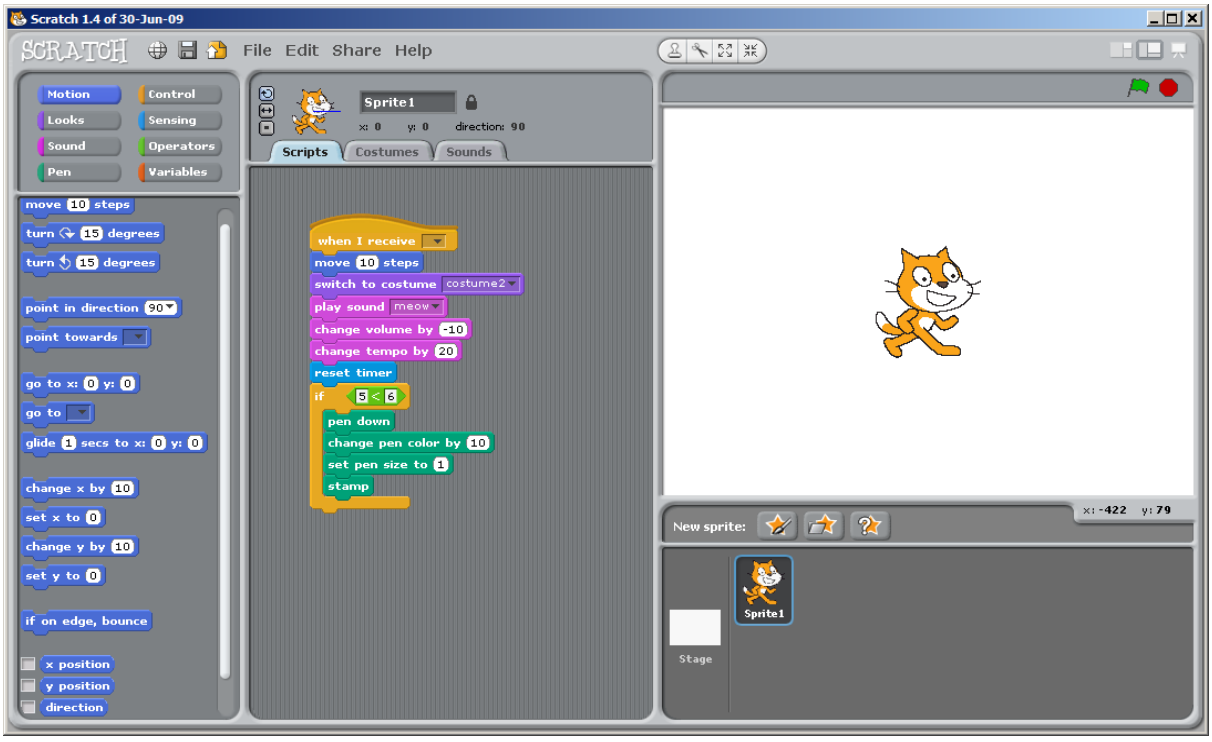


Keep the notion of a pivot table in the back of your mind. We will explain why an Excel pivot table is not useful for what we need to do unfortunately. But the general model of a pivot table is good. It is just that the pivot table functionality needs to be slightly different. Imagine this notion of "Legos" or "pieces

that snap together". See this example of Blockly[14] below. Blockly is based on ideas from Scratch which was created by MIT[15].



Imagine high-level objects that you "snap" or "glue" together using semantics or the business rules of the information itself. This is, as opposed to the books, sheets, columns, rows, and cells being glued together using presentation-oriented artifacts. Image that you added a work flow to these high-level objects that snap together, consider this interface of Scratch:



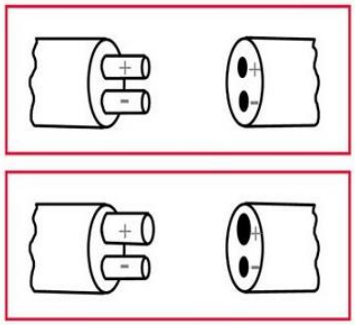
[14] Blockly, http://xbrl.squarespace.com/journal/2014/7/14/blockly.html
[15] Scratch, http://scratch.mit.edu/

Forget about the fact that Scratch is for creating animations and not financial reports. Think about how Scratch and Blockly and Excel pivot tables work rather than specifically about what they do.

Poka-yoke: Mistake proofing software

Poka-yoke[16] is a technique used to prevent mistakes through smarter design. Poka-yoke is a Japanese term that means "mistake-proofing". A poka-yoke is any mechanism consciously added to a process that helps an equipment operator avoid mistakes. Its purpose is to eliminate defects by preventing, correcting, or drawing attention to human errors as the errors occur.

For example, consider the graphic[17] below. You want someone to plug the plug into the receptacle such that positive and negative match up; inadvertently reversing this would have catastrophic consequences. In the top graphic notice that it is possible to make a mistake but in the bottom a mistake would be impossible because of the size differences in the positive and negative receptacle and plug.



Smart design means less user errors. Poka-yoke techniques can be used to create software that is easier to use, can eliminate certain types of user mistakes, and can help guide users to put the Lego pieces together to get what they want.

Specific tools as contrast to general tools

Another point that Frank Puppe points out in his book is the difference between a general tool and a domain specific tool[18]. Specific tools are always easier to use than general tools. Why? Because general tools have to be more flexible and so users of the applications have to learn more in order to manage the flexibility. With specific tools what you can do is more constrained, more specific. By definition, specific tools are easier use than general tools.

Imagine that you created a fairly general tool that, say, could be used business reporting and you could specialize that tool using different metadata for each special use case. Imagine you leveraged the

[16] Wikipedia, *Poka-yoke*, https://en.wikipedia.org/wiki/Poka-yoke
[17] Process Exam, *Six Sigma Tools - Poka Yoke*, http://www.processexam.com/six-sigma-tools-poka-yoke
[18] Frank Puppe, *Systematic Introduction to Expert Systems*, page 11, https://books.google.com/books?id=_kKqCAAAQBAJ (note that you can see the first several chapters)

multidimensional model[19] for business reporting. The functionality was general enough to be useful for various different business reporting needs, but also specific enough so that the functionality is easy for business professionals to use. And say you build the tool so you can, say, remove the US GAAP reporting scheme metadata and replace it with the IFRS metadata. Keep that thought in the back of your mind.

Multidimensional model

You leveraged the multidimensional model to make the business report flexible enough to serve many different use cases, but specific enough that the model would not be so complicated that the model was not approachable by business professionals. Think OLAP but without the limitations of OLAP[20]. Think the Microsoft Excel Pivot Table, but where the limitations of OLAP which are included in the Excel pivot table are removed. What are the limitations of OLAP and the current Microsoft Excel Pivot Table? Here is a summary of those limitations:

- OLAP forces you to use aggregation. Meaning, totals are never stored in an OLAP system. Totals are always computed. But in reporting many times you need and want to totals which serve as cross checks to make sure numbers that are supposed to agree do agree.
- There is no global standard for OLAP, whereas XBRL is a global standard and XBRL has a global standard dimensional model.
- OLAP cubes are rigid primarily because the cubes are stored in relational databases; creating new cubes is hard for business professionals.
- OLAP has limited computation support, mainly roll ups; XBRL supports roll ups, roll forwards, adjustments, variances, and many other sorts of mathematical relations.
- Because OLAP implementations are not standard; there is limited business rule support and inability to exchange business rules between implementations.
- Inability to transfer cubes between systems, each system is a "silo" which cannot communicate with other silos
- Inability to articulate metadata which can be shared between OLAP systems
- OLAP has a primary focus on numeric-type information and inconsistent support or non-existent for text data types
- OLAP systems tend to be internally focused within an organization and do not work well externally, for example across a supply chain
- OLAP tends to be read only; XBRL is read-write.

[19] *Introduction to the Multidimensional Model for Professional Accountants*, http://xbrl.squarespace.com/journal/2016/3/18/introduction-to-the-multidimensional-model-for-professional.html

[20] *Understanding Cell Stores and NOLAP, the Future of the Spreadsheet*, http://xbrl.squarespace.com/journal/2014/11/14/understanding-cell-stores-and-nolap-the-future-of-the-spread.html

So basically, image a global standard multidimensional model, such as the model provided by XBRL Dimensions[21] which allowed you to use OLAP when you needed it, but it did not compel you to use OLAP all the time.

High-Level Objects of a Business Report and Financial Report

A financial report, which is a specialization of a business report, has specific objects. The multidimensional model has objects. As we said, these objects can be organized into a **conceptual model** that describes a financial report[22]. I have given all of these objects names.

Conceptual model

Here is a summarized overview of those objects for reference (use the actual conceptual model for a more precise definition):

- **Report**: Report which communicates financial and nonfinancial information.
- **Component**: A component is a set of facts which go together for some specific purpose within a financial report.
- **Fact**: A fact defines a single, observable, reportable piece of information contained within a financial report, or fact value, contextualized for unambiguous interpretation or analysis by one or more distinguishing characteristics.
- **Characteristic**: A characteristic or distinguishing aspect provides information necessary to describe a fact or distinguish one fact from another fact.
- **Parenthetical explanation**: Facts may have parenthetical explanations which provide additional descriptive information about the fact.
- **Relation**: A relation[23] is some interaction between the pieces which make up a financial report.
- **Property**: A property is a trait, quality, feature, attribute, or peculiarity which is used to define its possessor and is therefore dependent on the possessor.
- **Block**: A block[24] is a part of a component that participates in the same concept arrangement pattern.
- **Slot**: A slot is simply the idea of an allotted place where something can be logically and sensibly placed in a fragment of a financial report, or Block.
- **Disclosure**: A Disclosure is simply a set of facts that is disclosed.
- **Topic**: A Topic is simply a set of Disclosures that are grouped together for some specific reason.
- **Exemplar**: An Exemplar is an example of a Disclosure from some other existing financial report.
- **Template**: A Template is a starting point or sample used to create a complete Disclosure.

---

[21] XBRL International, *XBRL Dimensions 1.0*, http://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html

[22] *Introducing the Conceptual Model of a Digital Financial Report*, http://xbrlsite.azurewebsites.net/2017/IntelligentDigitalFinancialReporting/Part02_Chapter05.1_IntroductionToTheConceptualModelOfDigitalFinancialReport.pdf

[23] *A Taxonomy of Part-Whole Relations*: http://csjarchive.cogsci.rpi.edu/1987v11/i04/p0417p0444/MAIN.PDF

[24] *Understanding Blocks, Slots, Templates and Exemplars*, http://xbrl.squarespace.com/journal/2015/5/11/understanding-blocks-slots-templates-and-exemplars.html

The next sections describe specific high-level objects in greater detail that help you understand how we are leveraging these objects.

Blocks

Think a financial report (which is a complex type of business report) as being made up of a collection of multiple fragments. The document *Disclosure Best Practices*[25] provides examples of about 65 different types of fragments that make up a financial report. This web page[26] links to the same 65 disclosures and examples from hundreds of examples from XBRL-based financial reports of those disclosures. The key thing to grasp here is the notion of patterns in the report fragments. This is one very basic example of a report fragment:

Inventory Disclosure [Abstract]	Period [Axis]	
	2016-03-31	2015-03-31
Inventory Disclosure [Abstract]		
Raw materials and supplies	7,993,000	7,417,000
Work-in-progress	13,147,000	6,466,000
Finished goods	5,600,000	2,891,000
Inventories	26,740,000	16,774,000

I call a piece, a fragment of a report, such as this a "**Block**". We just made up the notion of a Block[27] because it is useful. The pieces of a Block are held together with the semantics of the multidimensional model, strong semantics (rules) of a business report such as "roll up" or a "roll forward" etc., the rules of XBRL Dimensions, common financial reporting practices, and some other semantics. A report is held together by the relations between Blocks, what we call "intersections", by strong fundamental high-level relationships (i.e. the fundamental accounting concepts continuity cross checks) and strong financial reporting domain disclosure semantics (balance sheet, income statement, cash flow statement, long term debt maturities), and some other semantics.

