1. Guide to Building Expert System for Creating Financial Reports

One type of practical knowledge is **know-how**; how to accomplish something. This document explains how to accomplish something.

The purpose of this section is to explain how to construct an expert system that can be used by professional accountants to create internal and external financial reports. This document is intended to be read by software engineers and professional accountants; its primary role is to help those two groups of professionals from different domains communicate effectively about this topic.

Today, general purpose financial reports are typically created using Microsoft Word which is augmented by perhaps tens if not hundreds of Microsoft Excel spreadsheets depending on the size of the organization and complexity of the financial report.

We have taken a new approach to creating such reports. This new approach to creating financial reports throws out old-school paradigms of creating such general purpose financial reports altogether. It is a "greenfield" project employing idealized redesign. Idealized redesign[1] is a simple philosophy: imagine the ideal solution and work backward from there to where you are today. This philosophy eliminates imaginary constrains and obstacles before they are erected in one's mind.

The end result is identical to today's general purpose financial report which is to communicate financial information of an entity to investors, analysts, regulators, and other parties who use that information. But this new approach leverages modern technologies such as structured information, artificial intelligence, machine readable business rules, rules engines, global standards, the philosophies of Lean Six Sigma, and other such technologies, techniques, processes, procedures, and philosophies to completely redefine the process of creating general purpose financial reports.

This document is not based on speculation and it does not try and imagine what such a tool might look like. What this document does is explain a working proof-of-concept that was constructed to figure out exactly how to build such an expert system for general purpose financial report creation. The name we gave to this application is *Pesseract*[2]. Pesseract was created by Hamed Mousavi and me. You can download and experience the application for yourself for non-commercial use. This document explains techniques we used to create Pesseract.

So, Pesseract was created by a professional accountant and a software engineer collaborating to figure out the moving pieces of the puzzle and how to put those pieces together effectively to create a system that is simple to use but sophisticated and effective in terms of functionality. We both had to figure out the domain of knowledge engineering.

Professional accountants and other business professionals will team with computers more to get work done in today's digital environment. This teaming is natural. This teaming is much like how professional accountants use calculators to augment their

[1] Wharton School, University of Pennsylvania, *Idealized Redesign: How Bell Labs Imagined — and Created — the Telephone System of the Future*, http://knowledge.wharton.upenn.edu/article/idealized-design-how-bell-labs-imagined-and-created-the-telephone-system-of-the-future/

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

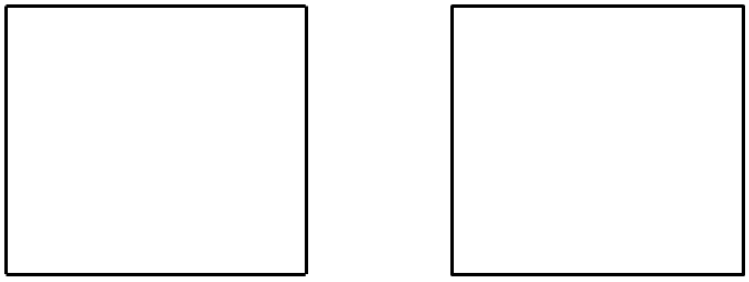
capabilities to perform mathematics more efficiently. Accounting, reporting, auditing, and analysis will be carried out in a digital environment.

One very critical point should be made clear. This document is not about building a tool for creating XBRL. XBRL is a global standard technical syntax. XBRL is only one technology employed to attain the desired goal. This document is about creating a tool for creating digital general purpose financial reports. Those reports can then be rendered in traditional human readable formats as HTML, PDF, or Word; or in machine-readable formats such as XBRL, RDF, JSON, or any other technical syntax which might be in fashion which you might desire including Inline XBRL. The focus here is financial reports, not technical syntax.

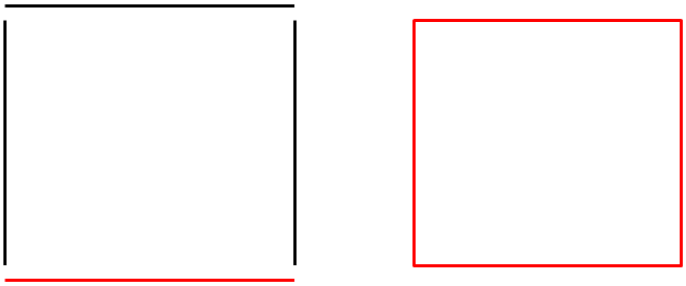
Finally, a financial report is a type of business report. Things that have been learned about creating XBRL-based financial reports can be used to understand the more general use case of XBRL-based business reports. These techniques work well for implementing accounting process automation.

1.1. 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.



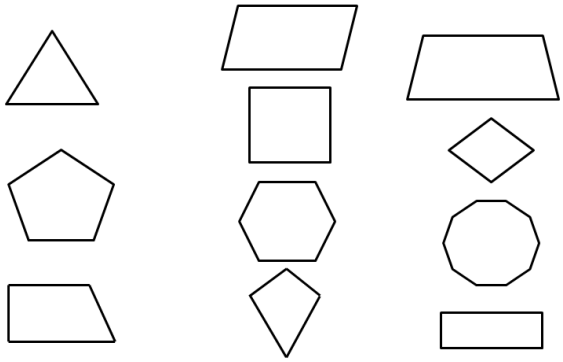
Below is another view of those same two objects. Note that one is really four lines and the other is a square:



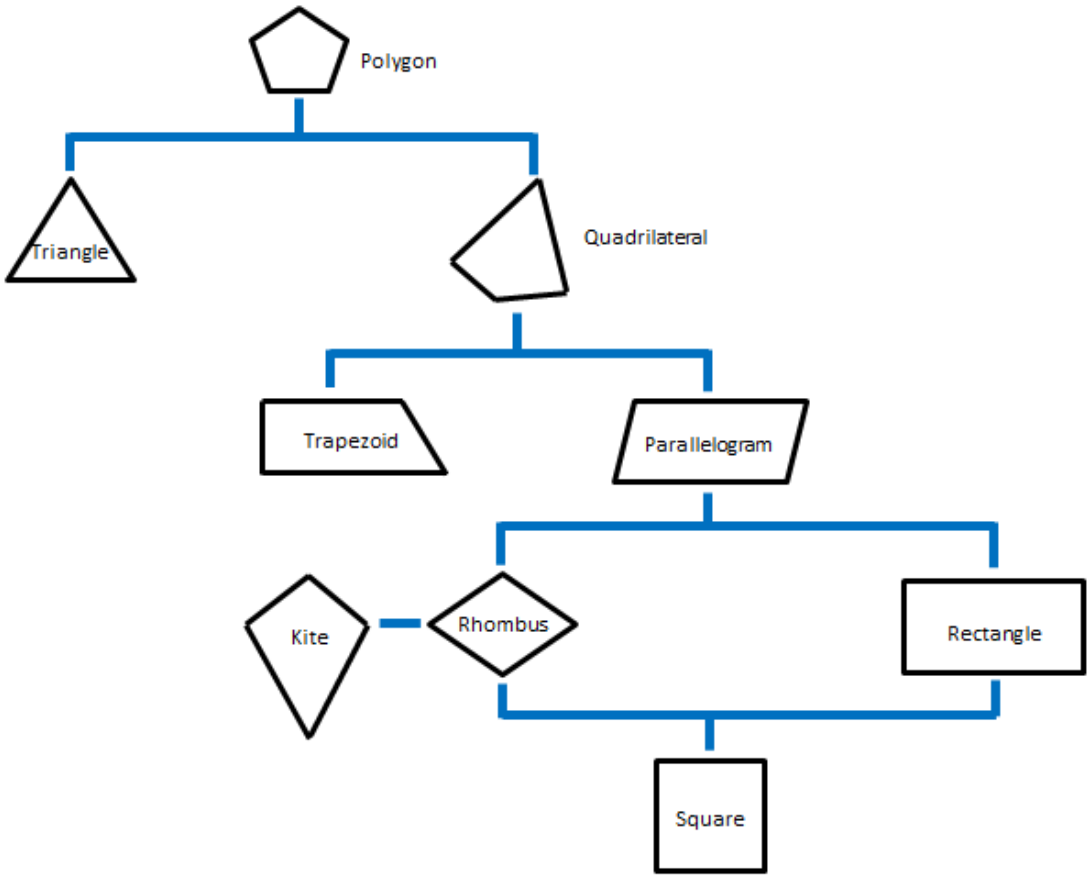
Imagine having to work with low-level objects in an application such as PowerPoint. To make your life easier, PowerPoint provides high-level objects that you can use to perform work rather than requiring you to create your presentations using only low-level objects. These higher-level objects are "compound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

1.1.1.Categories

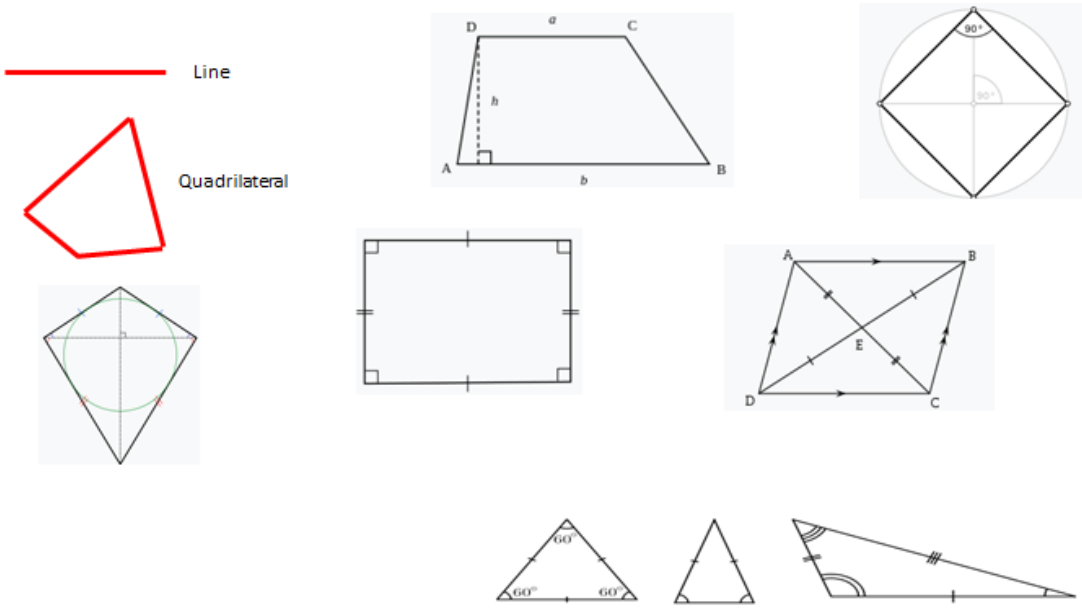
Here are a bunch of different compound or composite objects, all of which are made out of lines:



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:

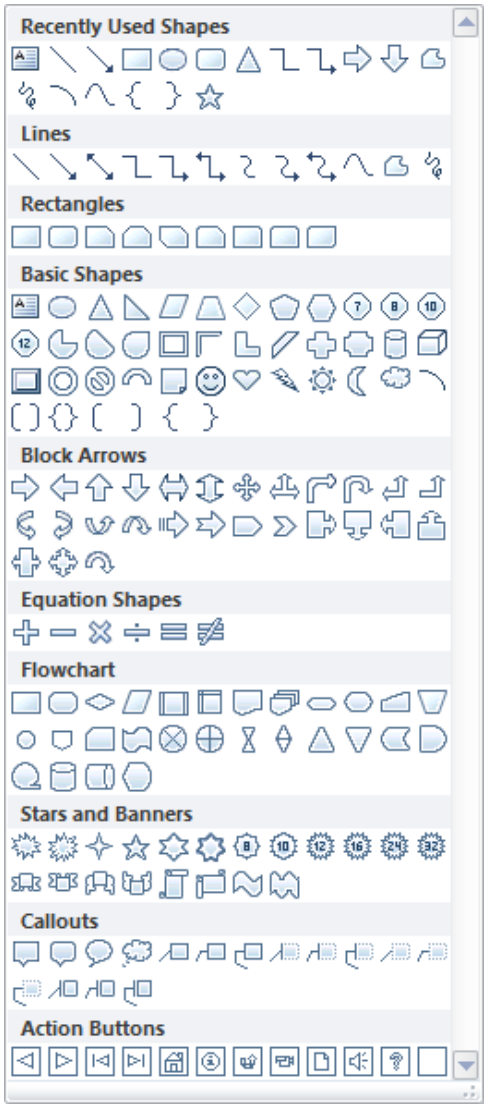


Objects can be described and identified by their **properties**. Note that lower-level objects like a line have fewer properties. Note that a quadrilateral has no properties other than that it is made up of four lines. But other objects are rich with properties. Object properties are **universal business 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.