The [Line Items] of a Block are organized in specific ways and we gave that a name, we called that the **Concept Arrangement Patterns**. Here are some examples of different types of Blocks differentiated by their concept arrangement pattern:

Roll Up

[25] *Disclosure Best Practices*, http://www.xbrlsite.com/2017/Prototypes/DisclosureAnalysis/DisclosureBestPractices.pdf
[26] Best Practice Examples of Disclosures for Campaign to Improve Disclosure Quality, http://www.xbrlsite.com/site1/2017/Prototypes/DisclosureAnalysis/All/
[27] YouTube.com, *Understanding Blocks*, https://www.youtube.com/watch?v=yI9yjD_T78I

Property, Plant and Equipment [Line Items]	Period [Axis]	
	2016-06-30	2015-06-30
Land	824,000,000	769,000,000
Buildings and improvements	12,393,000,000	10,800,000,000
Leasehold improvements	3,659,000,000	3,577,000,000
Computer equipment and software	17,391,000,000	13,612,000,000
Furniture and equipment	3,889,000,000	3,579,000,000
Total, at cost	38,156,000,000	32,337,000,000
Accumulated depreciation	(19,800,000,000)	(17,606,000,000)
Total, net	18,356,000,000	14,731,000,000

Roll Up + Member Aggregation

Property Plant And Equipment [Line Items]	Period [Axis]									
	2016-12-31					2015-12-31				
	Property, Plant and Equipment, Type					Property, Plant and Equipment, Type				
	Land	Buildings and Improvements	Equipment and Software	Construction in progress	Property, Plant and Equipment, Type	Land	Buildings and Improvements	Equipment and Software	Construction in progress	Property, Plant and Equipment, Type
Total property and equipment	14,705,000	103,742,000	25,206,000	27,841,000	171,494,000	13,380,000	77,412,000	16,149,000	17,424,000	124,365,000
Less accumulated depreciation					(30,187,000)					(14,641,000)
Property and equipment, net					141,307,000					109,724,000

Roll Forward:

Income Tax Disclosure [Abstract]	Period [Axis]		
	2016-01-01 - 2016-12-31	2015-01-01 - 2015-12-31	2014-01-01 - 2014-12-31
Income Tax Disclosure [Abstract]			
Summary of activity related to uncertain tax positions:			
Balance at January 1,	3,561,000	2,644,000	1,960,000
Additions Based on Tax Positions Related to the Current Year	258,000	331,000	311,000
Additions for Tax Positions of Prior Years	293,000	1,176,000	928,000
Prior Year Reductions	(776,000)	(1,000)	(370,000)
Statute Expirations	(609,000)	(589,000)	(94,000)
Settlements	(133,000)	0	(91,000)
Balance at December 31,	2,594,000	3,561,000	2,644,000

Roll Forward + Member Aggregation:

Component: (Network and Table)

Network: 100800 - Disclosure - **Carrying Amount of Goodwill (Detail)**

Table: Schedule of Goodwill [Table]

Reporting Entity [Axis] | 0000789019 http://www.sec.gov/CIK | Legal Entity [Axis] | Entity [Domain]

Period [Axis] | Statement Business Segmen...

Goodwill [Line Items]	2016-07-01/2017-06-30				2015-07-01/2016-06-30			
	Productivity and Business Processes	Intelligent Cloud	More Personal Computing	Segments [Domain]	Productivity and Business Processes	Intelligent Cloud	More Personal Computing	Segments [Domain]
Beginning Balance	6,678,000,000	5,467,000,000	5,727,000,000	17,872,000,000	6,309,000,000	4,917,000,000	5,713,000,000	16,939,000,000
Acquisitions	17,072,000,000 [1]	49,000,000	115,000,000	17,236,000,000	443,000,000	549,000,000	100,000,000	1,092,000,000
Other	(11,000,000)	39,000,000	(14,000,000)	14,000,000	(74,000,000)	1,000,000	(86,000,000)	(159,000,000)
Ending Balance	23,739,000,000	5,555,000,000	5,828,000,000	35,122,000,000	6,678,000,000	5,467,000,000	5,727,000,000	17,872,000,000

Adjustment:

Reporting Entity [Axis]	0000000001 http://www.sec.gov/CIK	
Legal Entity [Axis]	Consolidated Entity [Domain]	
Period [Axis]	2015-12-31	
		Drop Column Fields Here
Changes in Stockholders' Equity [Line Items]	Report Date [Axis]	Fact Value
Increase (Decrease) in Stockholders' Equity [Adjustment]		
Stockholders' equity, origionally stated	Origionally Stated Report Date [Member]	30,000,000
Correction of a prior period error	Restated Report Date [Domain]	12,000,000
Effect of mandatory change in accounting policy for adoption of FAS XXX	Restated Report Date [Domain]	(2,000,000)
Stockholders' equity, restated	Restated Report Date [Domain]	40,000,000

Variance:

Reporting Entity [Axis]	SAMP http://www.SampleCompany.com		
Legal Entity [Axis]	Consolidated Entity [Member]		
Period [Axis]	2010-01-01/2010-12-31		
	Reporting Scenario [Axis]		
Variance Analysis [Line Items]	Budgeted [Member]	Variance [Member]	Actual [Member]
Variance Analysis [Hierarchy]			
Sales	5,000	1,000	6,000
Cost of Goods Sold	3,000	1,000	4,000
Contribution Margin	2,000	(1,000)	1,000
Distribution Costs	1,000	0	1,000

Hierarchy:

Disclosure of Compensation Related Costs, Share-based Payments [Abstract]	Period [Axis]		
	2015-09-01 - 2016-08-31	2014-09-01 - 2015-08-31	2013-09-01 - 2014-08-31
Disclosure of Compensation Related Costs, Share-based Payments [Abstract]			
Dividend yield	0.19%	0.15%	0.11%
Expected volatility	38.06%	37.80%	38.30%
Risk-free rate of return	2.06%	1.19%	0.70%
Expected forfeiture rate	13.12%	13.73%	14.18%
Expected life	P6Y1M	P6Y1M	P6Y1M

Hierarchy + Members WITHOUT aggregation:

Defined Benefit Plan Disclosure [Line Items]	Period [Axis]		
	2016-08-31		
	Defined Benefit Plans and Other Postretirement Benefit Plans [Axis]		
	U.S. Plans	Non-U.S. Plans	Other Postretirement Benefit Plan [Member]
2017	46,881,000 [1]	44,537,000	10,259,000
2018	49,865,000 [1]	50,094,000	11,469,000
2019	53,277,000 [1]	55,964,000	12,598,000
2020	56,950,000 [1]	66,225,000	13,942,000
2021	61,361,000 [1]	75,166,000	15,830,000
2022-2026	373,921,000 [1]	416,507,000	110,756,000

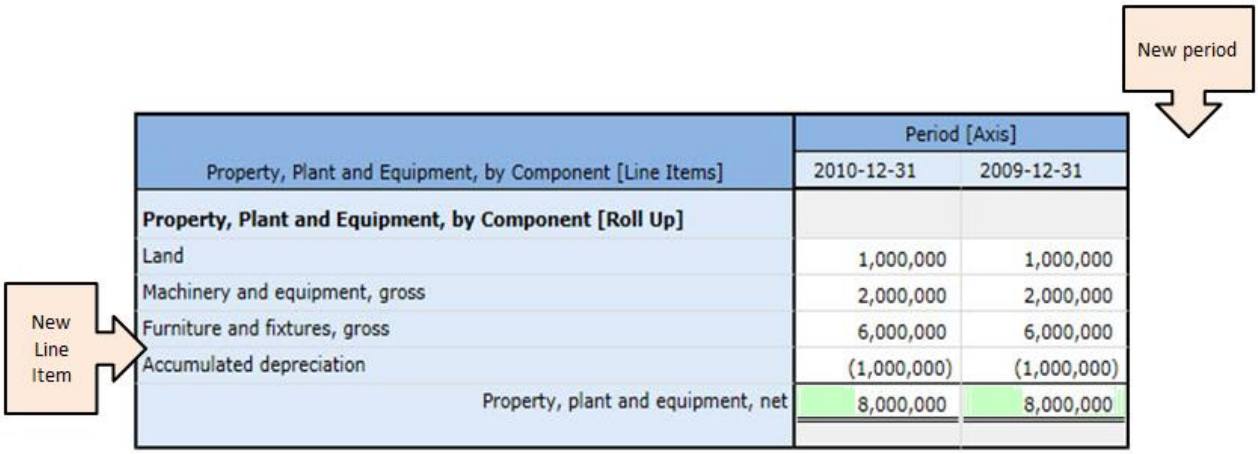
Roll Forward Info:

Reporting Entity [Axis]	SAMP http://www.SampleCompany.com
Legal Entity [Axis]	Consolidated Entity [Member]
	Period [Axis]
Weighted Average Grant Date Fair Value [Line Items]	2010-01-01/2010-12-31
Weighted Average Grant Date Fair Value [Roll Forward Info]	
Nonvested Fair Value, Beginning Balance	32.72
Granted	41.51
Vested	30.92
Forfeited	35.93
Nonvested Failr Value, Ending Balance	36.92

100% of all XBRL-based financial reports submitted to the SEC fit into the model described above, each report fragment can be distilled down to a Block. This is not a matter of opinion, this is 100% provable.

Slots

Blocks have "**slots**". A Slot is simply a place in a Block where it makes logical sense for new objects to be added to the Block. Different types of Blocks have different slots. Below you can see one Block, showing two Slots for that block. One Slot is that a new Line Item can be added within the roll up total. Or, a second Slot is that a new period can be added to the Block.

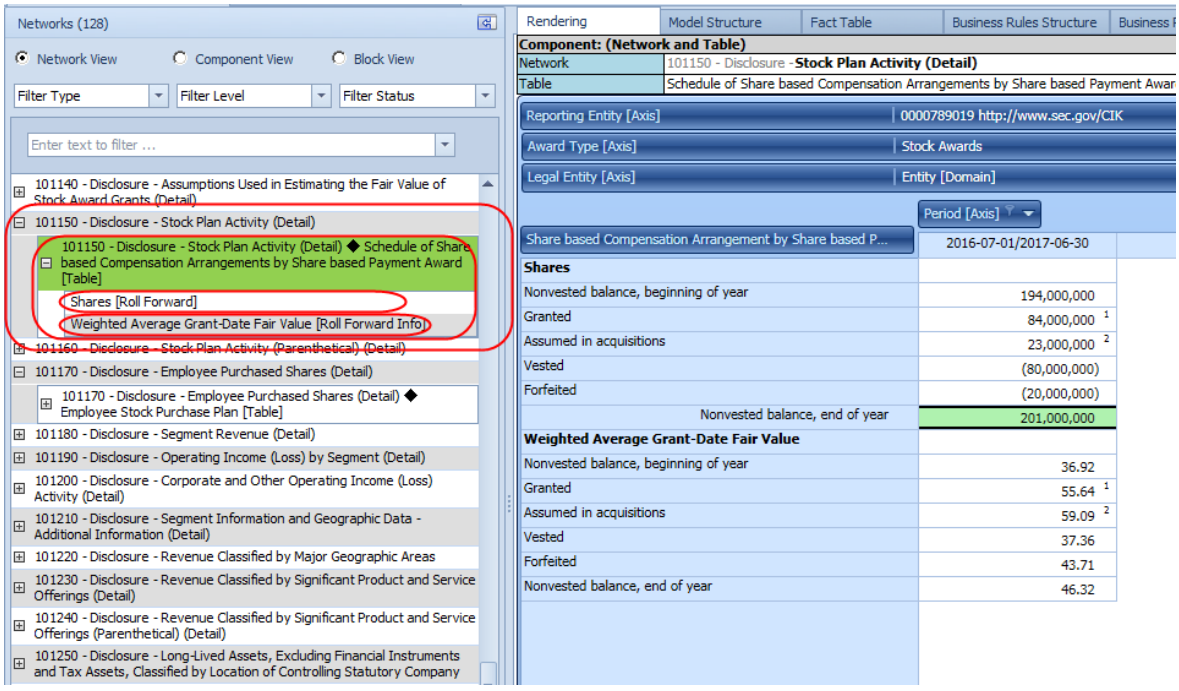


This is not a comprehensive discussion of Blocks and Slots, it only provides the general ideas of what a Block is, what a Slot is, and that a financial report can be broken down into a set of Blocks each of which has specific Slots.

Relations between Networks, Components, Blocks

A **Network** is a necessary XBRL technical syntax artifact but the semantics of a Network can be different in different implementations. A **Component** is necessary XBRL technical syntax artifact. A **Block** is an object created for convenience.

Below you see a Network that contains one Component that contains two Blocks. One Block is a roll forward and the other is a roll forward info.