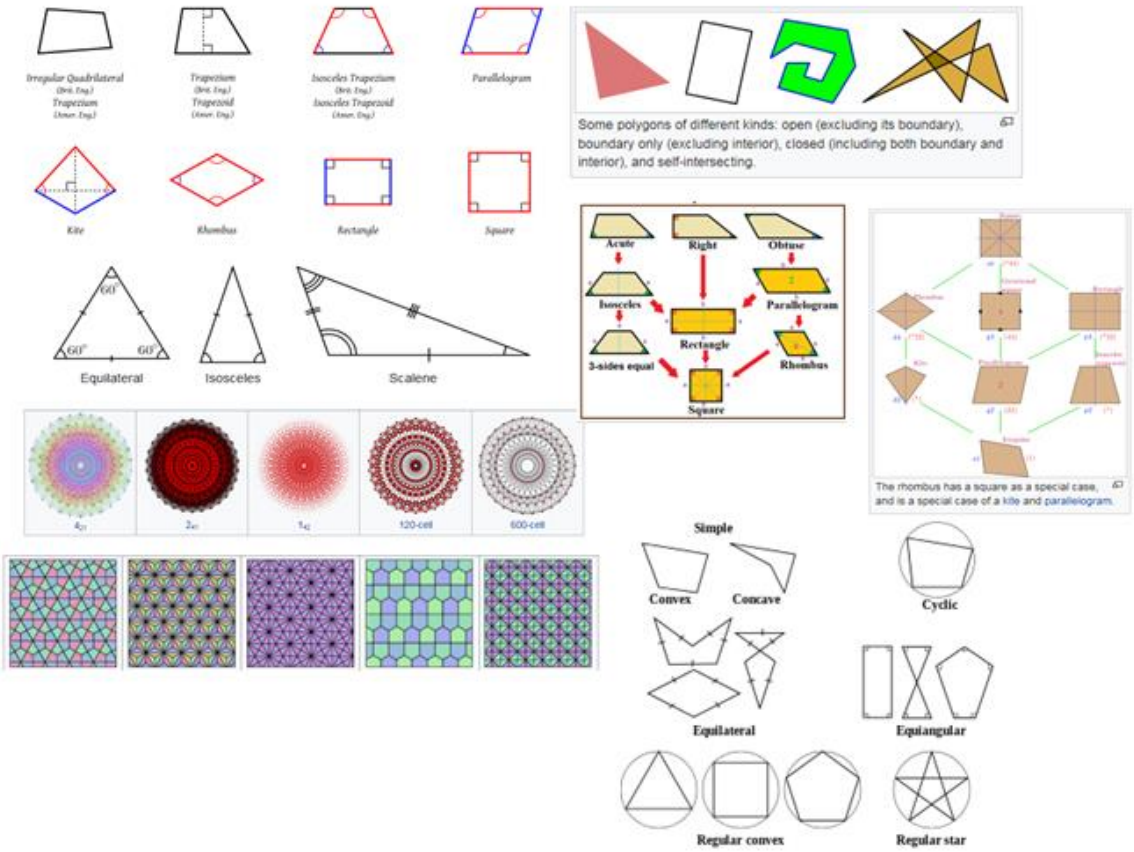
Once objects are identified, named, their properties identified, and objects classified they can be organized within software applications. Below is an example from Microsoft PowerPoint. The interface helps you work with categories of higher-level objects; rather than lower-level pieces. You can always 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.



1.1.2.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*.

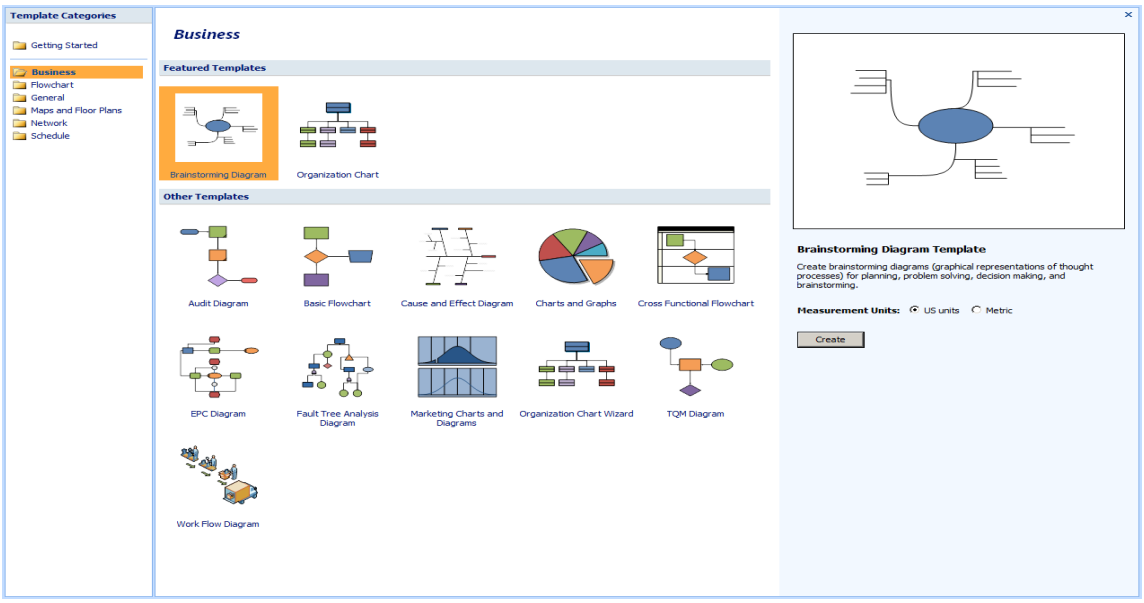
If you need more flexibility than what a square offers, create a new category of object and give it a name and defines that new object's properties as you may deem appropriate. Make things flexible where they need to be flexible. Don't just create general flexibility to play it safe…that makes working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

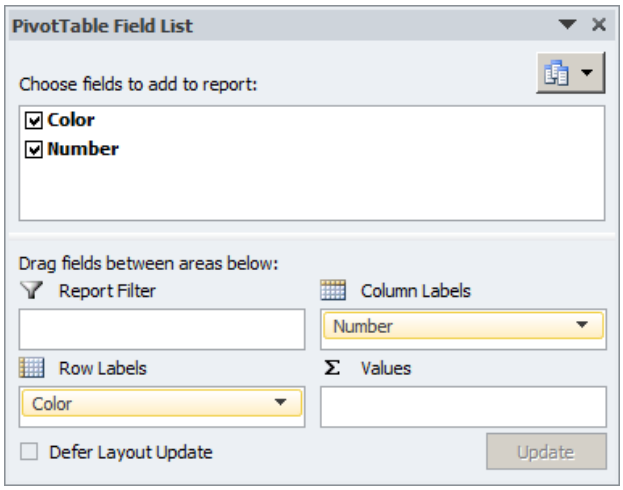
1.1.3. Templates

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:



1.1.4.Snapping pieces together like Legos

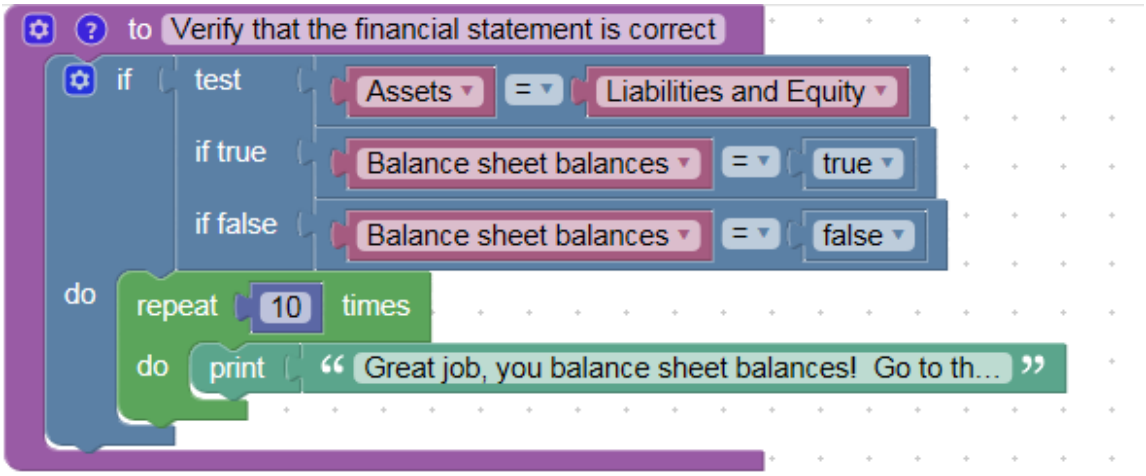
Business professionals work with objects at the highest possible level, snapping pieces together like Legos and watched over the objects by the software application leveraging the patterns and business rules which helps make sure business professionals don't make mistakes where that task is possible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



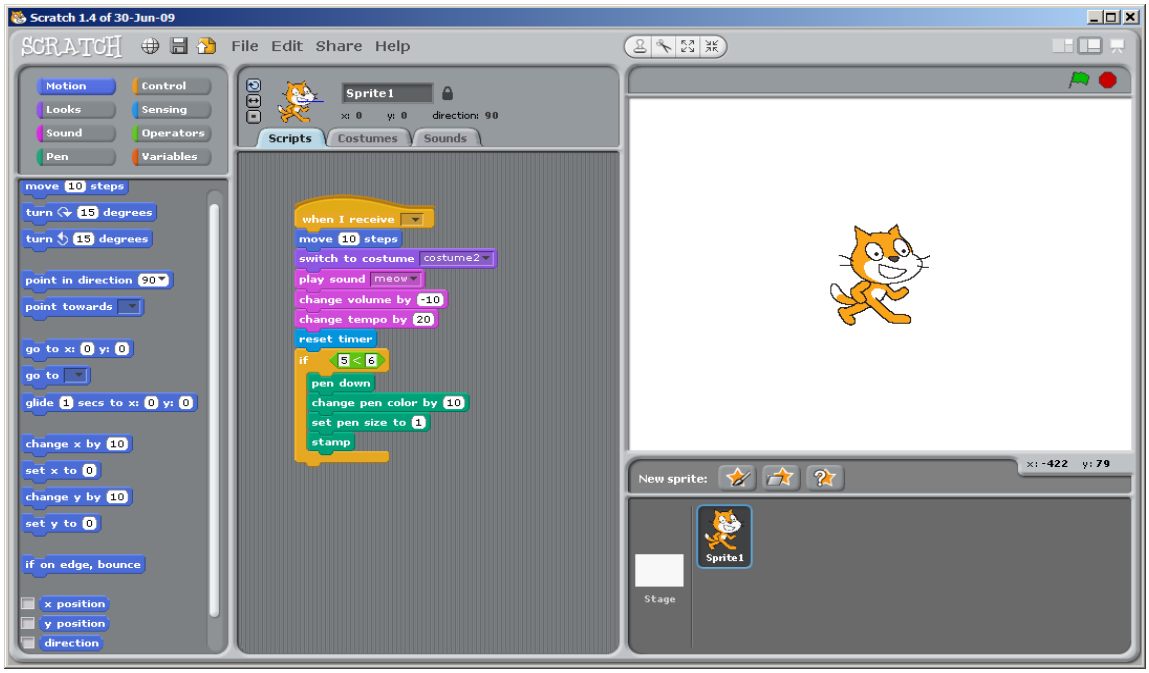
Keep the notion of a pivot table in the back of your mind. We will explain why an Excel pivot table is not useful for what we need to do unfortunately. But the general model of a pivot table is good. It is just that the pivot table functionality needs to be slightly different. Imagine this notion of "Legos" or "pieces that snap together". See this example of Blockly[3] below. Blockly is based on ideas from Scratch which was created by MIT[4].

[3] Blockly, http://xbrl.squarespace.com/journal/2014/7/14/blockly.html

[4] Scratch, http://scratch.mit.edu/



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:



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.

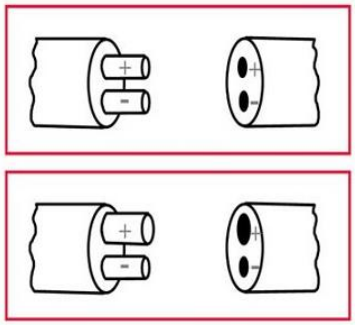
1.1.5.Poka-yoke: Mistake proofing software

Poka-yoke[5] 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

[5] Wikipedia, *Poka-yoke*, https://en.wikipedia.org/wiki/Poka-yoke

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[6] 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.

1.1.6.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[7]. 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 multidimensional model[8] 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.

1.2. Understanding Key Requirements

One very important notion to understand is that the objective of this software application is to employ useful technologies within the software application, but to bury the technology deeply within the software application exposing business users of the application to business domain related choices which they understand.

---

[6] Process Exam, *Six Sigma Tools - Poka Yoke*, http://www.processexam.com/six-sigma-tools-poka-yoke

[7] 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)

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

Technical choices have been absorbed by the software application using good design, best practices, and clever and creative software development techniques. The *Law of Conservation of Complexity* states in essence that complexity can never be removed from a system, but complexity can be moved. Every software application has an inherent amount of irreducible complexity. The question is who will deal with that complexity: the user, the application developer, or the platform developer? Anyone can create something that is complex. But it takes hard work to create something simple. Simple and simplistic are not the same thing.

Simplistic is dumbing down a problem in order to make the problem easier to solve. Simplistic ignores complexity in order to solve a problem which can get you into trouble. Simplistic is over-simplifying. Simplistic means that you have a naïve understanding of the world, you don't understand the complexities of the world. Removing or forgetting complicated things does not allow for the creation of a real world solution that actually works.

Simple is something that is not complicated, that is easy to understand or do. Simple means without complications. An explanation of something can be consistent with the real world, consider all important subtleties and nuances, and still be simple, straight forward, and therefore easy to understand.

The goal of this software is not to provide an incremental improvement to parts of the process of creating a financial report. The goal of this software is to improve the whole system of creating financial reports, a holistic improvement. The objective is to have a tool that makes sense when doing accounting, reporting, auditing, and analysis in a digital environment.

The following is a summary of the key requirements of an expert system for creating general purpose financial reports. These requirements are the constraints imposed on the system for creating general purpose financial reports.

1.2.1. Verifiably True and Fair Representation of Reported Financial and Nonfinancial Information

A primary requirement of this system is quality of the end result and an ability to explicitly verify that quality. There are significant ramifications and consequences for noncompliance with statutory and regulatory rules.