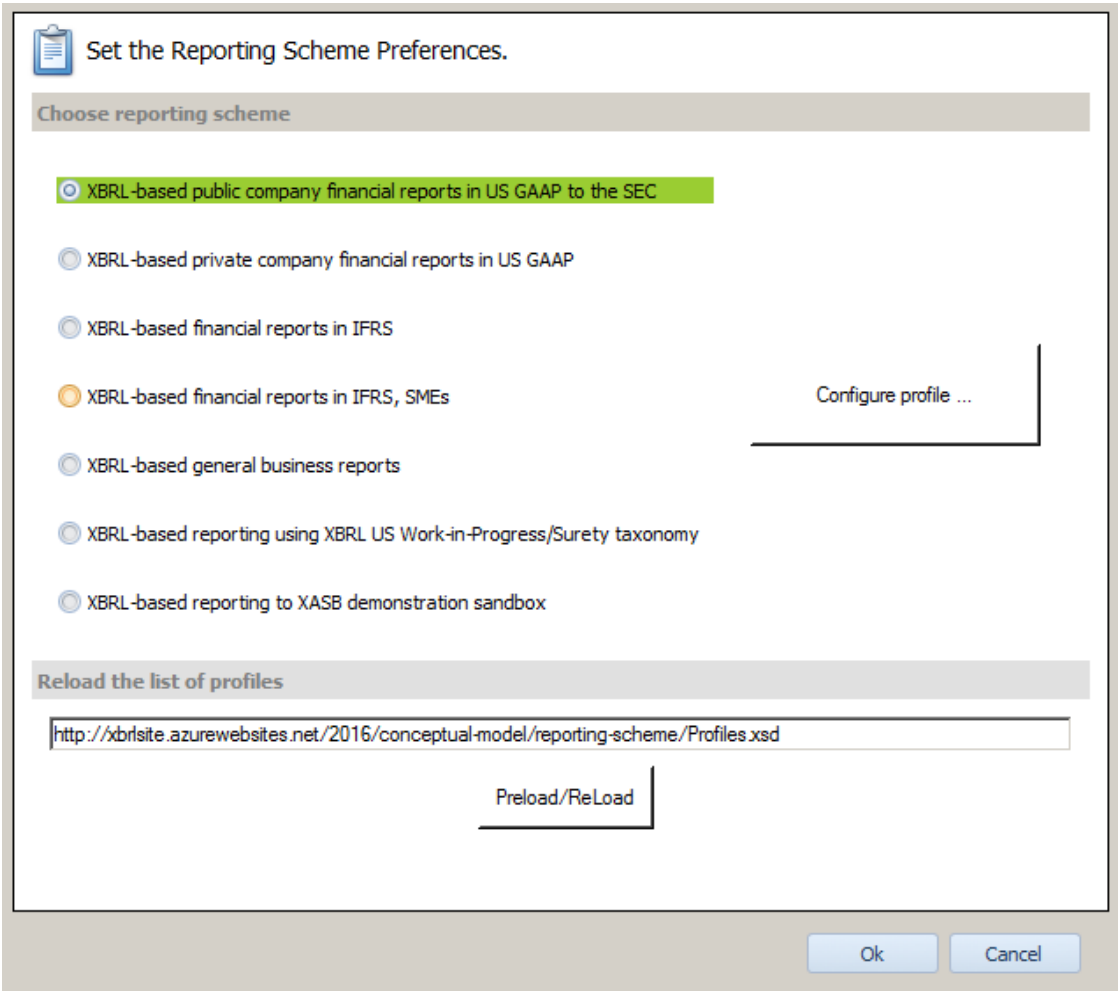
A **Block** always has meaning. A Block is some fragment of a report that is disclosed. A Disclosure is made up of one or many Blocks. All Blocks exist within Components. A **Component** is always the

combination of a Network and a [Table]/Hypercube. The meaning of a Component can be ambiguous. A **Network** may contain one or many [Table]s/Hypercubes. The meaning of a Network can be ambiguous.

Profiles

Each implementation uses different pieces of the XBRL technical syntax, no implementation ever uses all pieces of XBRL technical syntax. Each implementation can use technical syntax pieces differently where the semantics of a piece can be vague or ambiguous.

Profiles[28] are used to overcome these implementation details that can be different within different implementations.



Pesseract supports an explicit set of profiles. New profiles can be added. Any reporting scheme could use the General Business Report profile. The XASB demonstration sandbox is an implementation of the General Business Report Profile.

[28] YouTube.com, *Understanding Reporting Profiles*, https://www.youtube.com/watch?v=_dSLCoJDNSk

Pesseract is a knowledge based system

Pesseract is (or rather, will ultimately be) a purpose-build tool for business reporting and financial reporting which leverages the XBRL technical syntax. You cannot create a financial report using Pesseract yet. But you can view financial reports and verify that reports to make sure they don't have defects[29]. Pesseract does qualify as a true knowledge based system.

The **fact database** that Pesseract uses is one XBRL instance, a set of XBRL instances of one economic entity you wish to compare across many periods or otherwise want to work with (cross period comparison), a set of XBRL instances of a group of economic entities that you want to compare or otherwise work with (cross entity comparison), or the entire set of XBRL instances from the SEC EDGAR system which has been loaded into a database type application to make querying that information more efficient.

The **knowledge base of rules** that Pesseract uses is the base XBRL taxonomy created for US GAAP, IFRS, or some other reporting scheme; the entity specific XBRL taxonomy created and provided with an XBRL instance; sets of other universally applicable business rules applicable to every economic entity (for example "assets = liabilities and equity") or sets of economic entities (say specific rules for an unclassified balance sheet as contrast to a classified balance sheet). All rules are stored in the form of XBRL definition relations and XBRL Formula.

The **knowledge acquisition mechanism** is currently the manual entry of business rules into a software application and then the generation of XBRL definition relations and/or XBRL Formula using that software application.

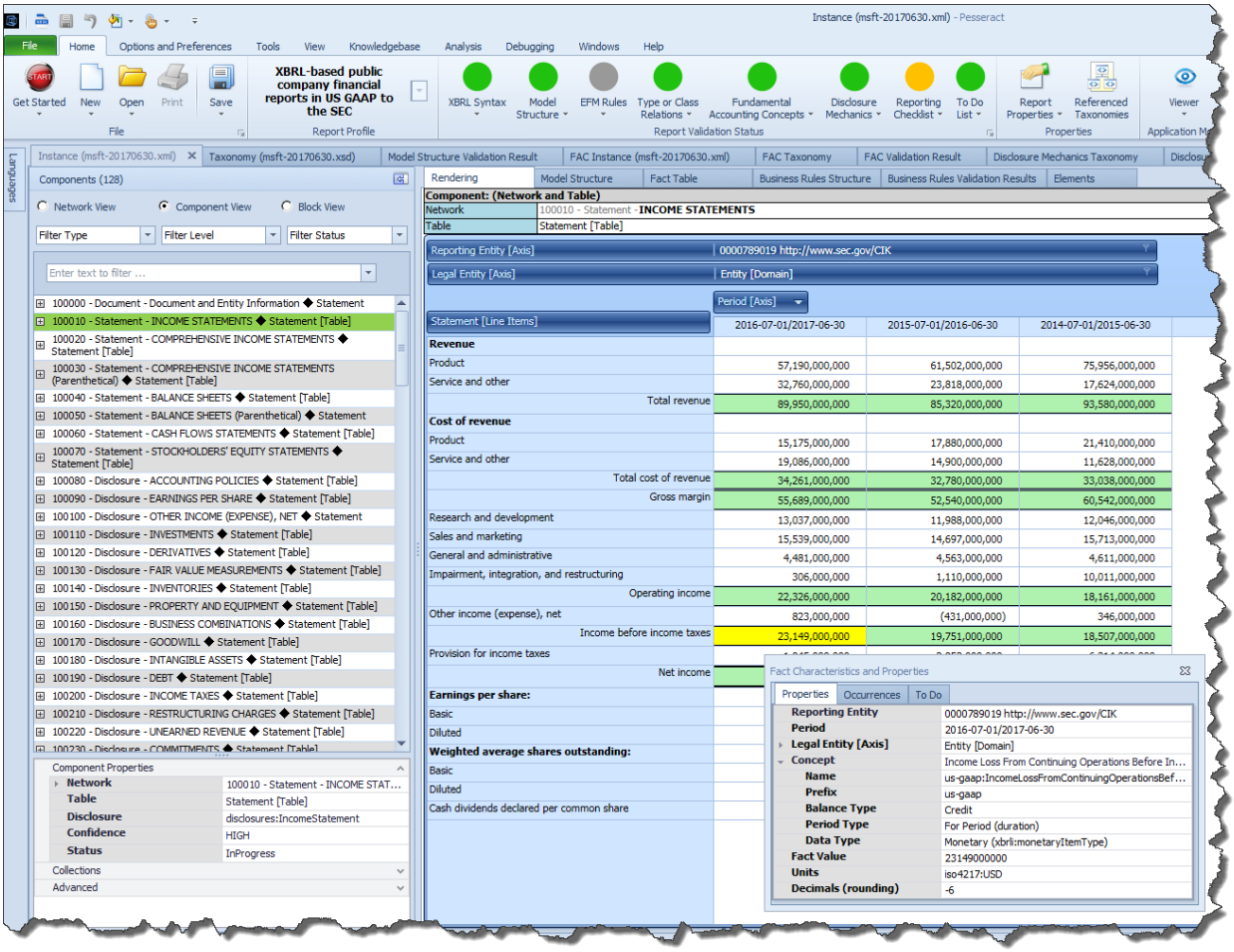
The **reasoning/inference/rules engine** is a model-driven forward chaining rules engine software application. The rules engine has both an API and GUI interfaces. The rules engine is essentially a *superset* of an XBRL Formula Processor. By superset what is meant is that the XBRL Formula Processor component of the rules engine is fully compliant to the existing XBRL International technical specification, but additional functionality has been added to overcome specific deficiencies in capabilities of the XBRL Formula specification. Specific supplemental capabilities added include:

- *Inference* which is the ability to logically derive new facts from existing facts and business rules.
- Model-driven *forward chaining*.
- Enhances access to XBRL taxonomy *model structure information*.
- *Enhanced problem solving logic* for specific high-level functionality required to enable disclosure mechanics and reporting checklist features.

The **justification mechanism** includes transparency into how conclusions are reached by the reasoning/inference/rules engine. This includes the origin of all facts and business rules used to reach conclusions and reports and other tools that enable a professional accountant to understand all aspects

[29] *Blueprint for Creating Zero-Defect XBRL-based Digital Financial Reports*, http://xbrlsite.azurewebsites.net/2017/Library/BlueprintForZeroDefectDigitalFinancialReports.pdf

of how such conclusions were reached. Information is summarized in a dashboard type style to help a professional accountant understand the state of the report.

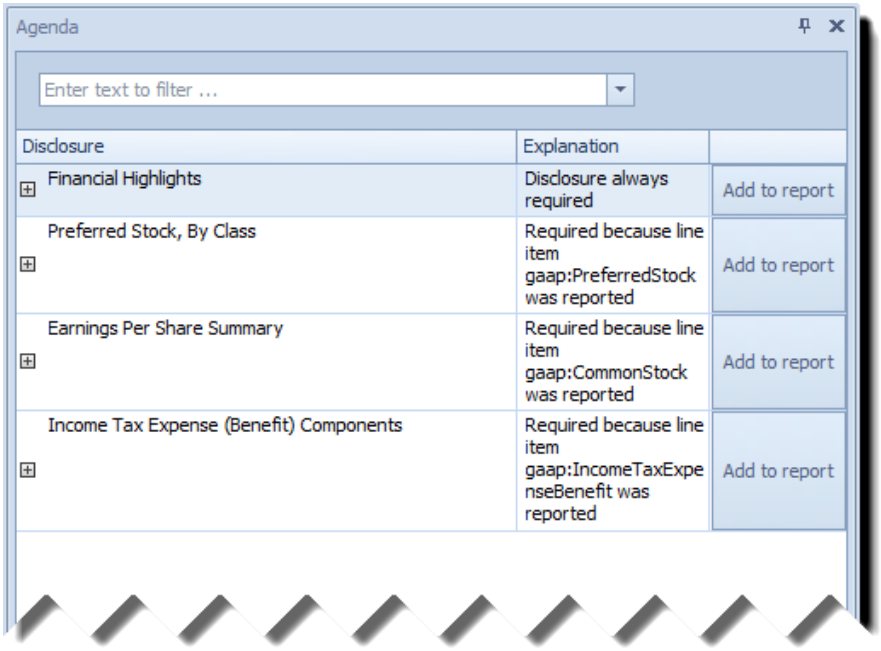


In addition to the basic components of a knowledge based system, three additional key features are provided. An **agenda** is managed by the software application and can be leveraged to organize work that needs to be performed. A **state machine** is provided to monitor key property values in order to manage processing. A **suggest disclosure template/exemplar intelligent agent** is provided to help in the process or creating disclosures.

Functionality is exposed to professional accountants with an easy-to-use graphical user interface (GUI). Functionality is also exposed to software developers with an easy-to-use application programming interface (API) so that custom features or entire processes can be created without having to reinvent basic processing of digital financial reports from scratch.

Agenda

The business professional is provided with an "Agenda[30]" which is a list of tasks which need to be performed:



The idea of an agenda came from CLIPS[31] which is a tool for building expert systems created by NASA.

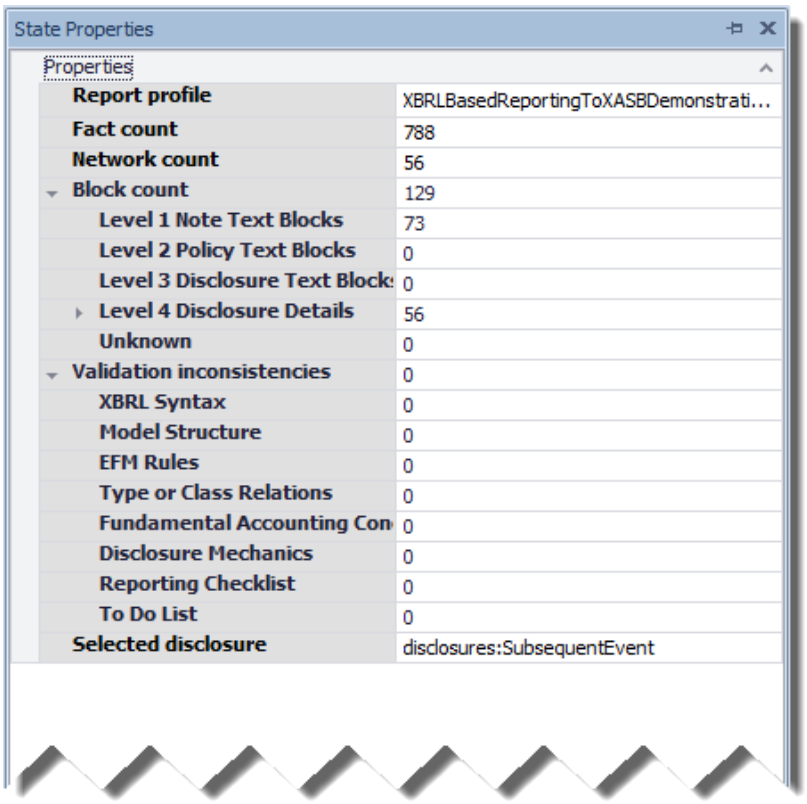
State Machine

A "State Machine[32]" monitors the properties of objects and provides this information to the application which provides information to the business professional so that they understand the tasks that need to be performed:

[30] YouTube.com, *Expert System for Creating a Financial Report*, minute 3:30 to 4:45, https://www.youtube.com/watch?v=ioapxF1gtQM

[31] Using CLIPS to Understand Expert Systems and Logic Programming, http://xbrl.squarespace.com/journal/2016/9/15/using-clips-to-understand-expert-systems-and-logic-programmi.html

[32] YouTube.com, *State Machine Basics*, https://www.youtube.com/watch?v=NuFGJHJ3jOM

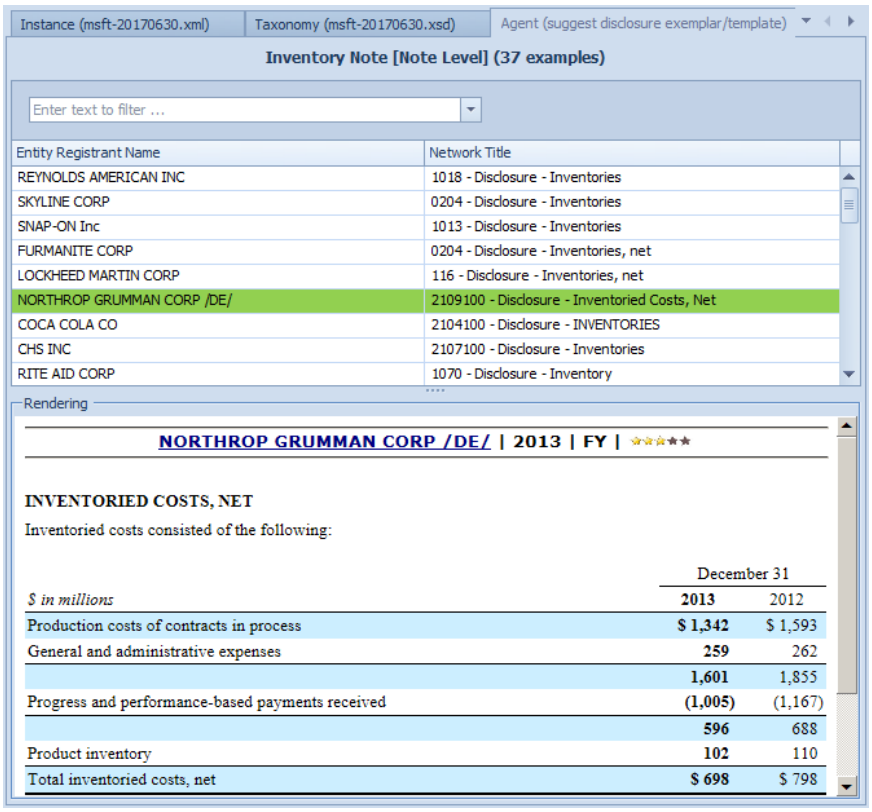


The state machine idea, or finite-state machine[33], comes from the world of video games.

Suggest Disclosure Template/Exemplar Intelligent Agent

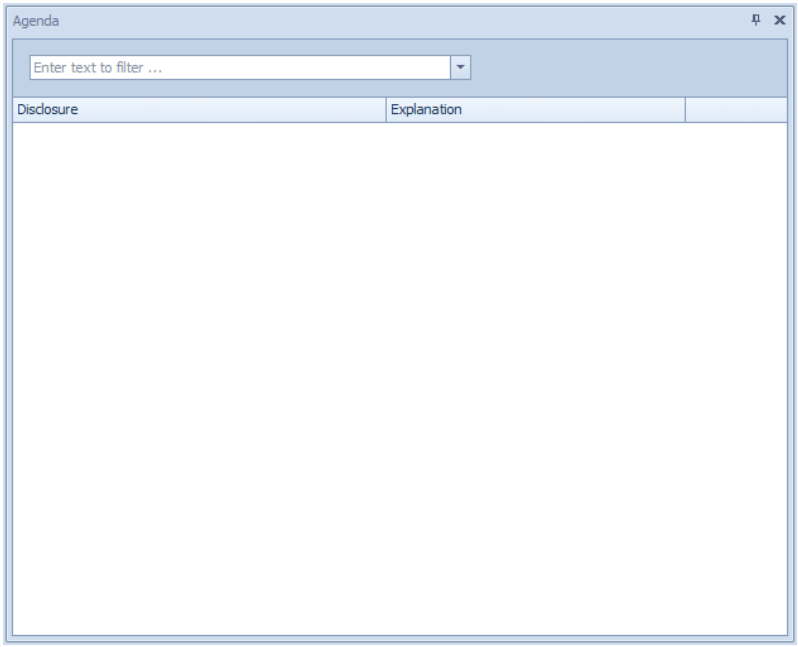
The suggest disclosure template/exemplar intelligent agent uses the current context (a state property) of the application user to suggest disclosure templates or disclosure exemplars (examples) from existing XBRL-based reports which can be "cut and pasted" (actually directly imported) into an existing report. The agent can also use the context of the agenda to add new disclosures from the reporting checklist based of reporting checklist business rules.

[33] Wikipedia, *Finite-state Machine*, https://en.wikipedia.org/wiki/Finite-state_machine



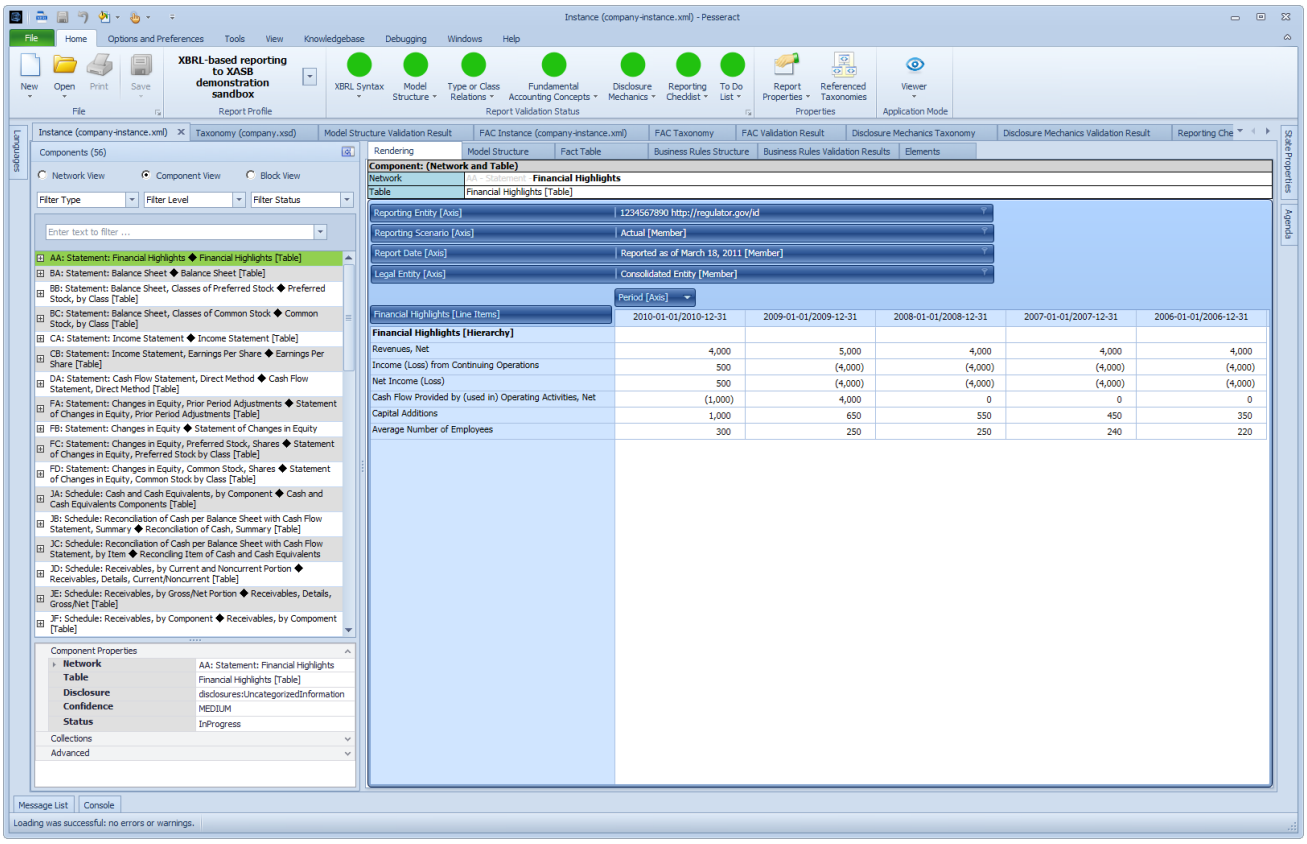
How you create a financial report in Pesseract

Let's begin at the end. This is an "agenda" or list of to do items. This agenda might not seem too interesting, because the agenda is empty. There is nothing that needs to be done, there is no work to be performed:

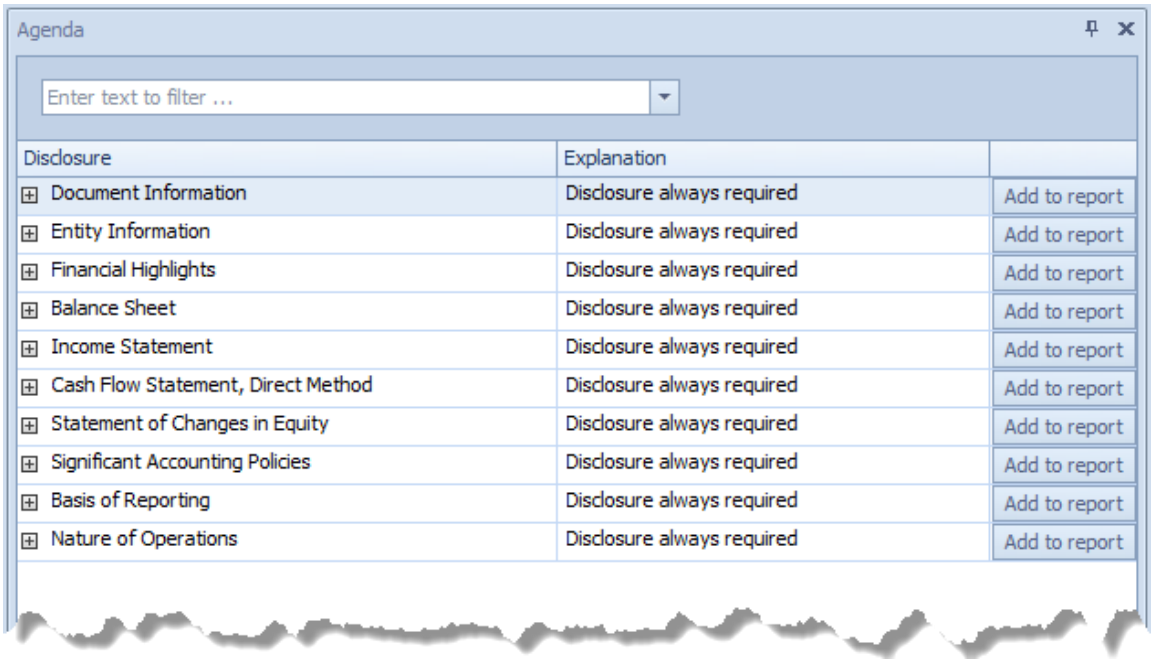


Another way of putting this is that all the **tasks** necessary to complete the creation of a financial report in compliance with rules and regulations of some reporting scheme have complete.

That is why the **agenda**, which is a list of the **tasks** which must be performed, is empty. Below you see a screen shot of the application which has an open report. All the indicator lights that are shown in the screen shot of a financial report creation application below are GREEN: because everything is done and because everything is consistent with expectations.