There are two categories of information which impact quality: *objective* and *subjective*. **Objective** information is mechanical and can be easily tested using computer processes. For example, whether a mathematical computation is correct is objective. **Subjective** information cannot be tested using computer processes. For example, if two or more allowed alternative approaches exist and there is no means of determining which is the best alternative, then one cannot be incorrect if one chooses alternative 1 as contrast to alternative 2. While computer processes can assist users in determining between subjective alternatives, the final choice is a matter of professional judgment of a trained accountant.

The systems objective is to automate as many objective decisions as possible, basically making it impossible for a business professional to make a mathematical or mechanical error. Those objective rules can be verified to be correct using automated machine-based processes. An explanation and justification mechanism must be provided not only for objective rules and also subjective information which must be verified by a professional accountant using manual processes.

Engineer and statistician W. Edwards Deming[9] defined quality as "predictability," and called variance "the enemy of quality." To achieve an intended outcome, Deming thought it was important to plan for **common-cause variation**, which can be predicted, and **special-cause variation**, which cannot be predicted.

Harold F. Dodge, one of the principal architects of the science of statistical quality control, said, "You cannot inspect quality into a product." In other words, once the inspection takes place, it's too late. Rather, data from the quality inspection needs to be utilized to continually improve the process.

Management consultant Joseph Juran, who focused on management training and the human element of quality control for a variety of businesses, stated that quality is "a fitness for use."

Businessman Philip B. Crosby, who developed the concept of Zero Defects while working as senior quality engineer at aircraft manufacturer The Martin Company, defined quality as "a conformance to requirements." He warned against the **high cost of nonconformance** and said that the desired performance standard of zero defects could only be achieved through the **proper management system**.

1.2.2.Representation and Presentation of Information

Reported information must be able to be represented in machine-readable form and presented in human-readable form. Traditional human readable formats such as Word, HTML, and PDF must be supported. As an option, reports can also be generated in machine readable serializations such as XBRL, JSON, RDF, etc. Or, reports could be rendered in newer human-readable presentations that are also readable by machines such as Inline XBRL.

The meaning conveyed by human readable presentations of information and machine readable formats must be the same and professional accountants must be able to determine if, in fact, the human and machine readable formats convey the same meaning. Having multiple versions of the same information that have to be compared to make sure they are the same, generally, manually, is not efficient.

1.2.3.Separation of the Three Tasks of Constructing a Report

There are three separate tasks that are required to create a financial report and these three tasks should be distinguished and separated:

- **Gathering of facts**: The facts that are reported within a financial report come from three different sources: (a) accounting system databases, (b) spreadsheets and other semi-structured documents, (c) manually entered into the system.
- **Structuring of model**: The structure of the model of a financial report articulates both the structure of each fragment of a report and the flow or sequencing of the fragments within the report.
- **Presentation of the facts**: Presentation or rendering of facts can be achieved in two different ways: (1) automating to generation of presentations/renderings using facts, structure of the model, and metadata; (2) manually mapping a fact to its location in a rendering. For example,

[9] *A Theory of Systems for Educators and Managers*, http://xbrl.squarespace.com/journal/2015/7/24/deming-a-theory-of-a-system-for-educators-and-managers.html

creating an Inline XBRL XHTML representation of a report still requires the "gathering of facts" and the "structuring of model"; but then requires additional work of determining precisely where in an XHTML document a fact will be presented.

The point here is that these are three different, independent tasks. The first two tasks are always required. The third task, presentation of facts, can be achieved using automated generation processes but the downside of that approach is the precision of the rendering might not be precise. Manual presentation of facts within a report can be very precise using Inline XBRL; but the downside is that the precise placement of the facts in the report is a manual process that entails additional work. A combination of manual and automated approaches can be utilized to minimize work.

1.2.4.Separation of Metadata into Three Categories

Metadata is leveraged by this system extensively. There are three categories of metadata that are used by an expert system for creating general purpose financial reports. The difference between the three categories of metadata is important to understand because it helps software engineers understand who maintains the metadata.

- **System specific metadata**: System specific metadata relates to differences in technical implementations within systems that use XBRL-based reports. For example, the SEC and ESMA have different system specific metadata. System specific metadata is managed using the notion of profiles[10] and managed by technical professionals, business professionals never need to make choices about system specific metadata.
- **Business reporting specific metadata**: Business reporting specific metadata relates to all business reports. A general purpose financial report is a type of business report. For example the notion of a "roll up" and "roll forward" and "adjustment" and other such concept arrangement patterns are exactly the same for all reporting systems. Business professionals have no role in creating business reporting specific metadata. However, business professionals may specify the need for certain business reporting specific metadata. This metadata is managed by technical professionals with input from business professionals. An example of business reporting specific metadata is the allowed relationships between report elements such as [Hypercube]s, [Abstract]s, and [Concept]s.
- **Business report specific metadata**: Business report specific metadata relates to individual financial reports. Business professionals maintain and use business report specific metadata assisted by business reporting specific metadata and system specific metadata. For example, a business professional might create a concept and that concept is always created within some concept arrangement pattern specified by business report specific metadata; as such an application can assist a user in the process of creating a concept based on the known business reporting specific metadata available to the software application.

[10] *XBRL-based Digital Financial Reporting Profiles and General Business Reporting Profile*, http://xbrlsite.azurewebsites.net/2018/Library/Profiles-2018-01-24.pdf

Metadata plays a critical role in reducing complexity which is exposed to business professionals. The correct application of metadata exposes business professionals to the appropriate level of complexity, generally business domain specific complexity with which business professionals are familiar.

1.2.5. Construction of a Financial Report Involves Workflow

The construction of a general purpose financial report involves workflow. Different fragments of a report have different states of completion. Different users might work on different areas of a report and my not be allowed to edit other areas of a report. The goal of a report creation project is to get every aspect of every detail of creating a report correct and manage all of those details. For example, professional accountants have the notion of a "To do" which is a task which needs to be completed. A "To do" could be related to a report, a fragment of a report, one detail of the structure of the model of a report, or a detail related to fact contained within a financial report. A "To do" could also relate to a mapping between a fact and where that fact is rendered within a report.

As such, dashboards and other project management tools are essential in managing the creation of a general purpose financial report.

Many details related to the state or status of a general purpose financial report can be collected using automated processes. For example, a list of all the roll ups of a report that do not properly foot can be gathered using an automated process. However, there will always be subjective information such as "To do"s which are accumulated and managed using more manual processes.

1.2.6. Software's Intimate Understanding of a Business Report and Financial Report

There are two types of software tools: general purpose tools and task specific tools. There are tradeoffs between general purpose and task specific tools. General purpose tools can do more things, but the tools tend to be harder to use because of their general nature and universal use. Whereas task specific tools are more limited and can perform fewer tasks, but because the tasks are specific the tools are easier to use.

This expert system for creating general purpose financial reports is a task specific tool. It performs one task and it performs that task well. The software tool has an intimate understanding of the conceptual model of a business report and the conceptual model of a financial report[11]. It understands things like the pieces of the business report and financial report and how those pieces interact with one another.

1.3. Important Background Information

While all of the information in this section is covered in the section referenced in the previous section, this section strives to provide a succinct, abridged synthesis of the most foundational information that is important to understanding how to build an expert system for creating general purpose financial reports.

[11] Conceptual Model of a Digital Financial Report, http://xbrl.squarespace.com/conceptual-model/

1.3.1.Expert Systems

Expert systems[12] is a branch of artificial intelligence. Expert systems, also called knowledge based systems or simply knowledge systems, are computer programs. The following is one definition of an expert system:

Expert systems are computer programs that are built to mimic human behavior and knowledge. Expert systems are for reconstructing the expertise and reasoning capabilities of qualified experts within some limited, narrow domain of knowledge in machine-readable form. A model of the expertise of a domain of knowledge of the best practitioners or experts is formally represented in machine-readable form and the expert system reaches conclusions or takes actions based on that information when trying to solve some problem. The computer program performs tasks that would otherwise be performed by a human expert.

Expert systems are the most commercially successful applications of artificial intelligence research[13]. There are currently thousands of expert systems employed world-wide in industry and government. Expert systems are driven by metadata.

Frank Puppe explains in his book *Systematic Introduction to Expert Systems*[14] that there are three general categories of expert systems:

- **Classification or diagnosis type**: helps users of the system select from a set of given alternatives. The system tends to be instructional in nature.
- **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 or create predictions based on the system.

The assembly of a financial report can be assisted by a construction-type expert system. Helping professional accountants understand what goes into that financial report can be assisted by a classification-type expert system. Creating forecasts and projections of future financial reports can be assisted by simulation-type expert systems.

1.3.2.Artificial Intelligence

As was said, expert systems is a branch of artificial intelligence[15]; and artificial intelligence is a is a branch of computer science. A more neutral term than artificial intelligence is the term machine intelligence[16]; both terms have the same meaning. Think of these tools as narrowly focused employees with great memories that are very good at performing one specific repetitive task well, over, and over, and over. Literally, like a machine.

---

[12] *Understanding the Components of an Expert System*, http://xbrl.squarespace.com/journal/2016/5/24/understanding-the-components-of-an-expert-system.html

[13] Edward Feigenbaum et. al, *KNOWLEDGE-BASED SYSTEMS IN JAPAN*, http://www.wtec.org/loyola/kb/execsum.htm

[14] Frank Puppe, *Systematic Introduction to Expert Systems, Knowledge Representations and Problem-Solving Methods*, page 11 (Note that you can read Parts I and II on Google Books here, https://books.google.com/books?id=_kKqCAAAQBAJ)

[15] *Introduction to Artificial Intelligence Terminology*, http://xbrl.squarespace.com/journal/2016/7/21/introduction-to-artificial-intelligence-terminology.html

[16] Shivon Zilis and James Cham, Harvard Business Review, *The Competitive Landscape for Machine Intelligence*, https://hbr.org/2016/11/the-competitive-landscape-for-machine-intelligence

Artificial intelligence programs that provide expert-level proficiency in solving problems by bringing to bear a body of knowledge about specific tasks are called knowledge based systems or expert systems. Artificial intelligence programs work using metadata.

There are two types of artificial intelligence, specialized and generalized. The type of artificial intelligence employed for this application is specialized artificial intelligence. There are two types of reasoning systems used by artificial intelligence, deductive and inductive. The type of artificial intelligence used to create an expert system for financial reporting will be deductive reasoning but could rapidly expand into inductive reasoning once enough metadata has been accumulated.

1.3.3.Creating an Expert System or Knowledge Based System

Computers are dumb beasts. Creating a knowledge based system[17] involves the transformation of machine-readable instructions in such a way as to explain to a machine how a system works and how to make a system work the way you want that system to work.

Then, brick-by-brick, much like building a house, business domain experts and software engineers can create tools that automate certain types of tasks in that process. Humans encode information, represent knowledge, and share meaning using machine-readable patterns, languages, and logic.

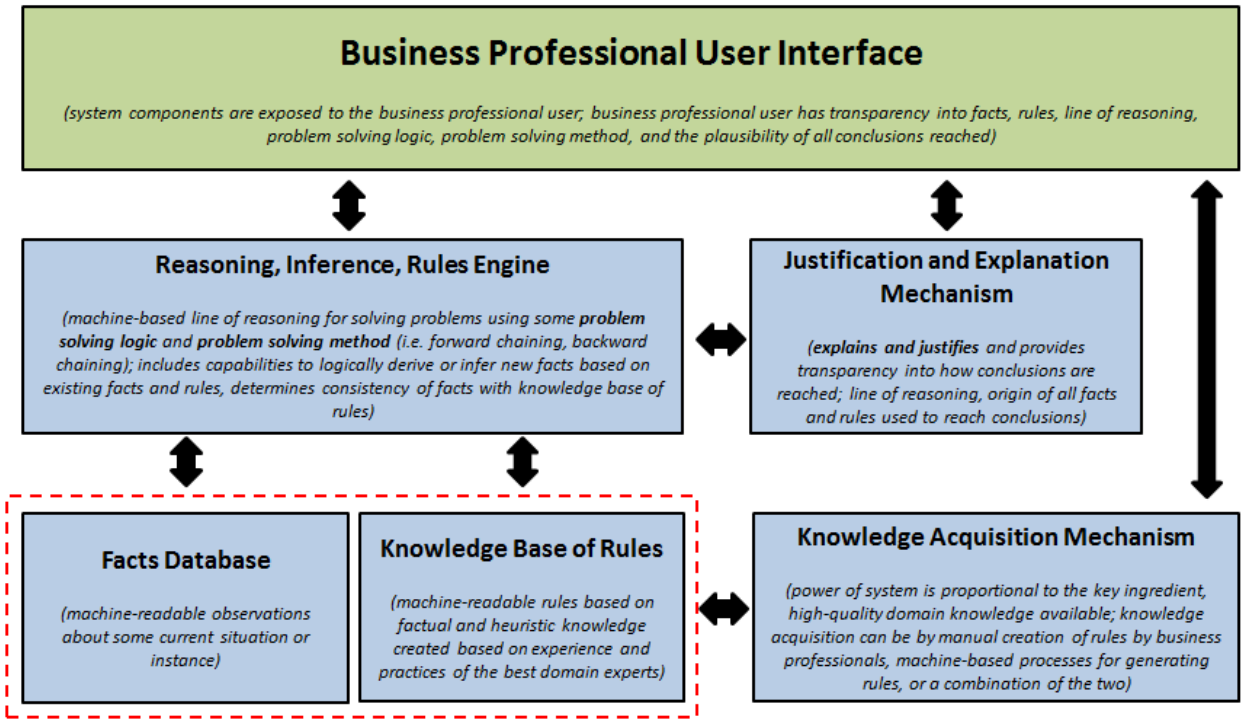
This is how you get the dumb beasts to perform useful work.

1.3.4.Components of a Knowledge Based System (Expert System)

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

Nothing is a "black box", rather every aspect of every decision made by a system is completely understandable to the business professional using the system. Below is a graphic of the components of a knowledge based system:

[17] Another term used to describe such systems is reasoning system.



1.3.5.Unique Handles for Identification

A handle can be thought of as a reference to something. In order to grab something, a computer needs to understand exactly what to grab. Suppose you were using some name to identify what to grab, if that name is not unique in the set of things you are searching to find what you want, a computer software application will be incapable of grabbing the right thing.

There are two important terms which should be understood, isomorphic and polymorphic[18].

- **Isomorphic** comes from the Greek term "iso" meaning "one". As used here isomorphic means that something has exactly one meaning.
- **Polymorphic** comes from the Greek term "poly" meaning more than one, many. As used here polymorphic means that something has more than one meaning.

The point here is that for a computer software application to accurately grab a piece of information that piece of information must be uniquely identifiable. The easier it is to obtain that unique identifier, or handle, the easier it is to grab the piece of information.

1.3.6.Business Rules

The Merriam-Webster dictionary defines anarchy[19] as "a situation of confusion and wild behavior in which the people in a country, group, organization, etc., are not

[18] Hypercubes/Dimensions Analysis: Summary, http://www.xbrlsite.com/Examples/Dimensions/Index.html
[19] Anarchy definition, Merriam-Webster, http://www.merriam-webster.com/dictionary/anarchy

controlled by rules or laws." Business rules prevent information anarchy[20]. Business rules are metadata.

Business rules guide, control, suggest, or influence behavior. Business rules cause things to happen, prevent things from happening, or suggest that it might be a good idea if something did or did not happen. Business rules help shape judgment, help make decisions, help evaluate, help shape behavior, and help reach conclusions.

Business rules arise from the best practices of knowledgeable business professionals. A business rule is a rule that describes, defines, guides, controls, suggests, influences or otherwise constrains some aspect of knowledge or structure within some problem domain.

Don't make the mistake of thinking that business rules are completely inflexible and that you cannot break rules. Sure, maybe there are some rules that can never be broken. Maybe there are some rules that you can break. It helps to think of breaking rules as penalties in a football game. The point is that the guidance, control, suggestions, and influence offered by business rules are a choice of business professionals. The meaning of a business rule is separate from the level of enforcement someone might apply to the rule.

A business rule states a fact about the world (declarative rule). A business rule can provide instructions (production rule).

1.3.7.Business Rules Engine

A business rules engine is a style of writing software. Anything achievable by a business rules engine is achievable by general functional programming. But, business rules engines can have an advantage in certain cases. For example, when you want to separate business logic from program logic and allow business professionals to control and maintain business rules.

A business rules engine, or rules engine, is a mechanism for executing business rules. Rules engines are optimized to figure out what rules apply to what facts and the order rules should be run and result is some consequence based on the facts and rules. A rules engine is a non-trivial application to create. The stronger the problem solving logic of the rules engine supports, the more powerful the rules engine.

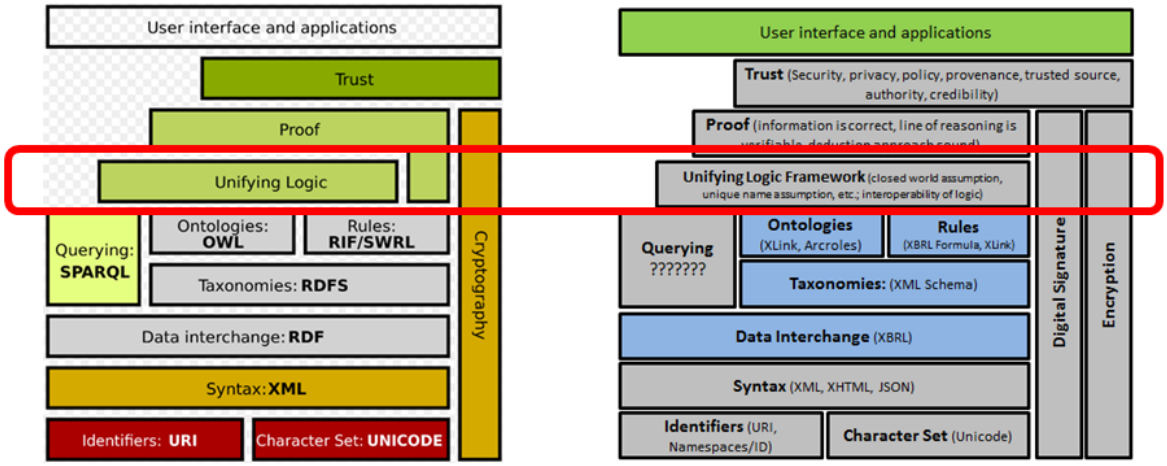
1.3.8.Multiple Technology Stacks

Arbitrary preferences, fads, trends, misinformation, and many other things influence how information technology professionals solve the same problem. As such, there will always be multiple technology stacks as opposed to every information technology professional using exactly the same approach to solving what amounts to be exactly the same problem.

Consider the comparison of these two technology architecture stacks[21]: Semantic Web Stack and XBRL Stack:

[20] *Understanding that Business Rules Prevent Anarchy*, http://xbrl.squarespace.com/journal/2016/7/15/understanding-that-business-rules-prevent-anarchy.html

[21] Unifying Logic Framework for Business, http://xbrl.squarespace.com/journal/2017/11/26/unifying-logic-framework-for-business.html



Each of the technical architecture stacks has a problem solving logic. The Semantic Web Stack on the left calls it a "Unifying Logic" and the XBRL Stack calls it a "Unifying Logic Framework".

The problem solving logic for the Semantic Web Stack is defined by the technical syntax offered by RDFS, OWL, RIF/SWIRL/SHACL. For the Semantic Web Stack, those technical syntaxes provide the boundaries for the problem solving logic. For the XBRL Stack, there really are no real boundaries outlined at all for the problem solving logic. Even if someone could provide the specifics of the problem solving power offered by RDFS, OWL, and RIF/SWIRL/SHACL; that list undoubtedly would not be understandable to professional accountants or other business professionals. But if business professionals are supposed to control and maintain business rules, then those business logic boundaries would best be clear and understandable. The bottom line here is that the boundaries of a problem solving logic should be understandable.

There are two complete systems that could very likely be used to create a complete financial report creation system using the semantic web stack: TopBraid[22] and ErgoAI[23]. These systems might be worth looking at to help you understand the features that such a system must likely have. It is doubtful that any business professional would ever be able to use solutions such as TopBraid or ErgoAI. However, if additional layers we built on top of those systems to reduce complexity, business professionals might likely be able to leverage general tools such as TopBraid or ErgoAI.

1.3.9.Reasoning Systems

There are two primary types of reasoning systems[24]: deduction[25] and induction[26]. These two videos help you understand the difference between deduction and induction: *What is Deduction and Induction*[27]; *What is Inductive Logic*[28].

---

[22] TopBraid, https://www.topquadrant.com/products/topbraid-enterprise-data-governance/
[23] ErgoAI, http://coherentknowledge.com/ergo-documentation/

[24] Wikipedia, *Reasoning System*, https://en.wikipedia.org/wiki/Reasoning_system
[25] Wikipedia, *Deductive Reasoning*, https://en.wikipedia.org/wiki/Deductive_reasoning
[26] Wikipedia, *Inductive Reasoning*, https://en.wikipedia.org/wiki/Inductive_reasoning
[27] YouTube.com, *What are Deduction & Induction?*, https://www.youtube.com/watch?v=iRcNQkWNWNk

The bottom line is that *deduction* is certain; you can be logically certain of the conclusions reached using deduction. That is the goal of deduction, certainty.

Induction is not based on certainty, induction is based on probability.

The black swan problem of induction[29] clearly explains the problem of induction. "No amount of observations of white swans can allow the inference that all swans are white, but the observation of a single black swan is sufficient to refute that conclusion."

Both deductive and inductive reasoning systems have appropriate uses. The Pesseract software application used for creating financial reports uses mainly deductive reasoning.

1.3.10. Problem Solving Method

There are basically two primary problem solving methods: forward chaining and backward chaining. Each method has pros and cons. Forward chaining can do everything that backward chaining can do; but backward chaining cannot do everything that forward chaining can do. It is best to use both approaches, applying each when circumstances dictate[30]. Sequential processing using series of "IF…THEN statements" could do everything that a forward chaining reasoning engine could do; however, maintenance and flexibility issues would likely result.

1.3.11. Problem Solving Logic Expressiveness and Safety

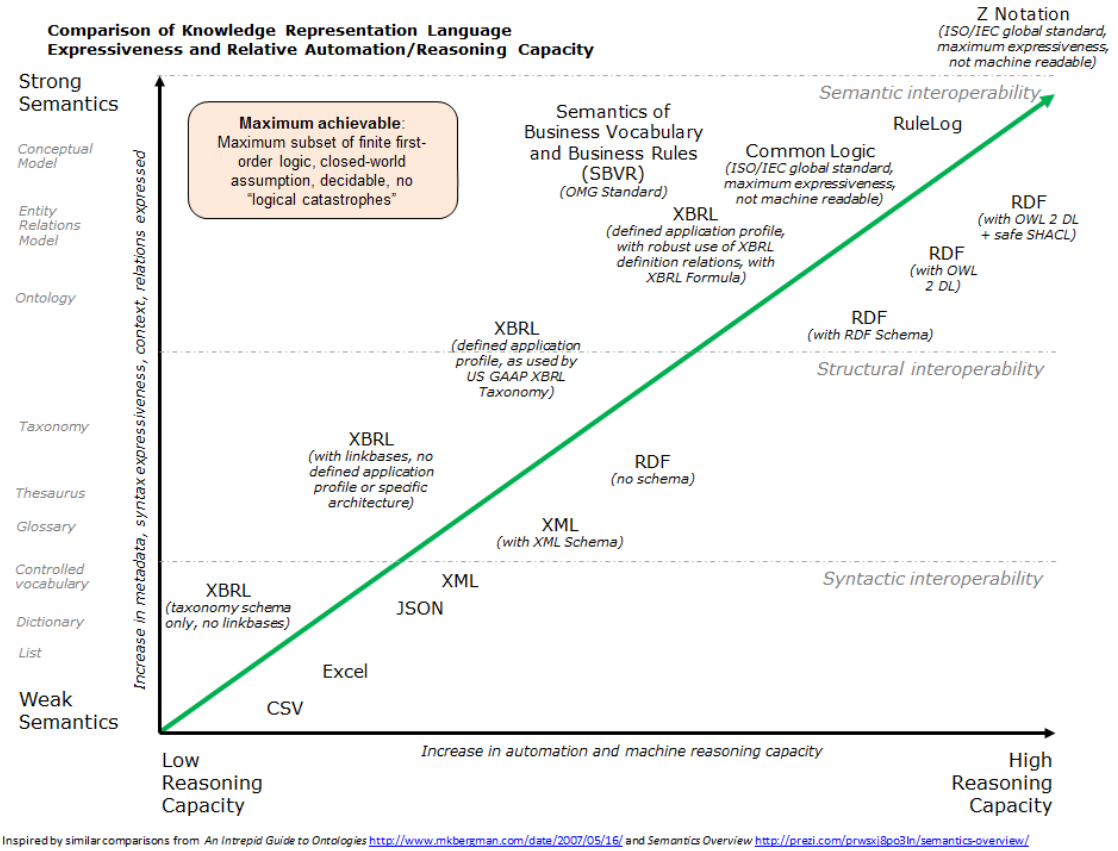
Problem solving logic can best be described by two things: (a) the expressive power of the logic and (b) the safety of the logic to avoid catastrophic failure. The power of a problem solving logic is directly related by an ability to express rules. Different technical syntax support different problem solving logic. The following graphic compares and contrasts the relative expressive power of different syntax that can be used to express rules.

---

[28] YouTube.com, *What is Inductive Logic?,* https://www.youtube.com/watch?v=4ZKa1S1wPy4

[29] Capture, Black Swans and the Problem of Induction, https://capturehighered.com/predictive-modeling/black-swans-problem-induction/

[30] *Differentiating Forward and Backward Chaining*, http://xbrl.squarespace.com/journal/2016/5/27/differentiating-forward-and-backward-chaining.html

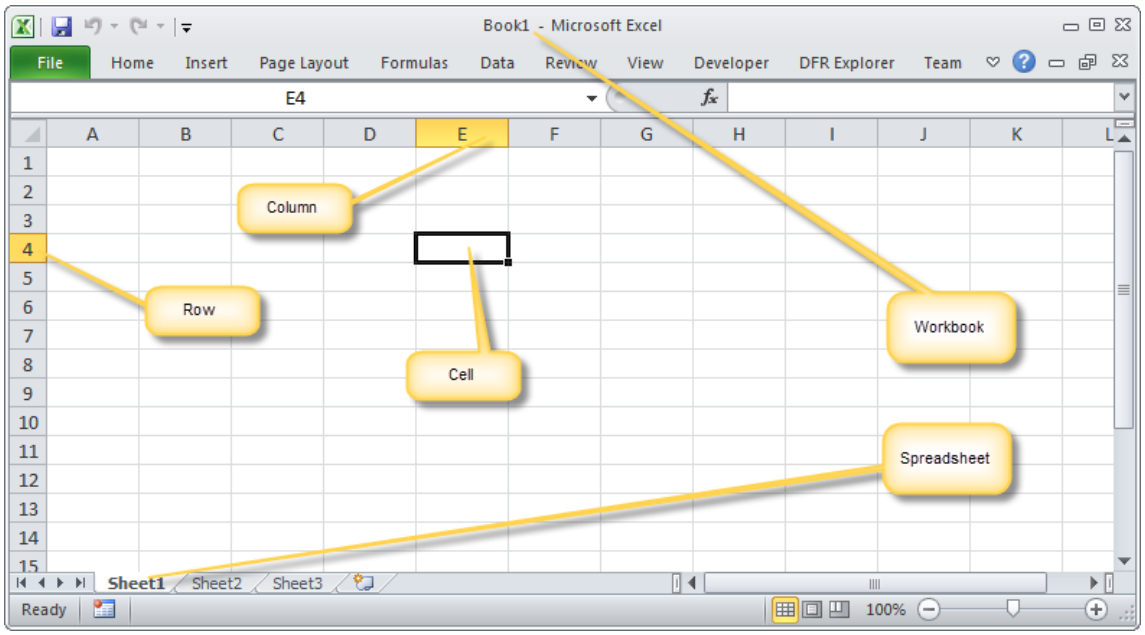


The lowest denominator in implementing logic in software is what is called an NAND gate[31] . All logic systems can be converted into NAND gates. Theoretically, any logic function can be realized by correctly combining together enough NAND gates. What certain logical systems do is provide high-level functions that make doing complex tasks that might involve hundreds if not thousands of set of NAND gates easier to achieve.