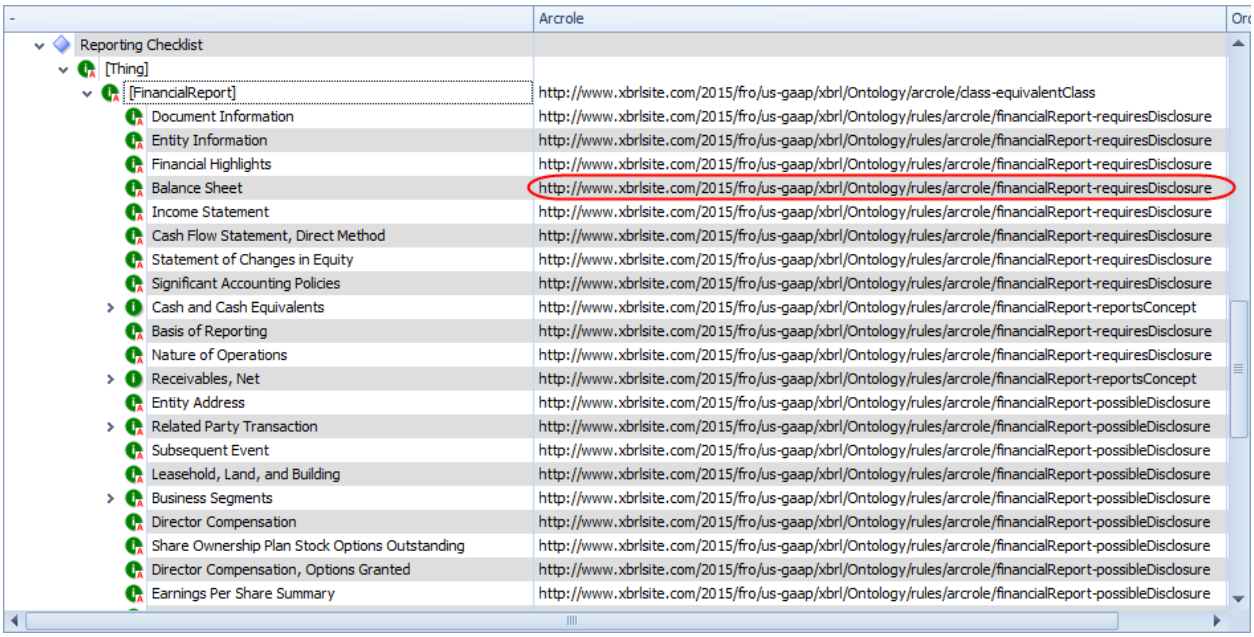
But when you begin the process of creating a financial report, the **agenda** starts off looking more like this screen shot below which lists all the disclosures which are required to be provided in a financial report that must meet some set of regulatory and statutory requirements. Your task is to create each of these disclosures and **comply** with regulatory and statutory requirements.



The agenda above shows all the **fragments**, or parts, of a financial report that must be provided per the requirements of some financial reporting scheme. When you start creating a report and the report is empty (i.e. because you have not imported or created anything yet); then you simply have a listing of the required financial report fragments or disclosures.

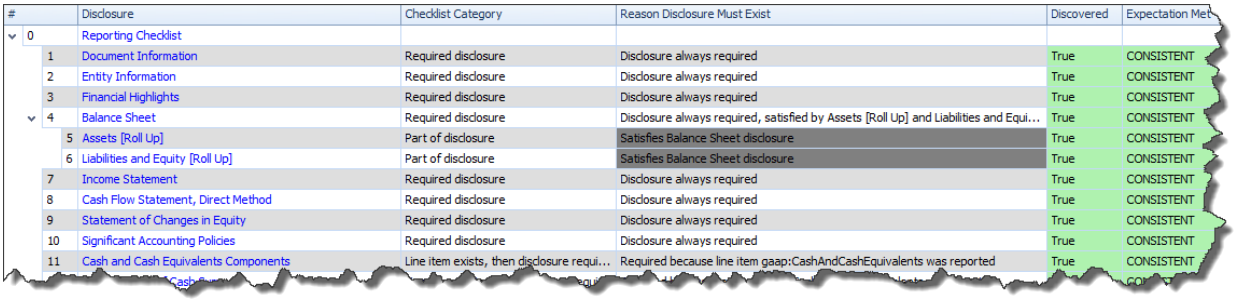
The requirements of what is to go into a financial report are dictated by the rules and regulations of the financial reporting scheme upon which the financial report is based. Humans understand those requirements by reading the requirements. But how does a software application understand those requirements? Software applications understand those requirements by articulating those requirements in some machine-readable format and then interacting with that information.

For example, here is a set of rules about what needs to be reported. Note the XBRL arcrole circled in red in the diagram. That arcrole is used to document the fact that a balance sheet is a required disclosure of a financial report.



-	Arcrole
Reporting Checklist	
[Thing]	
[FinancialReport]	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/arcrole/class-equivalentClass
Document Information	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Entity Information	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Financial Highlights	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Balance Sheet	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Income Statement	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Cash Flow Statement, Direct Method	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Statement of Changes in Equity	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Significant Accounting Policies	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Cash and Cash Equivalents	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-reportsConcept
Basis of Reporting	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Nature of Operations	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-requiresDisclosure
Receivables, Net	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-reportsConcept
Entity Address	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Related Party Transaction	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Subsequent Event	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Leasehold, Land, and Building	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Business Segments	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Director Compensation	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Share Ownership Plan Stock Options Outstanding	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Director Compensation, Options Granted	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure
Earnings Per Share Summary	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/financialReport-possibleDisclosure

Above is a screenshot of an XBRL taxonomy that contains those machine-readable rules[34] for the reporting scheme I am using. You can read those rules using any off-the-shelf XBRL taxonomy tool. Below is that same information in easier to understand terms meant to be read by humans working with a software application:



#	Disclosure	Checklist Category	Reason Disclosure Must Exist	Discovered	Expectation Met
0	Reporting Checklist				
1	Document Information	Required disclosure	Disclosure always required	True	CONSISTENT
2	Entity Information	Required disclosure	Disclosure always required	True	CONSISTENT
3	Financial Highlights	Required disclosure	Disclosure always required	True	CONSISTENT
4	Balance Sheet	Required disclosure	Disclosure always required, satisfied by Assets [Roll Up] and Liabilities and Equi…	True	CONSISTENT
5	Assets [Roll Up]	Part of disclosure	Satisfies Balance Sheet disclosure	True	CONSISTENT
6	Liabilities and Equity [Roll Up]	Part of disclosure	Satisfies Balance Sheet disclosure	True	CONSISTENT
7	Income Statement	Required disclosure	Disclosure always required	True	CONSISTENT
8	Cash Flow Statement, Direct Method	Required disclosure	Disclosure always required	True	CONSISTENT
9	Statement of Changes in Equity	Required disclosure	Disclosure always required	True	CONSISTENT
10	Significant Accounting Policies	Required disclosure	Disclosure always required	True	CONSISTENT
11	Cash and Cash Equivalents Components	Line item exists, then disclosure requi…	Required because line item gaap:CashAndCashEquivalents was reported	True	CONSISTENT

In the screen shot above, you can see the information about which disclosures are required. The software application reads that information and then populates the agenda based on whether some required disclosure exists or does not exist within the current version of the financial report.

So, you can see from the list above that an “Income Statement” is a required disclosure per the reporting scheme used. In this case an imaginary reporting scheme called XASB which is used for testing and prototyping[35] was used.

But how exactly does the software application understand if the disclosure exists or not?

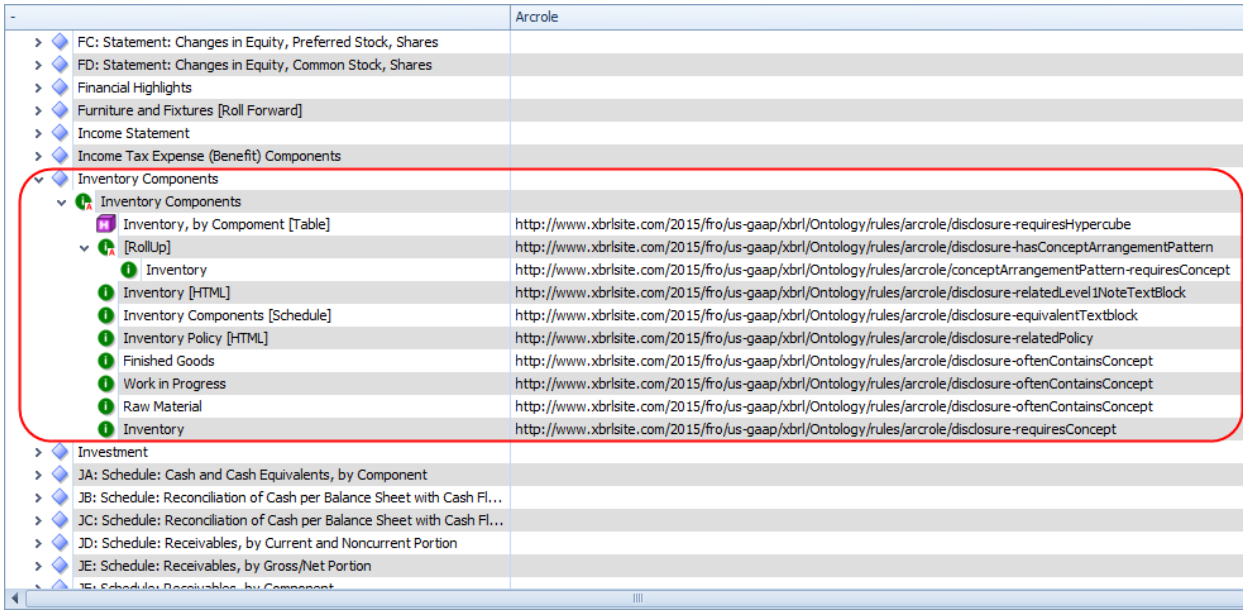
---

[34] Reporting Checklist for Imaginary XASB reporting scheme in the XBRL format, http://xbrlsite.azurewebsites.net/2016/conceptual-model/reporting-scheme/xasb/reporting-checklist/ReportingChecklist-xasb-rules-def.xml

[35] Note that this also works with the US GAAP and IFRS reporting schemes.

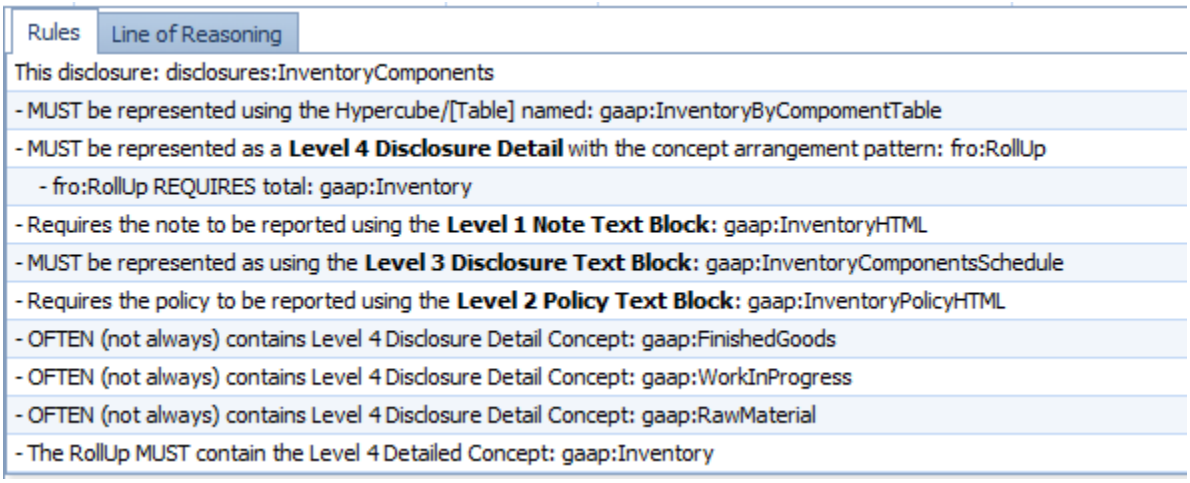
Well, because the information in the XBRL-based structured report is composed of lots of individually identifiable pieces of information and the software application can interact with those pieces. To help the software application understand the pieces it is interacting with, we give the software more machine-readable information.

In machine-readable XBRL syntax, again readable by any XBRL tool, the rules look like the following[36].