1.3.12. Using Conceptual Models

Business professionals work with conceptual models every day. For example, the workbooks, spreadsheets, rows, columns, and cells of an electronic spreadsheet are a conceptual model. The ease and simplicity of an electronic spreadsheet allows the average business professional to make use of this helpful tool.

[31] Wikipedia, *NAND Gate, retrieved June 7, 2017*, https://en.wikipedia.org/wiki/NAND_logic



1.3.13. Multidimensional Model

Models help communication and provide a framework for understanding. The multidimensional model is a model for understanding information[32]. Every professional accountant works with multidimensional information every day but don't generally realize it.

Just like an electronic spreadsheet has a model (workbook, spreadsheet, row, column, cell); a digital financial report has a model. The model of a digital financial report follows the multidimensional model. Here are the high-level pieces of a digital 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. Facts can be numbers, text, or prose.
- **Characteristic**: A characteristic describes a fact (a characteristic is a property of a fact). A characteristic provides information necessary to describe a fact and distinguish one fact from another fact. A fact may have one or many distinguishing characteristics.
- **Fact table**: A fact table is a set of facts which go together for some specific reason. All the facts in a fact table share the same characteristics.
- **Relation**: A relation is how one thing in a business report is or can be related to some other thing in a business report. These relations are often called business rules. There are three primary types of relations (others can exist):
    - **Whole-part**: something composed exactly of their parts and nothing else; the sum of the parts is equal to the whole (roll up).

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

  - **Is-a**: descriptive and differentiates one type or class of thing from some different type or class of thing; but the things do not add up to a whole.
  - **Computational business rule**: Other types of computational business rules can exist such as "Beginning balance + changes = Ending Balance" (roll forward) or "Net income (loss) / Weighted average shares = Earnings per share".
- **Grain**: Grain is the level of depth of information or granularity. The lowest level of granularity is the actual transaction, event, circumstance, or other phenomenon represented in a financial report. The highest level might be a line item on a primary financial statement such as a balance sheet.

1.4. Functioning of the Expert System

This section explains certain specific key pieces of the expert system, why they exist, and provides examples of what each piece does. While the pieces are explained separately many pieces work together in concert to orchestrate the creation of a general purpose financial report. I have tried to explain these pieces in the most optimal order.

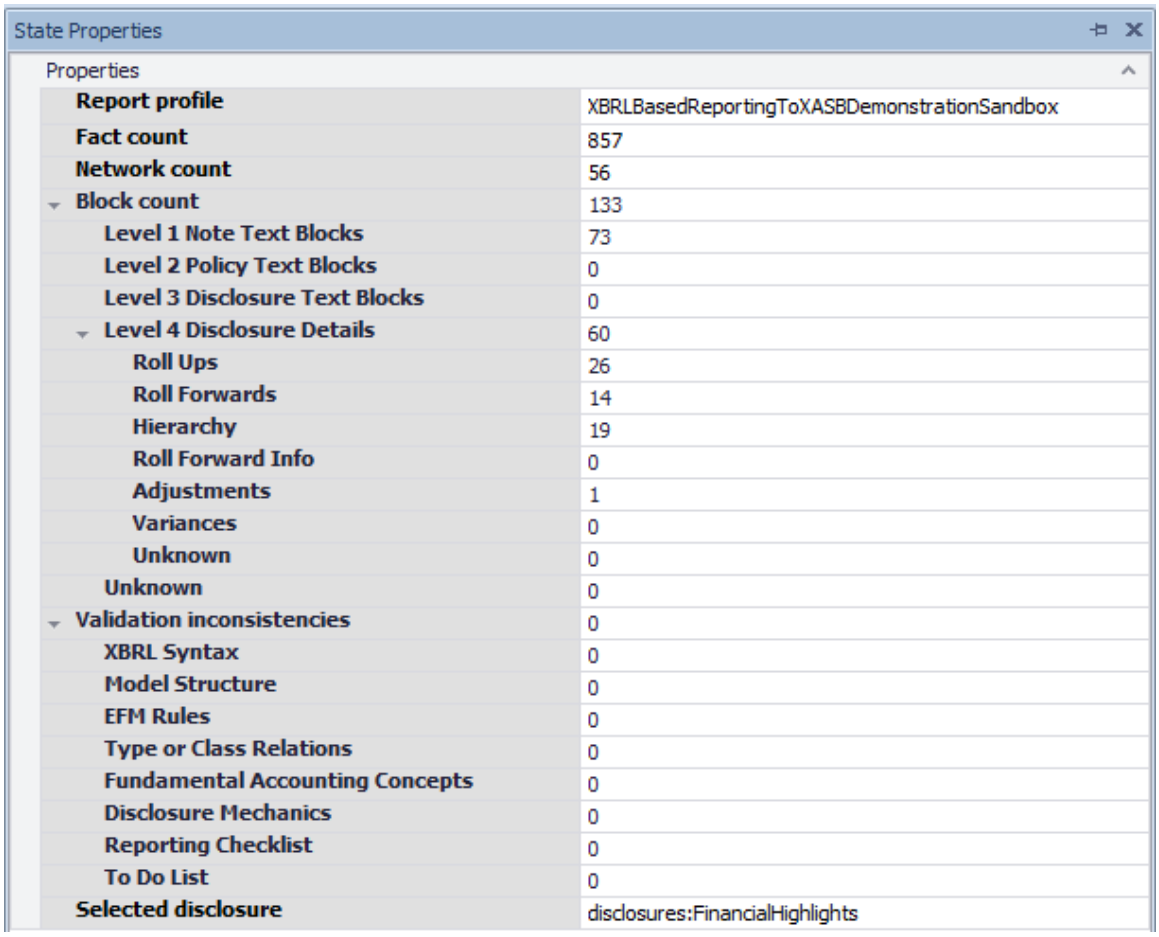
Essentially, this section explains the mechanisms created (i.e. state machine, agenda, business rules engine, dashboards, explanation mechanism), techniques (blocks, slots, profiles), and metadata (disclosures, topics, exemplars, prototypes) that interact to provide the desired user experience to the professional accountant using the software tool.

1.4.1. State Machine

A state machine[33] is a formal process of managing state or some combination of variables that currently exists within the application. The diagram below shows an example of the state variables of the Pesseract application[34]. Note that the exact number of facts is known as is the exact number of validation inconsistencies, the disclosure which is current selected by the user, and other such information.

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

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



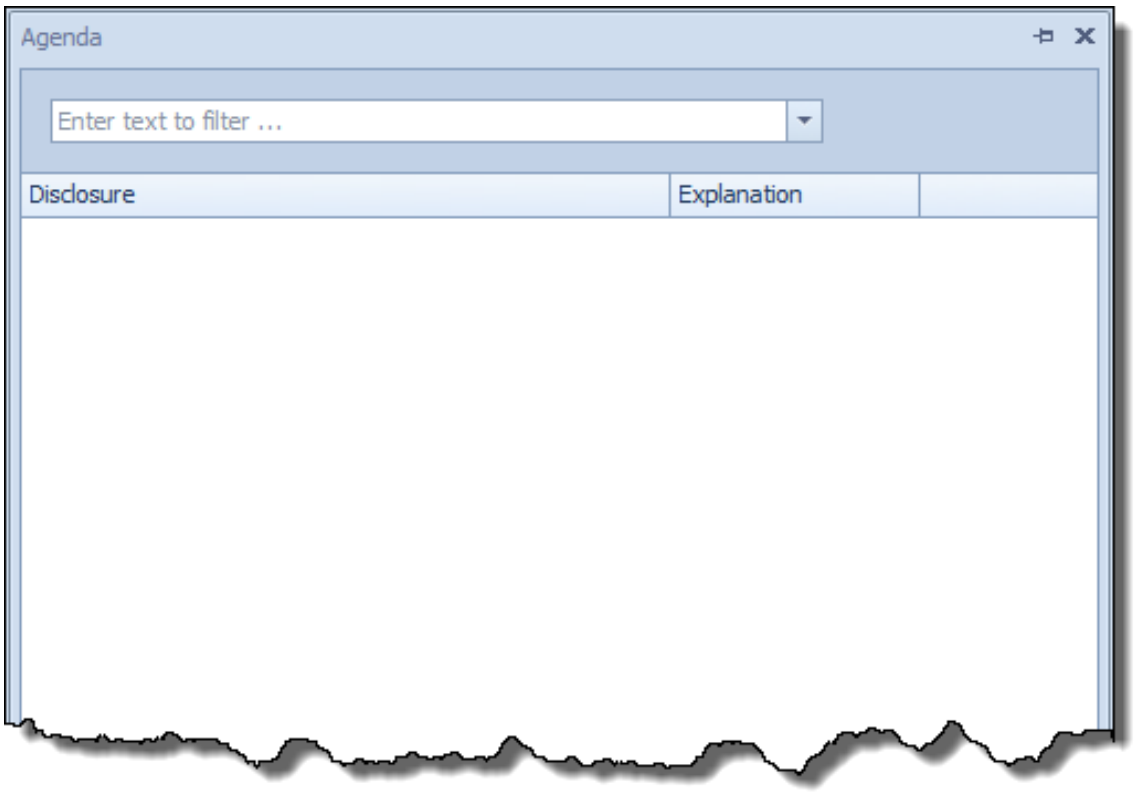
The idea of using a state machine came from video games such as SIM City. While basically every software application has to manage state in some manner, if there are a lot of state variables it is best to create a formal process for managing state.

1.4.2.Agenda

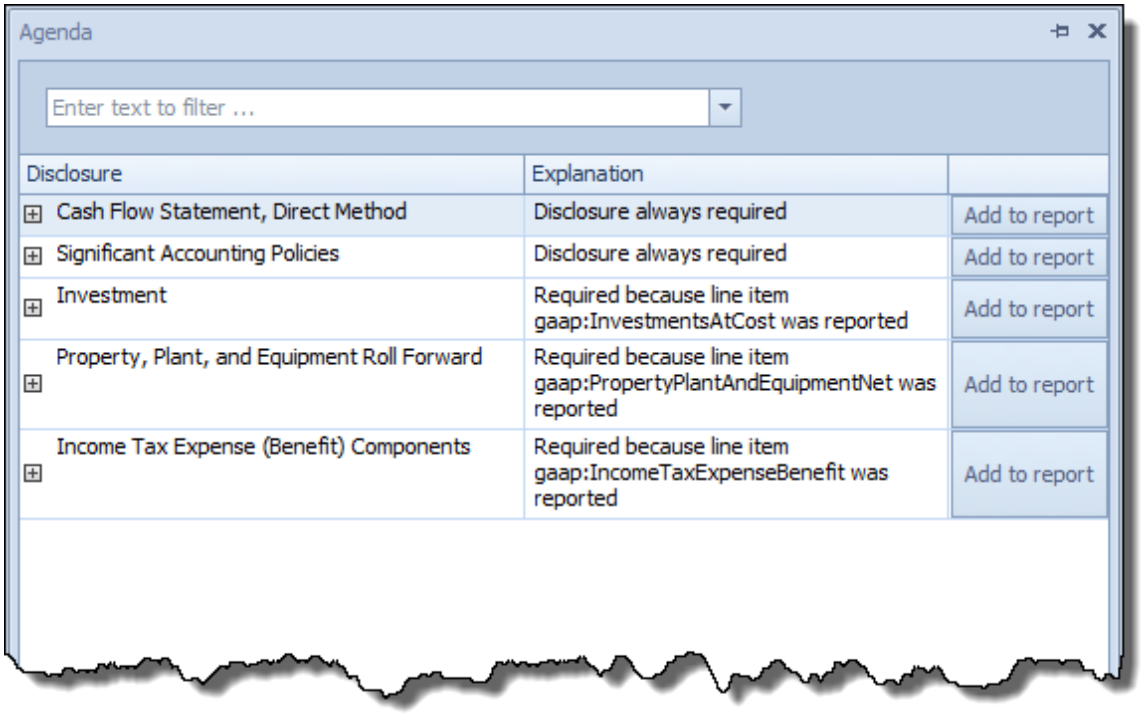
An agenda is simply a list of tasks which need to be completed by the system. Both manual tasks, such as "To dos", and automated tasks might be listed in the agenda. The idea of an agenda came from the expert system Clips[35].

Below are two screen shots of the agenda of Pesseract. The first screen shot is rather uninteresting, the agenda is empty. The reason the agenda is empty is because all the tasks necessary to complete the general purpose financial report you are working on have been completed.

[35] *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



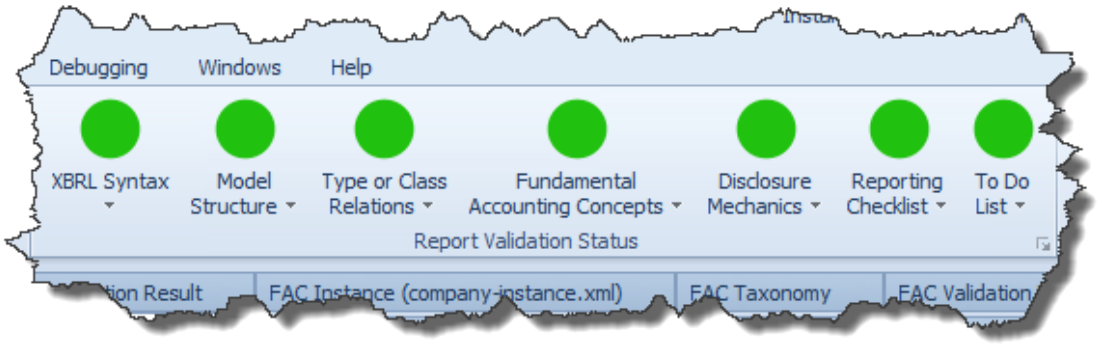
Contrasting this to the second agenda shown below, there are a number of tasks that remain to be completed in order for the financial report to be considered complete.



The agenda is driven by machine-readable business rules. For example, one set of business rules that drives the agenda is the reporting checklist[36].

1.4.3. Validation Dashboard

A dashboard is a visually easy way of absorbing information about a report. A validation dashboard is a dashboard that summarizes the validation results for a report. For example, below the Pesseract *Report Validation Status* dashboard is shown. From the dashboard the state of the report can be quickly and easily understood.



There is a connection between the agenda and the validation dashboard. Both the agenda and the validation dashboard are driven by some of the same metadata.

To better understand dashboards it is recommended that you read *Blueprint for Creating Zero-defect XBRL-based Digital Financial Reports*[37].

1.4.4. Explanation Mechanisms

Pesseract goes to great lengths to provide complete transparency into the operations of the reasoning systems employed. Easy to understand interfaces are provided to business rules, the line of reasoning is explained, and dashboards are provided that provided details into all validation results.

Below you see the details of the disclosure mechanics validation, the business rules used for testing one disclosure, and the line of reasoning used to reach the conclusion for one line item on the detail report:

Details:

[36] Example reporting checklist rules, http://xbrlsite.azurewebsites.net/2016/conceptual-model/reporting-scheme/xasb/reporting-checklist/ReportingChecklist-xasb-rules-def.xml

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

#	Disclosure	Category	Level	Pattern	Disclosure Found	Disclosure Consistent	Representation Concept [TEXT BLOCK]
1	Assets [Roll Up]	Unknown	Level4Detail	RollUp	True	CONSISTENT	NOT-EXPECTED
2	Balance Sheet	Statement	Level4Detail	Component	True	CONSISTENT	-
3	Basis of Reporting	Unknown	Level1TextBlock	TextBlock	True	CONSISTENT	Overall Financial Report Presentation an...
4	Buildings [Roll Forward]	Unknown	Level3TextBlock/Level4Detail	RollForward	True	CONSISTENT	Property, Plant, and Equipment Roll For...
12	Cash and Cash Equivalents Components	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Cash and Cash Equivalents Component...
13	Cash Flow Statement, Direct Method	Unknown	Level4Detail	RollUp	True	CONSISTENT	NOT-EXPECTED
14	Common Stock, By Class	Unknown	Level3TextBlock/Level4Detail	Hierarchy	True	CONSISTENT	Common Stock by Class [Schedule]
15	Director Compensation	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Directors Compensation [Schedule]
16	Director Compensation, Options Granted	Unknown	Level3TextBlock/Level4Detail	Hierarchy	True	CONSISTENT	Directors Compensation Options Grante...
17	Document Information	Unknown	Level4Detail	Hierarchy	True	CONSISTENT	NOT-EXPECTED
18	Earnings Per Share Summary	Unknown	Level4Detail	Hierarchy	True	CONSISTENT	NOT-EXPECTED
19	Entity Address	Unknown	Level4Detail	Hierarchy	True	CONSISTENT	NOT-EXPECTED
20	Entity Information	Unknown	Level4Detail	Hierarchy	True	CONSISTENT	NOT-EXPECTED
21	Financial Highlights	Unknown	Level3TextBlock/Level4Detail	Hierarchy	True	CONSISTENT	Financial Highlights [HTML]
22	Furniture and Fixtures [Roll Forward]	Unknown	Level3TextBlock/Level4Detail	RollForward	True	CONSISTENT	Property, Plant, and Equipment Roll For...
23	Income Statement	Unknown	Level4Detail	RollUp	True	CONSISTENT	NOT-EXPECTED
24	Income Tax Expense (Benefit) Components	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Income Tax Expense (Benefit) Compone...
25	Inventory Components	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Inventory Components [Schedule]
26	Investment	Unknown	Level3TextBlock/Level4Detail	Hierarchy	True	CONSISTENT	Investments [Schedule]
27	Land [Roll Forward]	Unknown	Level3TextBlock/Level4Detail	RollForward	True	CONSISTENT	Property, Plant, and Equipment Roll For...
28	Leasehold, Land, and Building	Unknown	Level3TextBlock/Level4Detail	Hierarchy	True	CONSISTENT	Leasehold Land and Buildings [Schedule]
29	Liabilities and Equity [Roll Up]	Unknown	Level4Detail	RollUp	True	CONSISTENT	NOT-EXPECTED
30	Long-Term Debt Components	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Long-Term Debt Components [Schedule]
31	Long-Term Debt Current and Noncurrent Por...	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Long-Term Debt Current and Noncurrent...
32	Long-Term Debt Instruments	Unknown	Level3TextBlock/Level4Detail	Hierarchy	True	CONSISTENT	Long-Term Debt Instruments [Schedule]
33	Long-Term Debt Maturities	Unknown	Level3TextBlock/Level4Detail	RollUp	True	CONSISTENT	Long-Term Debt Maturities [Schedule]

Business rules for one disclosure:

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

Line of reasoning for one line of the detailed report:

Rules	Line of Reasoning
####Disclosure mechanics validation explanation for disclosure: disclosures:InventoryComponents####	
Level 4 Disclosure Detail	
Looking for blocks with concept arrangement pattern: RollUp	
Looking for Hypercube/[Table]: gaap:InventoryByCompomentTable	
*FOUND Hypercube/[Table]: gaap:InventoryByCompomentTable in network:	
Looking for Concept: gaap:Inventory	
*FOUND Concept: gaap:Inventory in network:	
Looking for Concept: gaap:Inventory	
*FOUND Concept: gaap:Inventory in network:	
Concept located in network: JG: Schedule: Inventory, by Component	
Level 1 Note Text Block	
Looking for Level 1 note text block: gaap:InventoryHTML	
*FOUND Level 1 note text block: gaap:InventoryHTML in network:	
Text block located in network: XE: Note: Inventory Note	
Level 3 Disclosure Text Block	
Looking for Level 3 Disclosure Text Block: gaap:InventoryComponentsSchedule	
*FOUND Level 3 Disclosure Text Block: gaap:InventoryComponentsSchedule in network:	
Text block located in network: XE: Note: Inventory Note	
Level 2 Policy Text Block	
Looking for Level 2 policy text block: gaap:InventoryPolicyHTML	
*FOUND Level 2 policy text block: gaap:InventoryPolicyHTML in network:	
Text block located in network: XB: Note: Significant Accounting Policies Note	
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 ####	

The above examples are from the working proof-of-concept Pesseract. Also available is the complete validation details, rules, and lines of reasoning implemented within a commercial software application which you can test drive for yourself[38]. The primary thing to note is the extremely readable explanation mechanisms that help professional accountants understand the conclusions which were reached by the expert system.

1.4.5.Blocks

The notion of a Block[39] is one of the harder ideas to explain and understand. Blocks are explained in detail in the document, *Putting the Expertise into an XBRL-based Knowledge Based System for Creating Financial Reports* [40].

A Block is a unit of a report that was created in order to make interacting with and otherwise working with a report easier. Individual facts are many times too small working sets to be useful. Networks tend to be too large to work with. A Block is a useful unit of a report that makes doing certain things significantly easier.

---

[38] XBRL Cloud Implementation of the reporting checklist and disclosure mechanics, http://xbrlsite.azurewebsites.net/2017/Prototypes/XASB/Disclosure%20Mechanics%20and%20Reporting%20Checklist.html

[39] YouTube.com, *Understanding Blocks*, https://www.youtube.com/watch?v=yI9yjD_T78I

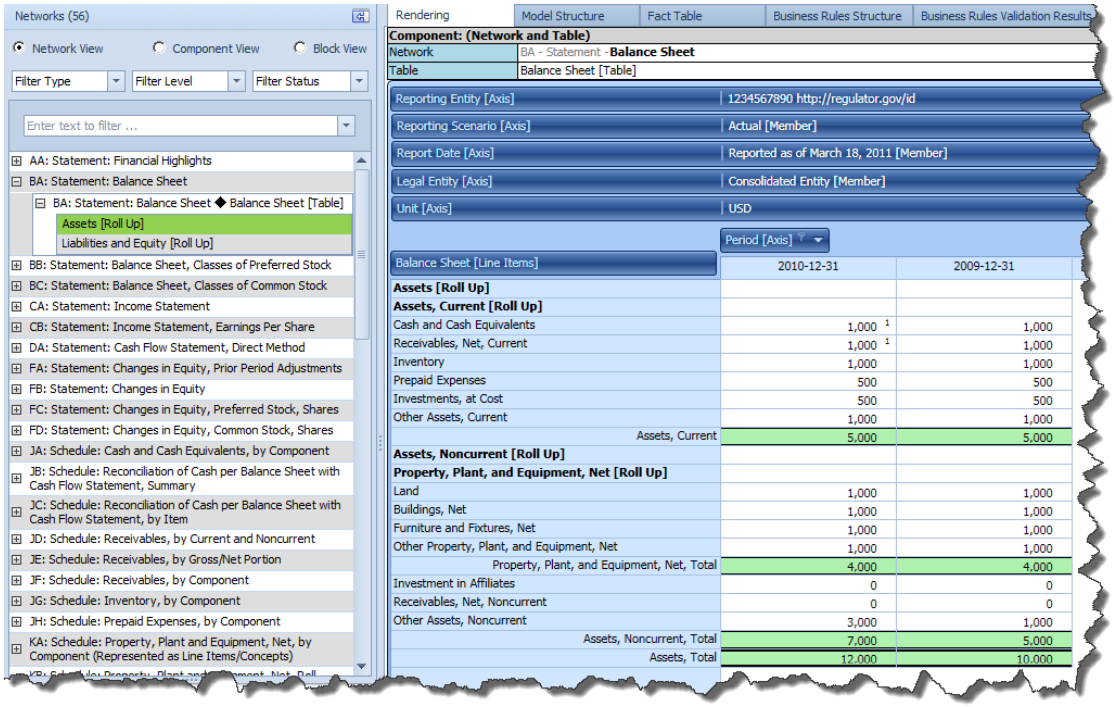
[40] *Putting the Expertise into an XBRL-based Knowledge Based System for Creating Financial Reports*, http://pesseract.azurewebsites.net/PuttingTheExpertiseIntoKnowledgeBasedSystem.pdf

Essentially, a Block is a fragment of a report that has the same concept arrangement pattern. For example, there is a simple basic block:

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

To understand Blocks, you need to understand the pros and cons of working with Networks and Components. Networks tend to be too large and inconsistent to work with. Components, which is a Network plus a Hypercube is a more useful working unit, but it still has drawbacks. For example, a Hypercube could contain a roll up and a roll forward and makes rendering both correctly challenging.

Blocks are very consistent structures and make many, many things possible. Below is a screenshot of the Block "Assets [Roll Up]" which is part of the Component "Balance Sheet", which is part of the Network "Statement – Balance Sheet":



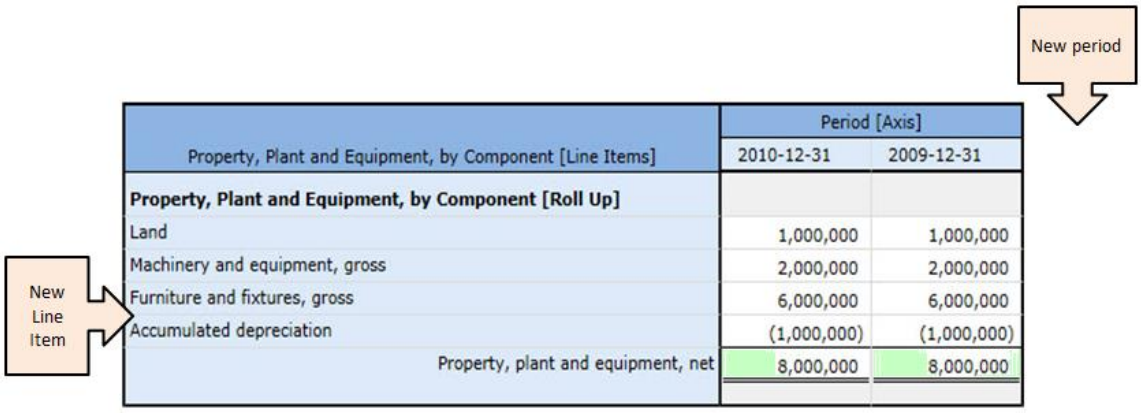
Note that then notion of blocks was inspired by MIT's Scratch application[41] and Blockly[42].

[41] MIT, Scratch, https://scratch.mit.edu/

[42] Blockly, http://xbrl.squarespace.com/journal/2014/7/14/blockly.html

1.4.6.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. For example, a roll up and a roll forward have different slots. Below you can see one Block, which is a roll up, showing two Slots for that block. One Slot is that a new Line Item can be added within the roll up total, essentially adding a new item to the roll up. Or, a second Slot is that a new period can be added to the Block.