-	Arcrole
FC: Statement: Changes in Equity, Preferred Stock, Shares	
FD: Statement: Changes in Equity, Common Stock, Shares	
Financial Highlights	
Furniture and Fixtures [Roll Forward]	
Income Statement	
Income Tax Expense (Benefit) Components	
Inventory Components	
Inventory Components	
Inventory, by Compoment [Table]	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-requiresHypercube
[RollUp]	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-hasConceptArrangementPattern
Inventory	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/conceptArrangementPattern-requiresConcept
Inventory [HTML]	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-relatedLevel1NoteTextBlock
Inventory Components [Schedule]	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-equivalentTextblock
Inventory Policy [HTML]	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-relatedPolicy
Finished Goods	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-oftenContainsConcept
Work in Progress	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-oftenContainsConcept
Raw Material	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-oftenContainsConcept
Inventory	http://www.xbrlsite.com/2015/fro/us-gaap/xbrl/Ontology/rules/arcrole/disclosure-requiresConcept
Investment	
JA: Schedule: Cash and Cash Equivalents, by Component	
JB: Schedule: Reconciliation of Cash per Balance Sheet with Cash Fl...	
JC: Schedule: Reconciliation of Cash per Balance Sheet with Cash Fl...	
JD: Schedule: Receivables, by Current and Noncurrent Portion	
JE: Schedule: Receivables, by Gross/Net Portion	

While the information above is readable by humans, you have to understand XBRL to understand what the rules are saying. That is not particularly difficult, but it can be easier. Below is that same information but the rules have been converted from XBRL technical syntax into a more readable controlled natural language syntax that is intended to be used by professional accountants:



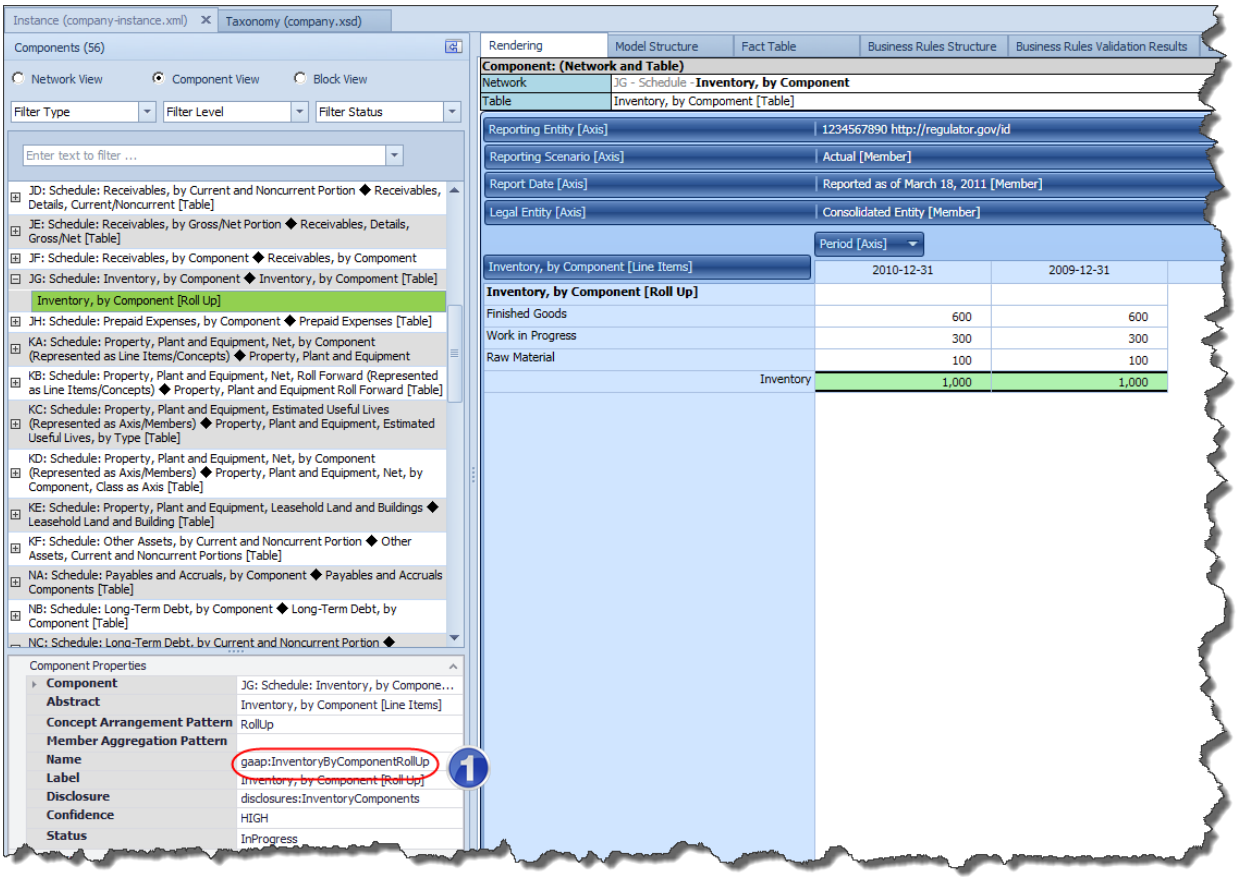
Rules | Line of Reasoning

This disclosure: disclosures:InventoryComponents

- MUST be represented using the Hypercube/[Table] named: gaap:InventoryByCompomentTable
- MUST be represented as a **Level 4 Disclosure Detail** with the concept arrangement pattern: fro:RollUp
  - fro:RollUp REQUIRES total: gaap:Inventory
- Requires the note to be reported using the **Level 1 Note Text Block**: gaap:InventoryHTML
- MUST be represented as using the **Level 3 Disclosure Text Block**: gaap:InventoryComponentsSchedule
- Requires the policy to be reported using the **Level 2 Policy Text Block**: gaap:InventoryPolicyHTML
- OFTEN (not always) contains Level 4 Disclosure Detail Concept: gaap:FinishedGoods
- OFTEN (not always) contains Level 4 Disclosure Detail Concept: gaap:WorkInProgress
- OFTEN (not always) contains Level 4 Disclosure Detail Concept: gaap:RawMaterial
- The RollUp MUST contain the Level 4 Detailed Concept: gaap:Inventory

[36] Disclosure Mechanics rules for the Income Statement in XBRL format, http://xbrlsite.azurewebsites.net/2016/conceptual-model/reporting-scheme/xasb/disclosure-mechanics/1292-rules-def.xml

Those rules specifically and uniquely identify the report fragment of the financial report that is the inventory components. Look at the screen shot below, notice the blue #1 where the disclosure name has been circled in red. This shows that the report fragment which is the inventory components has been identified[37].



And so as you can see, the inventory components disclosure has been identified by the software. The identification of each disclosure works this same way. The Disclosure Mechanics rules[38] provide specific information about each disclosure[39]. These rules provide structural information, mathematical relation information, and other logically oriented information about a disclosure. These Disclosure Mechanics rules serve *three important purposes*:

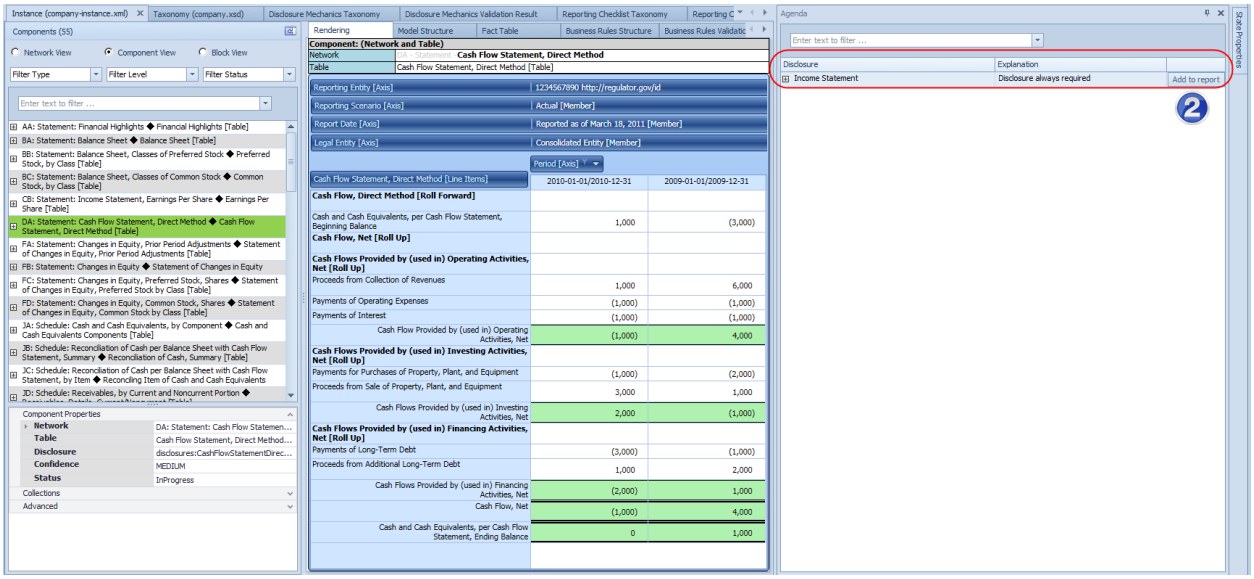
[37] An important thing to note is that because each disclosure is specifically identified by a unique XBRL hypercube (called [Table] in the US GAAP and IFRS taxonomy) it is easy to identify the income statement; all you need to do is find the hypercube called "gaap:IncomeStatementTable". Because in the US GAAP XBRL Taxonomy [Table]s are not unique, this approach cannot be used.

[38] Charles Hoffman, *Understanding Disclosure Mechanics*, http://xbrlsite.azurewebsites.net/2016/Analysis/UnderstandingDisclosureMechanics.pdf

[39] Here are the Disclosure Mechanics rules for all of the disclosures that exist in the imaginary XASB reporting scheme that I created for testing and prototyping, http://xbrlsite.azurewebsites.net/2016/conceptual-model/reporting-scheme/xasb/disclosure-mechanics/disclosure-mechanics-xasb.xsd

- **Identify each disclosure**: Using prototype theory[40], the disclosure mechanics rules are used to help identify which disclosure each report fragment is representing.
- **Verify consistency of disclosure with expectation**: Using the structured nature of XBRL, the disclosure mechanics rules are used to make sure the disclosure is represented consistently with the rules of the reporting scheme, rules of math, logical structure, etc.
- **Taxonomy and report navigation**: Using the structured nature of XBRL and the information in the disclosure mechanics rule, the rules are used to navigate the XBRL taxonomy to find the income statement in the reporting scheme taxonomy.

OK, so now let's delete a disclosure, the income statement report fragment. Because we deleted the report fragment the income statement disclosure that fragment is representing no longer exists in the report. After deleting the income statement, notice the agenda which looks like the screen shot below which lists the income statement:



You can see that the Income Statement appears in the agenda on the right, note the BLUE #2. The income statement is also not in the list of report fragments which are listed on the right.

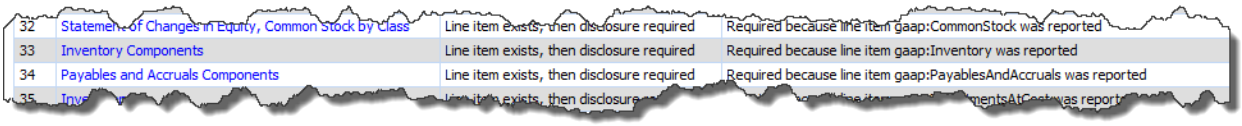
And so now the income statement can be re-imported or re-created, the Disclosure Mechanics rules will help you find the taxonomy elements you might be interested in, the rules will watch over you to make sure you created the income statement correctly, and when you do have the income statement created correctly the entry disappears from the agenda and all of the validation indicators[41] show GREEN which helps you understand that you created the income statement correctly. Alternatively, you can import an income statement template or exemplar provided by the suggest disclosure template/exemplar intelligent agent.

[40] Wikipedia, Prototype Theory, retrieved June 22, 2017, https://en.wikipedia.org/wiki/Prototype_theory
[41] Charles Hoffman, *Blueprint for Creating Zero-defect Digital Financial Reports*, http://xbrlsite.azurewebsites.net/2017/Library/BlueprintForZeroDefectDigitalFinancialReports.pdf

There are two additional pieces of information which are important to understand.

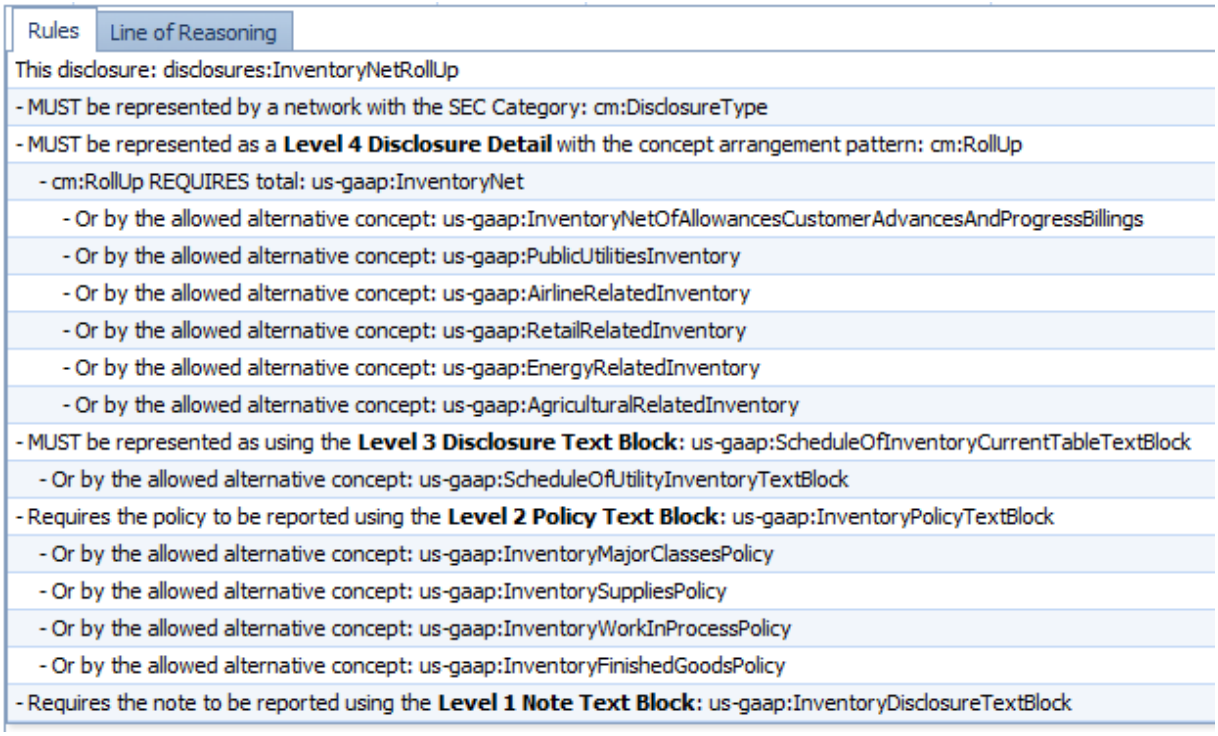
First, some disclosures are only required if a specific line item is reported somewhere in the financial report, typically in a primary financial statement but not always. For example, if the line item "Inventories" is reported on the balance sheet, then the disclosure of inventory components is required per my imaginary XASB reporting scheme and the inventory policy is also required to be disclosed.

Here is the rule that specifies that the disclosure *Inventory Components* is required if the line item "gaap:Inventory" is reported on the balance sheet:



32	Statement - of Changes in Equity, Common Stock by Class	Line item exists, then disclosure required	Required because line item gaap:CommonStock was reported
33	Inventory Components	Line item exists, then disclosure required	Required because line item gaap:Inventory was reported
34	Payables and Accruals Components	Line item exists, then disclosure required	Required because line item gaap:PayablesAndAccruals was reported
35	Inve	Line item exists, then disclosure	mentsAtCost was reporte

The rules specified above relate to the XASB reporting scheme which I created for demonstration purposes. Here is the inventory components disclosure rules specified for the US GAAP reporting scheme in controlled natural language syntax[42]:



Rules | Line of Reasoning

This disclosure: disclosures:InventoryNetRollUp

- MUST be represented by a network with the SEC Category: cm:DisclosureType
- MUST be represented as a **Level 4 Disclosure Detail** with the concept arrangement pattern: cm:RollUp
  - cm:RollUp REQUIRES total: us-gaap:InventoryNet
    - Or by the allowed alternative concept: us-gaap:InventoryNetOfAllowancesCustomerAdvancesAndProgressBillings
    - Or by the allowed alternative concept: us-gaap:PublicUtilitiesInventory
    - Or by the allowed alternative concept: us-gaap:AirlineRelatedInventory
    - Or by the allowed alternative concept: us-gaap:RetailRelatedInventory
    - Or by the allowed alternative concept: us-gaap:EnergyRelatedInventory
    - Or by the allowed alternative concept: us-gaap:AgriculturalRelatedInventory
- MUST be represented as using the **Level 3 Disclosure Text Block**: us-gaap:ScheduleOfInventoryCurrentTableTextBlock
  - Or by the allowed alternative concept: us-gaap:ScheduleOfUtilityInventoryTextBlock
- Requires the policy to be reported using the **Level 2 Policy Text Block**: us-gaap:InventoryPolicyTextBlock
  - Or by the allowed alternative concept: us-gaap:InventoryMajorClassesPolicy
  - Or by the allowed alternative concept: us-gaap:InventorySuppliesPolicy
  - Or by the allowed alternative concept: us-gaap:InventoryWorkInProcessPolicy
  - Or by the allowed alternative concept: us-gaap:InventoryFinishedGoodsPolicy
- Requires the note to be reported using the **Level 1 Note Text Block**: us-gaap:InventoryDisclosureTextBlock

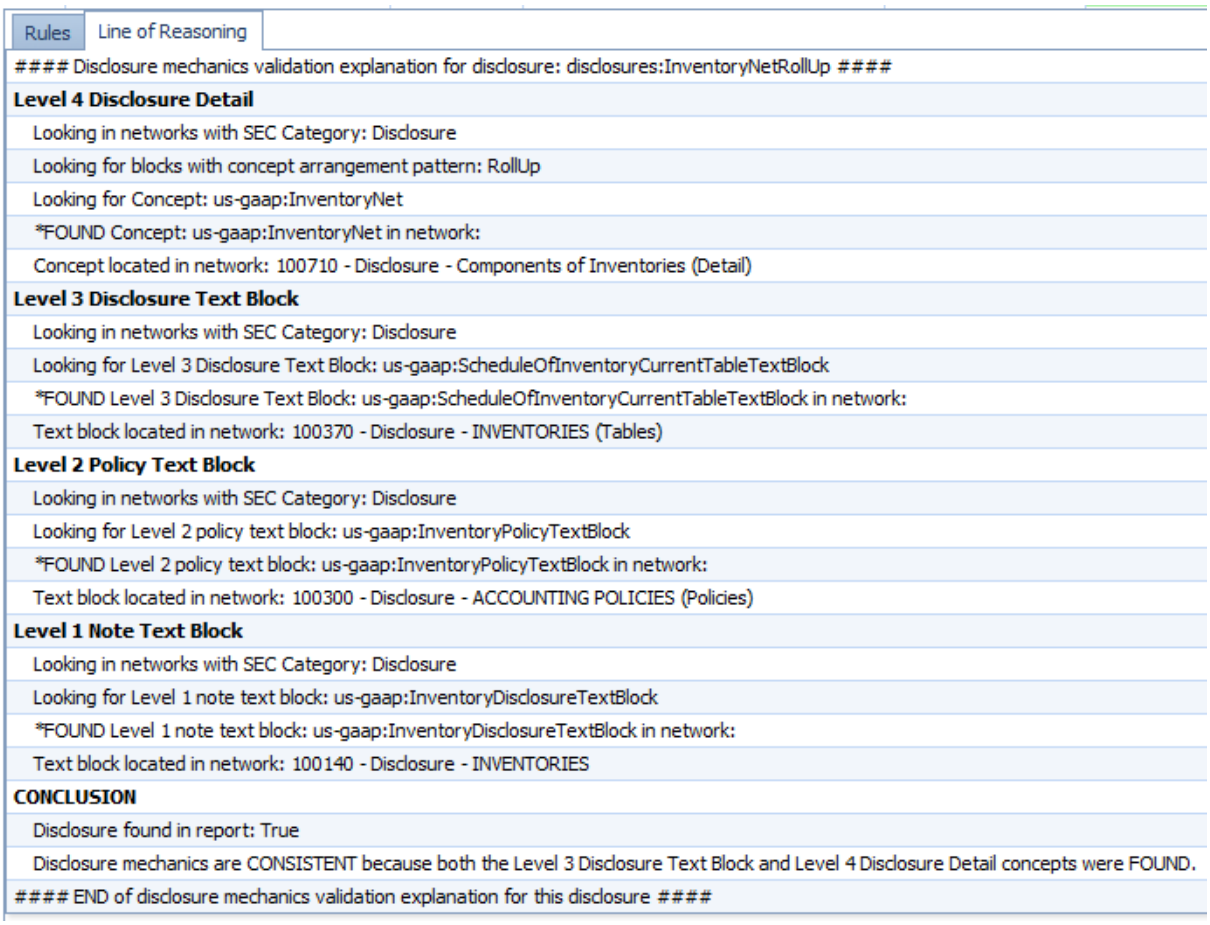
So the above rule can be used to check many inventory related disclosures. But you have to specify the rules for automated processes to check the report against the rules. The only way that you can check to see if the report is created correctly is to do so *manually* if machine-readable rules are not provided.

[42] Here are the rules in the XBRL technical syntax, XBRL definition relations, http://xbrlsite.azurewebsites.net/2016/conceptual-model/reporting-scheme/us-gaap/disclosure-mechanics/517-rules-def.xml

This is an important concept to understand: no machine readable rule, no automated check. It is impossible to specify 100% of everything that must be checked in machine readable form. As such, manual effort will always be necessary. But manual effort is required if you don't specify a rule either.

Finally, the disclosure mechanics rules are not yet totally complete, they still need work. Here are two examples of things that need to be added. The first example relates to materiality. What if inventory is not material? That is not a problem; you just specify the materiality threshold to the test within the disclosure mechanics rule. Say, 5% or 10% of total assets. That is not difficult for the software to automate. Or what if inventory components were disclosed on the balance sheet rather than in a disclosure? Again, this is not a problem. You just have to modify the algorithm the software uses to see if the representation is consistent or inconsistent with your expectations. Are expectations ambiguous? Again, not a problem…then your rules can be ambiguous also.

Below is the line of reasoning used by the software algorithm as it tries to find the inventory components related disclosure items in a report:



Rules	Line of Reasoning
#### Disclosure mechanics validation explanation for disclosure: disclosures:InventoryNetRollUp ####	
Level 4 Disclosure Detail	
Looking in networks with SEC Category: Disclosure	
Looking for blocks with concept arrangement pattern: RollUp	
Looking for Concept: us-gaap:InventoryNet	
*FOUND Concept: us-gaap:InventoryNet in network:	
Concept located in network: 100710 - Disclosure - Components of Inventories (Detail)	
Level 3 Disclosure Text Block	
Looking in networks with SEC Category: Disclosure	
Looking for Level 3 Disclosure Text Block: us-gaap:ScheduleOfInventoryCurrentTableTextBlock	
*FOUND Level 3 Disclosure Text Block: us-gaap:ScheduleOfInventoryCurrentTableTextBlock in network:	
Text block located in network: 100370 - Disclosure - INVENTORIES (Tables)	
Level 2 Policy Text Block	
Looking in networks with SEC Category: Disclosure	
Looking for Level 2 policy text block: us-gaap:InventoryPolicyTextBlock	
*FOUND Level 2 policy text block: us-gaap:InventoryPolicyTextBlock in network:	
Text block located in network: 100300 - Disclosure - ACCOUNTING POLICIES (Policies)	
Level 1 Note Text Block	
Looking in networks with SEC Category: Disclosure	
Looking for Level 1 note text block: us-gaap:InventoryDisclosureTextBlock	
*FOUND Level 1 note text block: us-gaap:InventoryDisclosureTextBlock in network:	
Text block located in network: 100140 - Disclosure - INVENTORIES	
CONCLUSION	
Disclosure found in report: True	
Disclosure mechanics are CONSISTENT because both the Level 3 Disclosure Text Block and Level 4 Disclosure Detail concepts were FOUND.	
#### END of disclosure mechanics validation explanation for this disclosure ####	

Again, professional accountants will never be free from 100% of the manual drudgery of detecting whether the math of a report is correct, if all necessary disclosures are provided, or other logical or regulatory requirements have been properly met. But, there is a lot that can be automated. The agenda helps you manage your report creation process to the degree that machine-readable rules can

be provided and therefore are leveragable by software applications in the financial report creation process.

These rules for inventory components were tested against 100% of 10-Ks filed in 2015 for all 6,466 public companies that report to the SEC. Note that the new rule (above) is more comprehensive than the older initial rules (below) because information from this analysis was used to improve the rules. Here are the results:

DisclosureFound	DisclosureConsistent	RepresentationConcept_TextBlock	RepresentationConcept_Detail	Count	Percent	
FALSE	CONSISTENT	NOT-FOUND	NOT-FOUND	3,612	56%	1
TRUE	CONSISTENT	us-gaap:ScheduleOfInventoryCurrentTableTextBlock	us-gaap:InventoryNet	1,721	27%	2
TRUE	INCONSISTENT	NOT-FOUND	us-gaap:InventoryNet	1,061	16%	3
TRUE	INCONSISTENT	us-gaap:ScheduleOfInventoryCurrentTableTextBlock	NOT-FOUND	46	1%	4
TRUE	CONSISTENT	us-gaap:ScheduleOfUtilityInventoryTextBlock	us-gaap:InventoryNet	19	0%	5
TRUE	INCONSISTENT	us-gaap:ScheduleOfUtilityInventoryTextBlock	NOT-FOUND	7	0%	6
				6,466	100%	7
			Consistent with expectation	5,352	83%	
			Inconsistent with expectation	1,114	17%	
			Total	6,466	100%	

Here is how you interpret each line item; the numbers below correspond to the red numbers on the right hand side of the diagram above:

- **Line #1** indicates that NEITHER the Level 3 Disclosure Text Block nor the Level 4 Disclosure Details were found, meaning this disclosure is not present and is **consistent with expectations** should the disclosure not exist in the report. Basically, 56% of public companies do not report inventory.
- **Line #2** indicates that BOTH the Level 3 Disclosure Text Block AND the Level 4 Disclosure Detail line items WERE found. This is **consistent with expectations**. A total of 27% of public companies report using these concepts.
- **Line #3, Line #4, and Line #7** indicates an inconsistency because either the Level 3 Disclosure Text Block was found OR the Level 4 Disclosure Detail was found but NOT BOTH. Thus, those lines are **inconsistent with expectations**.
- **Line #5** indicates that 19 companies used an ALTERNATIVE Level 3 Disclosure Text Block to report the inventory components roll up, but the same Level 4 Disclosure Detail concept. This is **consistent with expectations**.
- **Line #7** indicates the total population of public company financial reports analyzed, which is 6,466 10-K filings as of March 31, 2016.