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

1.4.7.Disclosures

A disclosure is simply something that is disclosed within a financial report[43]. A disclosure is a set of facts that go together for some specific purpose. For example, "Maturities of long-term debt" is a disclosure. As we are using the term, a "balance sheet" is also a disclosure. The primary point we are trying to make here is that we are giving specific unique handles to every disclosure which might be included within a financial report[44]. Today, these disclosures do not have formal names. The benefit of this is that computer software can work with disclosures.

1.4.8.Topics

A topic is simply a heading or category into which a disclosure might be placed. A topic is an approach to organizing disclosures. Rather than working with a list of hundreds or thousands of disclosure, that list can be broken into helpful groups that make certain tasks easier for professional accountants. Topics are presentation related. Different professional accountants might have different preferences as to how a disclosure might be organized into a presentation.

1.4.9.Profiles

Each implementation or system uses different pieces of the XBRL technical syntax. No implementation or system ever uses all pieces of XBRL technical syntax. Profiles[45] are used to overcome these implementation details that can be different

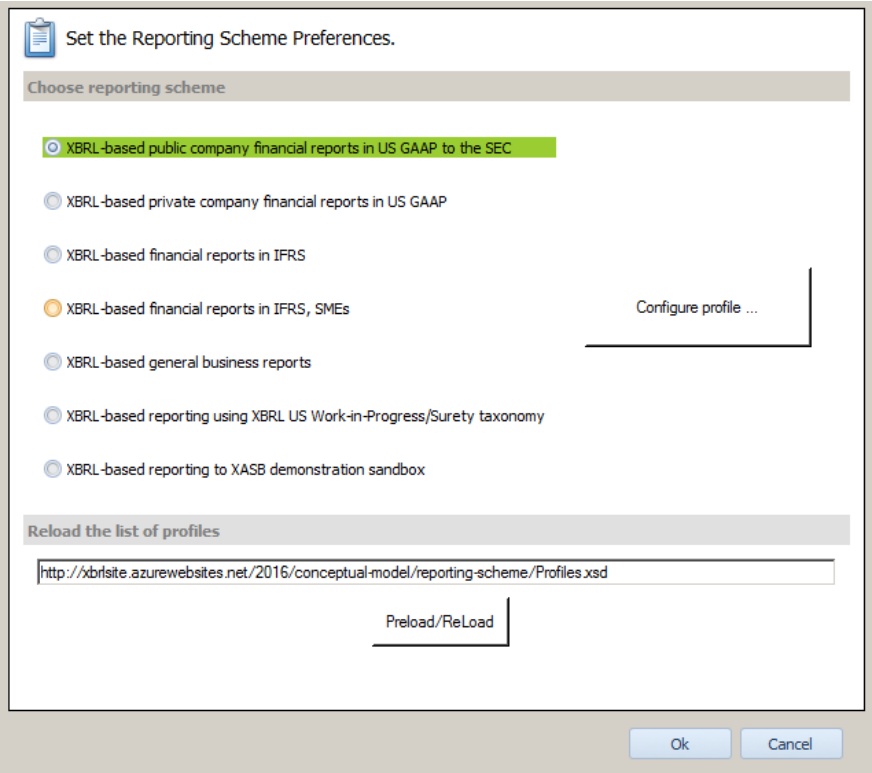
[43] Note that the term "disclosure" and "presented on the face of the financial statements" are used differently by accountants. Here, the term disclosure means all information reported, whether reported in a primary financial statement or within the disclosure notes of a financial report.

[44] US GAAP Disclosures, http://www.xbrlsite.com/2015/fro/us-gaap/html/Disclosures/Detail/index.html

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

within different implementations. A profile, or application profile, is simply an implementation style[46].

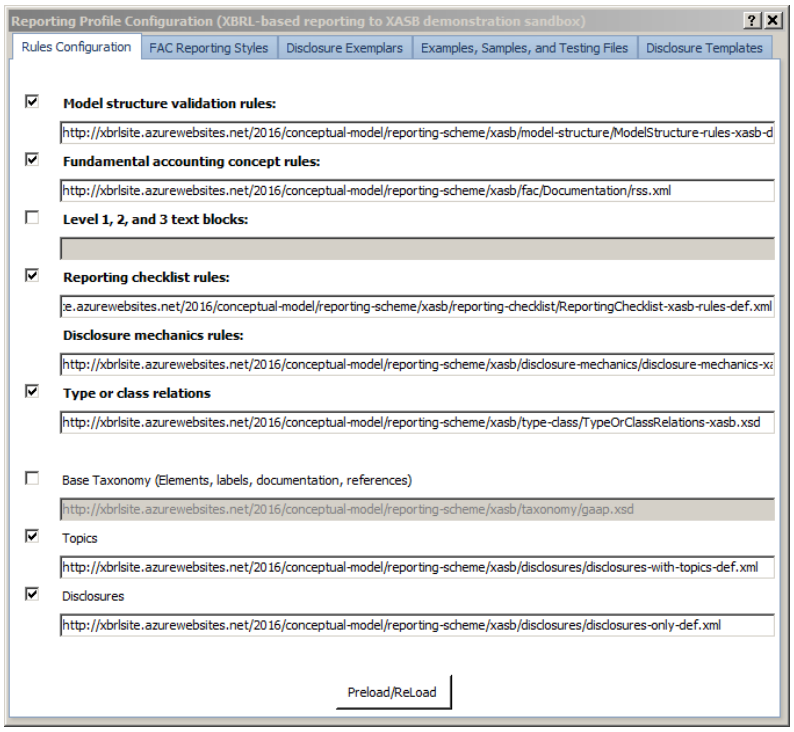
There is one specific profile worth mentioning. Defining an application profile for XBRL takes a tremendous amount of knowledge. Rather than each implementation defining its own profile, Pesseract also implements a special profile called the *General Business Reporting XBRL Application Profile*[47]. This profile is a well-thought-out, well tested, best practice implementation of XBRL. The application profile takes the very best ideas of XBRL implementations and avoids bad ideas creating a proven XBRL application profile that can serve the needs of most business professionals have.



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.

[46] XBRL-based Digital Financial Reporting Profiles and General Business Reporting Profile, http://xbrlsite.azurewebsites.net/2018/Library/Profiles-2018-01-24.pdf

[47] General Business Report Profile, http://xbrlsite.azurewebsites.net/2017/Library/GeneralBusinessReportingProfile-2017-12-20.pdf



The software application user is not bothered with configuration details; however, should new profiles be desired or changes made to preferences in metadata used, the application is flexible enough to allow adjustments. No metadata is hard coded.

1.4.10. Templates

A template[48] is an organized set of information would generally make up some fragment of a report. A template can be imported into the application rather than keying in information. Templates can be organized into a library. Pesseract has a machine readable library of templates for US GAAP[49], IFRS[50] and the XASB[51] prototype reporting scheme.

1.4.11. Exemplars

An exemplar is simply an example. Suppose you wanted to create a balance sheet. A good way to do that is to look at other balance sheet examples that help you understand what your balance sheet might look like[52].

The financial statements of public companies that are submitted to the Securities and Exchange Commission are created by some of the best professional accountants in

[48] Reporting Template Visual Index, http://www.xbrlsite.com/2013/ReportingTemplates/2013-05-15/TemplateIndex/Visualization.html

[49] US GAAP template library, http://xbrlsite.azurewebsites.net/DigitalFinancialReporting/Templates/us-gaap/2017-05-07/rdf.xml

[50] IFRS template library, http://xbrlsite.azurewebsites.net/DigitalFinancialReporting/Templates/ifrs/2017-05-07/rdf.xml

[51] XASB template library, http://xbrlsite.azurewebsites.net/2016/conceptual-model/reporting-scheme/xasb/templates/rdf.xml

[52] Disclosure Best Practices, http://xbrlsite-app.azurewebsites.net/DisclosureBestPractices/DisclosureBestPractices.aspx?DisclosureName=BalanceSheet

the world. The disclosures of these public companies serve as excellent machine-readable examples of literally any disclosure you might desire to create.

1.4.12. Prototypes

A prototype is similar to an example, more like a template. A prototype specifies how to create some specific disclosure that is reported within a financial report. If you were going to create a new disclosure you might use an exemplar from some other public company that has created the same disclosure or you might use a prototype or template that exemplifies a disclosure.

Prototypes are useful not only for creating new disclosures but are also useful in querying for disclosure information within some set of financial reports.

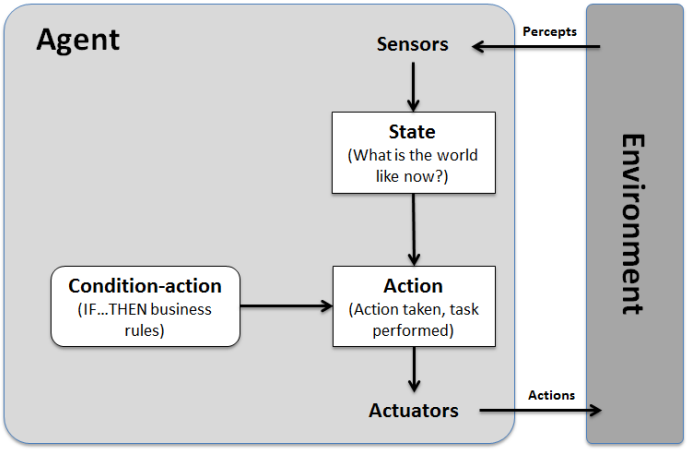
1.4.13. Business Rules Engine

Pesseract implements a deductive reasoning system that uses forward chaining. An off-the-shelf Microsoft.Net rules engine was used under the hood, but that rules engine was significantly constrained by the conceptual model of a business report and financial report.

An XBRL Formula processor is not sufficient[53] for the tasks required by professional accountants creating financial reports. As such, XBRL Formula related functionality was supplemented.

1.4.14. Intelligent Agent

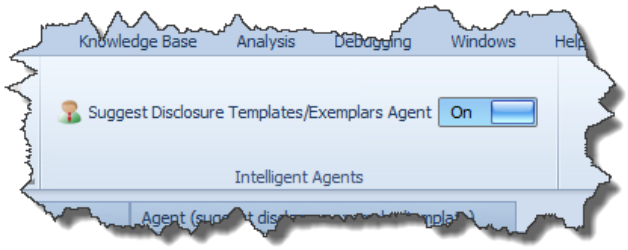
An intelligent agent[54] is an abstract notion that links the real world agent and the notion of agency with an implementation of that functionality within software. An intelligent agent is the abstract functionality of a system similar to a computer program; it is not the computer software program itself. An agent is an entity capable of sensing the state of its environment and acting upon it based on a set of specified rules.



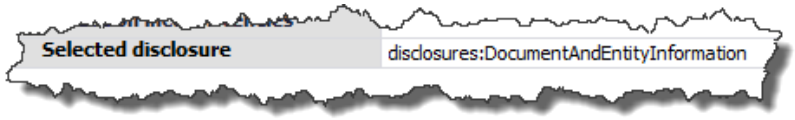
[53] Specific Deficiencies in Capabilities of Existing XBRL Formula Processors, http://xbrl.squarespace.com/journal/2016/9/26/specific-deficiencies-in-capabilities-of-existing-xbrl-formu.html

[54] Comprehensive Introduction to Intelligent Software Agents, http://xbrlsite.azurewebsites.net/2017/IntelligentDigitalFinancialReporting/Part01_Chapter02.7_ComprehensiveIntroductionToIntelligentSoftwareAgents.pdf

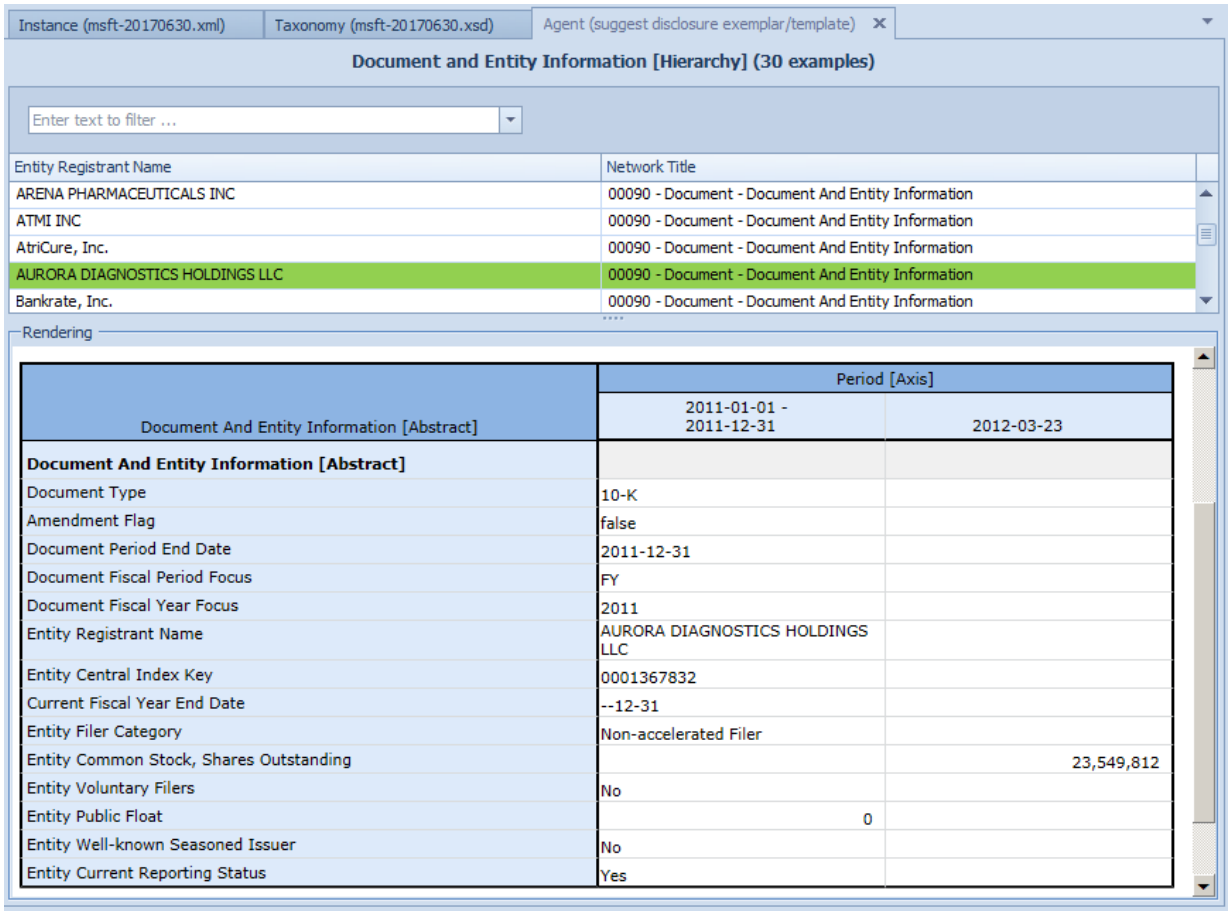
For example, Pesseract has a "suggest disclosure template/exemplar agent":



The agent monitors the state of the "Selected disclosure" value in the state machine returning that value to the agent:



The agent uses that value to provide a list of templates or exemplars which could be useful in helping the user of the software to create that disclosure.