So, those are the results for this one disclosure. This same process was used for about 64 other disclosures. These disclosures will be further tested during my campaign to improve the disclosure quality of XBRL-based reports submitted to the SEC[43].

Don't fear intelligent machines such as knowledge-based software for creating financial reports. Gain the skills necessary to work with these machines. Humans augmented by machine capabilities, much like an electronic calculator enabling a human to do math quicker, will empower knowledge workers who know how to leverage the use of those machines.

If you want to see the application that I have shown in the screen shots working, please watch this video below or read the blog post that describes knowledge-based systems in more detail[44]:

https://www.youtube.com/watch?v=ioapxF1gtQM

Comparison of Financial Reporting or Business Reporting Solutions

Pesseract is a specific tool as contrast to a general tool; we discussed the difference between specific and general earlier. Pesseract uses a global standard technical syntax, XBRL. Contrasting Pesseract to other reporting solutions helps you better understand Pesseract and what you need from such a tool to be able to do.

Semantic Web Stack Reporting Solution (TopQuadrant)

The software company TopQuadrant offers a product called *Data Governance and Vocabulary Management: Customizable Workflows*[45]. In my personal opinion, this is the sort of functionality business people would need/want from a business reporting product. This is basic functionality necessary for any business reporting system. This is their product description:

"Users can now create and use custom workflows for various data governance and curation processes. Pre-built workflow templates are also provided. Enhanced Operating Model – TopBraid EDG's data governance operating model options now provide access to business and data subject areas, associated governance roles, organizational structures, issues, policies, metrics, dashboard, and other governance assets. Advanced Data Profiling – Data source profiling is implemented when its metadata is imported into TopBraid EDG. This enables automated reasoning about data and connections of technical metadata to business metadata. Interactive Visualizations – Updated versions of TopBraid EDG's unique LineageGram and NeighborGram features with intuitive capabilities for flexible viewing and drilling down on focused areas of governance knowledge graphs."

---

[43] *Campaign to Improve Disclosure Quality of XBRL-based Public Company Financial Reports Submitted to the SEC*, http://xbrl.squarespace.com/journal/2017/10/18/campaign-to-improve-disclosure-quality-of-xbrl-based-public.html

[44] Charles Hoffman, *World's First Expert System for Creating Financial Reports*, http://xbrl.squarespace.com/journal/2017/4/27/worlds-first-expert-system-for-creating-financial-reports.html

[45] *Data Governance and Vocabulary Management: Customizable Workflows*, http://www.dataversity.net/topquadrants-topbraid-5-4-release-delivers-real-world-data-governance-enterprises/

One limitation, a shortcoming of the TopQuadrant product in my view, is that they are focused too much on **internal information exchange** and should provide better functionality for **external statutory and regulatory reporting**. When you deal with external statutory and regulatory reporting use cases, the risk of noncompliance increases and so that risk needs to be minimized.

TopQuadrant is helping me create an RDF/OWL/SHACL model of the information conveyed by an XBRL-based financial report. This will allow XBRL to be converted from/to RDF/OWL/SHACL.

TopQuadrant is a general tool which one can use to solve specific problems. TopQuadrant's product is way, way, way too hard for business professionals to use or even get their heads around. But TopQuadrant has what they call a "Custom Workflow". Technical people can build, for business people functionality that is easier for the business people to use; creating a specific solution which is easier to use.

Pesseract could serve as an easy-to-use interface to the TopQuadrant reporting product overcoming the ease of use issue. Pesseract is focused on the needs of external statutory and regulatory reporting. Perhaps a partnership can be created with TopQuadrant where the best functionality of each product can be leveraged providing the market with an excellent but flexible and powerful business reporting solution.

Business Logic Platform (FlexRule)

The product FlexRule[46] is another reporting solution. FlexRule is positioned as a "business logic platform." This is a description of the FlexRule product:

"**Automate and Connect Anything and Everyone!** Processes deliver the end to end flow of work. This covers everything from Robo-Services, Applications, Business Events, Reporting to Transactional Systems. With Process Robotics, FlexRule can automate the end to end process steps. This delivers business outcomes that uses consistent decision logic via Decision Automation leveraging Data Connection & Composition. FlexRule mimics human and system processes and workflow including long running transactions and escalations to the appropriate people."

FlexRule is likewise a complete business reporting solution. But, it too is a general solution. FlexRule does not have an inherent understanding of the XBRL technical syntax, but you can use FlexRule with XBRL. But, you have to take responsibility for developing a high-level model in the solution you create because that high-level model is not inherently available in FlexRule.

Business Logic System Gap Analysis

The following is a comparison and gap analysis that highlights the capabilities of numerous business reporting systems/platforms or business logic systems/platforms[47]:

[46] FlexRule, http://www.flexrule.com/solution/

[47] *Business Logic System Gap Analysis*, http://xbrlsite.azurewebsites.net/2017/Library/BusinessLogicSystemGapAnalysis.jpg

Business Rules Processor	For more information	Information Syntax/Format	XBRL Processor	XBRL Formula Processor	Supports Inference	Problem Solving Method	Supports Probabilistic Reasoning	State Machine	General Problem Solving Logic Capabilities	Fact Database	Knowledge Base	Rule Creation Interface (XBRL Taxonomy, XBRL Formula, Other)	Understanding of Business Report Conceptual Model	Understanding of Financial Report Conceptual Model	Justification and Explanation Mechanism (Transparency into Line of Reasoning and Information Provenance)	Knowledge Acquisition Mechanisms
Arelle (Open source API level interface)	http://arelle.org/	Standard XBRL	Yes	Yes	No	Sequential	No	No	INCOMPLETE, Limited to XBRL Formula	XBRL Instance	XBRL Taxonomy	None provided	No	No	Not provided, can be created	Manual
XBRL Development Tools (Altova)	https://www.altova.com/xbrl-tools	Standard XBRL	Yes	Yes	No	Sequential	No	No	INCOMPLETE, Limited to XBRL Formula	RaptorXML+XBRL Server	RaptorXML+XBRL Server	XBRL Specific but oriented to technical users	No	No	Not provided, can be created	Manual
Sphinx (CoreFiling)	https://www.corefiling.com/products/sphinx/	Standard XBRL	Yes	Yes	No	Sequential	No	No	INCOMPLETE, Limited to XBRL Formula	XBRL Instance	XBRL Taxonomy	XBRL Specific but oriented to technical users	No	No	Unknown	Manual
Clean Score (XBRL Cloud)	https://www.xbricloud.com/cleanscore.html	Profile based Standard XBRL	Yes	Yes	Yes	Sequential	No	No	GOOD (Subset of RuleLog)	XML Infoset stored in file system	XML Infoset stored in file system	None provided	Yes	Yes	Good, usable by business professionals	Manual
Pesseract Knowledge Based Financial Report Creation System	http://pesseract.azurewebsites.net/	Profile based Standard XBRL	Yes	No	Yes	Forward chaining	No	Yes	GOOD (Subset of RuleLog)	XML Infoset stored in file system	XML Infoset stored in file system	None provided at present time, will be business user oriented	Yes	Yes	Good, usable by business professionals	Manual
FlexRule Business Logic Platform	http://www.flexrule.com/solution/	Proprietary or general format	No	No	Yes	Forward chaining	No	No	BETTER (Larger subset of RuleLog)	Proprietary or general format	Proprietary or general format	Comprehensive but oriented to technical users	No	No	Unknown	Manual
InRule (InRule Technologies)	http://www.inrule.com/products/inrule/	Proprietary or general format	No	No	Yes	Forward chaining	No	No	BETTER (Larger subset of RuleLog)	Proprietary format	Proprietary format	Oriented toward non-technical users and business professionals	No	No	Unknown	Manual
Smarts (Sparkling Logic)	https://www.sparklinglogic.com/smarts-decision-manager/	Proprietary or general format	No	No	Yes	Forward chaining	No	No	BETTER (Larger subset of RuleLog)	Proprietary or general format	Proprietary or general format	Comprehensive but oriented to technical users	No	No	Unknown	Manual
Fluent Editor (Cognitum)	http://www.cognitum.eu/semantics/FluentEditor/	Standard RDF, OWL, SWRL	No	No	Yes	Forward chaining	No	No	BETTER (Larger subset of RuleLog)	RDF stored in file system	RDF, OWL, SWRL stored in file system (Semantic Web Stack)	Comprehensive but oriented to technical users	No	No	Unknown	Manual
TopBraid Platform (TopQuadrant)	https://www.topquadrant.com/technology/topbraid-platform-overview/	Standard RDF, SHACL, RDFS, OWL, SPIN	No	No	Yes	Forward and Backward chaining	Yes	No	BEST (RuleLog plus)	RDF triple store repository	SHACL, RDFS, OWL, SPIN stored in file system (Semantic Web Stack)	Comprehensive but oriented to technical users; rule creation templates usable by business	No	No	Good, usable by business professionals	Manual or Automated
Ergo Logic System (Coherent Knowledge)	http://coherentknowledge.com/financial-domain-application/	Standard RDF, RIF	No	No	Yes	Forward and Backward chaining	Yes	No	BEST (RuleLog plus)	RDF stored in file system	RDF, OWL, RIF stored in file system (Semantic Web Stack)	Comprehensive but oriented to technical users	No	No	Good, electronic audit trail with provenance usable by business professionals	Manual or Automated

Each system/platform has its pros and cons and not one system/platform is perfect. Comparing and contrasting these different solutions helps you understand the pros and cons of different solutions.

Next steps for Pesseract, the product roadmap

As we said, **Pesseract is a commercial quality proof-of-concept; it is not yet a product**. The purpose of Pesseract is to determine the feasibility of leveraging a conceptual model and using that high-level model to create a powerful yet easy to use software application that is approachable by business professionals. We believe that we have successfully proven the feasibility. The following is a description of what we want Pesseract to be:

> "Pesseract is purpose-build software tool for disclosure management and financial report creation. Pesseract is a platform and a framework of intelligent XBRL-based digital financial reporting products which collect information about financial report creation projects and allow this information to be coordinated across all other representations of the project, so that every statement, policy, and disclosure is based on internally consistent and complete information from the same underling financial information database. The result is zero defect financial reports created using efficient and effective processes."

Next phase in roadmap

The following is a summary of the next steps for Pesseract. The primary goal is to complete a product that can be used to create/edit/import an entire report and review that report for accuracy for all existing supported profiles:

1. **Refactor** any underlying technical functionally or architecture that needs to be refactored taking into consideration what we now know about what Pesseract needs to ultimately be.
2. Create, tune, and synchronize the business rules engine so that it provides all the functionality of a standard **XBRL Formula processor** plus all of the functionality and capabilities that are necessary but not part of an XBRL Formula Processor[48]. Specifically, this includes:
    - Inference
    - Access to all XBRL taxonomy structural and other information
    - Forward chaining
    - Maximum expressive power of problem solving logic but not allowing any logical catastrophes
3. **Save** a report and the related taxonomy.
4. Support the **creation/edit of reports** leveraging the "Legos" and "snap together" and "business rules as glue" ideas articulated in this document.
5. **Import fact tables** from Excel.
6. **Export RDF fact table**.
7. **Copy/paste of a Block** from one report to another report.
8. Complete the **suggest disclosure template/exemplar intelligent agent**.
9. Search, filter and other functionality to **locate report elements in a base taxonomy.**
10. **Bug fixes** of existing functionality.
11. **To do tracking**.

[48] *Specific Deficiencies in Capabilities of Existing XBRL Formula Processors*, http://xbrl.squarespace.com/journal/2016/9/26/specific-deficiencies-in-capabilities-of-existing-xbrl-formu.html

12. **Report properties**.

Features to be added in subsequent phases in product roadmap

The following is a summary of other feature which will ultimately be included in Pesseract. **This functionality is specifically not targeted for the next development phase of Pesseract**: (in no specific order)

- Pass all XBRL 2.1, XBRL Dimensions, and XBRL Formula conformance suites tests.
- SEC EFM validation.
- Other minimum criteria validation.
- Improve current dynamic rendering in application.
- Support for ESMA reporting profile.
- Support for State and Local Government (GASB) reporting profile.
- Convert a Level 4 Disclosure Detail to a Level 3 Disclosure Text Block.
- Convert a set of Level 3 Disclosure Text Blocks to Level 1 Note Text Blocks.
- Serialize an entire report (a set of Level 1 Note Text Blocks) to HTML, Microsoft Word, and PDF.
- Add additional business rule capabilities.
- Export RDF, OWL, and SHACL.
- Import RDF, OWL, and SHACL.
- Localizations for the languages Spanish, Chinese, Japanese, Persian, Arabic, French.
- Efficient and scalable fact database.
- Analysis features (query fact database, comparisons, etc.).
- Document publishing features including Inline XBRL.
- Base taxonomy creation features.
- Business rules creation interface and capabilities (FAC, Disclosure Mechanics, Reporting Checklist, XBRL Formula).
- Enhanced Reporting Checklist.
- Build that supports Mac OS.
- Multi-user support.