This is only one very basic example of the type of functionality which can be orchestrated by the expert system for creating general purpose financial reports.

1.4.15. Machine-readable Business Rules

To make an expert system work you need both the software that can process and work with machine-readable business rules; but you also need the machine-readable business rules themselves. Without machine-readable rules the machines have nothing to process. Without the rules engine, the utility of machine-readable rules is not really apparent.

But put the rules engine and the machine-readable rules together and something that seems like magic occurs. But it is not really magic.

I have created three important sets of machine-readable rules:

1. Financial reporting using US GAAP.
2. Financial reporting using IFRS.
3. Financial and business reporting using the general profile[55]. (I sometimes refer to this as XASB.)

The most extensive set of metadata exists for US GAAP. Modeled after what was learned from creating the US GAAP metadata, financial reporting metadata was created for IFRS using the same framework. Finally, what I call the "general profile" was created avoiding the bad ideas of the US GAAP/IFRS XBRL Taxonomies and SEC/ESMA and other implementations of XBRL-based financial reporting; and leverages all the good ideas that have been proven to work.

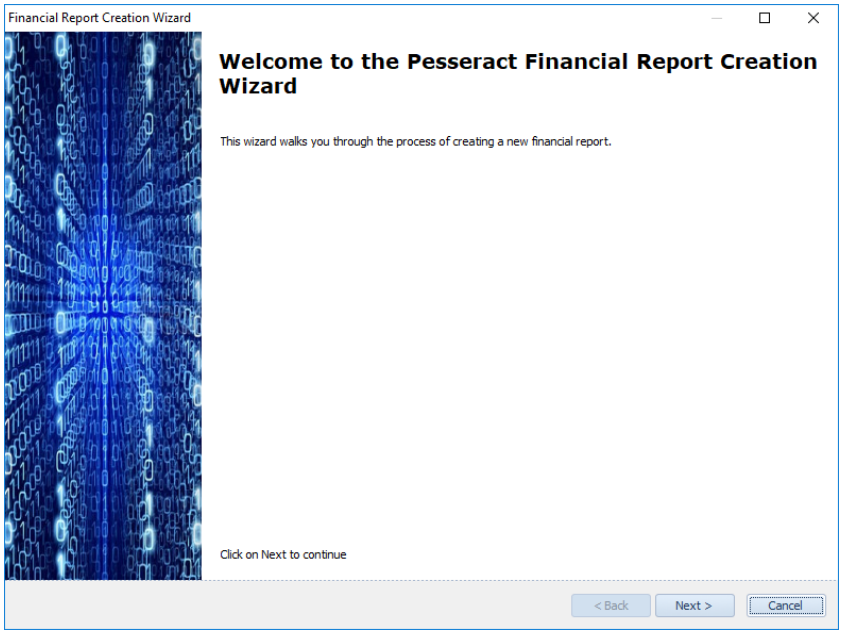
All machine-readable business rules are represented using the XBRL global standard technical syntax. The most current versions of these machine-readable rules can be found on my blog on the Conceptual Model page[56]. If you want to be certain that you have the most current versions, please contact me.

1.4.16. Wizards

A wizard is simply an approach to walking a user of an application through a series of steps. For example, Pesseract has a "Financial Report Creation Wizard" that walks the user through a series of steps that help them create a financial report:

[55] General Business Reporting XBRL Application Profile, http://xbrlsite.azurewebsites.net/2017/Library/GeneralBusinessReportingProfile-2017-12-20.pdf

[56] Conceptual Model of a Digital Financial Report, http://xbrl.squarespace.com/conceptual-model/



The wizard queries the user of the application for the following information:

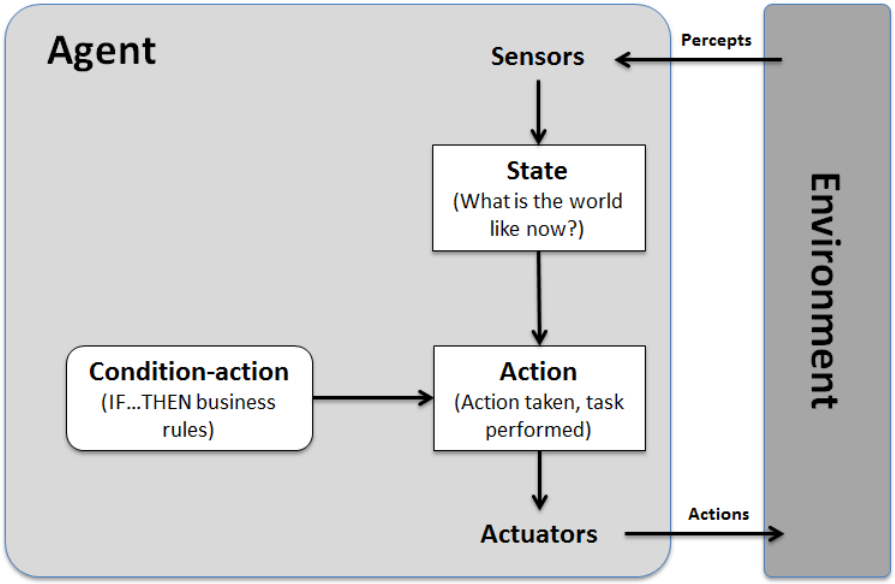
- Which **reporting profile** or reporting scheme are they using?
- What is the **legal form** of the economic entity?
- What is the main **accounting activity** of the economic entity?
- What is the **industry sector** of the economic entity?
- What is the **reporting style** of the economic entity?
- Other report information.
- Other user preferences
- Uses the collected information to suggest a list of **templates** the user can use to populate the report.

The answers to the questions are stored in the **state machine**. Then, the application uses that information to pre-filter **metadata**, point the user in the right direction when performing a task, etc.

1.4.17. Intelligent agents

Intelligent software agents[57] are computer code written in a specific way. An **agent** is an entity capable of **sensing** the **state** of its **environment** and **acting** upon it based on a **set of specified rules**. An agent performs specific tasks on behalf of another. In the case of software, an agent is a software program. Consider that definition of an agent and look at the graphic below to get an idea of how an intelligent agent software works:

[57] *Introduction to Intelligent Software Agents for Business Professionals*, http://xbrlsite.azurewebsites.net/2016/Library/02_IntroducingIntelligentAgents.pdf



An intelligent agent is software that assists people and acts on their behalf. Intelligent agents work by allowing people to:

- delegate work that they could have done to the agent software,
- perform repetitive tasks,
- remember things you forgot,
- intelligently find, filter and summarize complex information,
- customize information to your preferences,
- learn from you and even make recommendations to you.

Pesseract uses a number of intelligent agents to assist the user of the application in the performance of their work, augmenting the knowledge of the application user.

1.5. Testing the System

Pesseract was not created and then tested. The approach used to create Pesseract was to let it evolve based on a reverse-engineering of XBRL-based US GAAP[58] and IFRS[59] financial reports that have been submitted to the U.S. Securities and Exchange Commission (SEC).

What seems to be quite confusing to untrained observes is the inconsistencies that exist within XBRL-based financial reports. Many of these inconsistencies relate to XBRL representation errors in those reports. Common errors are well understood[60]. For those that take the time to understand and even measure inconsistencies and

[58] US GAAP, http://xbrl.squarespace.com/journal/2018/7/28/us-gaap-test-data-2017-10-ks.html

[59] IFRS, http://xbrl.squarespace.com/journal/2018/7/14/updated-list-of-ifrs-filings.html

[60] Digital Financial Report Creation Best Practices, http://xbrlsite.azurewebsites.net/2017/IntelligentDigitalFinancialReporting/Part03_Chapter06.5_DigitalFinancialReportCreationBestPractices.pdf

quality issues[61], how to correctly create an XBRL-based digital financial report becomes quite apparent.

Financial reports are not one big thing. A financial report is multiple much smaller things that interact with one another in known ways. US GAAP was created in the 1930s and has matured into remarkably consistent disclosure patterns which can be understood, measured, and leveraged. Disclosure best practices can be leveraged.

The average accountant just wants to create their disclosures correctly, particularly private companies that have accountants that are generally less skilled than public company accountants. Not every disclosure needs to be a hand-crafted work of art. If software takes care of the routine, mundane disclosures then the time of an accountant is freed up to focus on the areas that truly need the majority of their effort and professional judgement.

Further, the default settings of the software are flexible. While initially, defaults error on the side of using best practices as much as possible (i.e. the default is to opt in to using best practices); software users can choose to consciously over ride best practices choosing whatever approach they may desire to report any specific disclosure. The user is then responsible for any consequences that might result.

Today virtually all software takes the approach to force the professional accountant to figure out everything. One software vendor proudly described this approach as, "Giving the software user enough rope to hang themselves." This is not helpful to software users and laziness on part of the software developer.

What follows is a description of leveragable patterns discovered in XBRL-based financial reports submitted to the SEC by public companies. Most reports used US GAAP but an increasing number if IFRS reports are being analyzed. Using these patterns can make using software easier.

1.5.1.Model Structure

The model structures of XBRL-based reports submitted to the SEC has always been relatively consistent but has become even more consistent over the years. The model structure is simply the relationship between different categories of report elements used to create the model of an XBRL-based report. This matrix shows an analysis of the fiscal year 2015 10-Ks submitted to the SEC using US GAAP (about 6,000 public companies):

		Network	Table	Axis	Member	LineItems	Abstract	Concept
		485,842	212,618	430,552	1,360,484	212,733	727,374	3,155,641
Child	Network	0	0	0	0	0	0	0
	Table	513	0	0	4	4	212,090	11
	Axis	0	430,549	0	0	0	3	0
	Member	0	0	503,078	857,390	3	13	0
	LineItems	29	212,570	0	0	30	104	0
	Abstract	483,334	18	0	2	101,932	141,774	314
	Concept	8	0	1	49	1,178,684	1,969,653	7,246

[61] Quarterly XBRL-based Public Company Financial Report Quality Measurement (June 2018), http://xbrl.squarespace.com/journal/2018/6/29/quarterly-xbrl-based-public-company-financial-report-quality.html

RED cells show XBRL errors of which there are none because XBRL-based reports are 99.99% or better consistent with XBRL technical syntax rules. GREEN cells show appropriate relationships. ORANGE cells show modeling errors. If you add all the values in the ORANGE cells you get only 307 modeling errors for a total of about 6,099,406 relationships. That is an error rate of .005%. YELLOW cells are not considered errors, but they are not best practices either and should be avoided as the relations tend to be illogical.

1.5.2.Reporting Styles

Reporting styles[62] is simply patterns of how companies create balance sheets, income statements, statements of comprehensive income, and cash flow statements. About 90% of all public companies use one of 29 specific reporting styles.

ReportingFrameCode	Filings	Filings with No Errors	Total errors	Average errors	Percent without Error	% of Total Filings	Cumulative %
COMID-BSC-CF1-ISM-IEMIB-OILY-SPEC6	1,911	1,701	333	.2	89.20%	32%	32%
COMID-BSC-CF1-ISS-IEMIB-OILY-SPEC1	847	746	156	.2	88.08%	14%	46%
COMID-BSC-CF1-ISS-IEMIB-OILY-SPEC2	742	679	92	.1	91.51%	13%	59%
INTBX-BSU-CF1-ISS-IEMIX-OILN	460	427	48	.1	92.83%	8%	67%
COMID-BSC-CF1-ISM-IEMIB-OILY-SPEC9	160	152	10	.1	95.00%	[illegible]	69%
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
INTBX-BSU-CF1-ISS-IEMIX-OILN-[illegible]	22	15	11	.5	68.18%	0%	89%
COMID-BSU-CF1-IS6-IEMIX-OILN	21	19	2	.1	90.48%	0%	89%
COMID-BSC-CF1-ISM-IEMIX-OILY-SPEC7	21	13	9	.4	61.90%	0%	89%
COMID-BSC-CF1-ISS-IEMIB-OILY-SPEC1A	20	19	1	.1	95.00%	0%	90%
Total	5,326	4,711	915				

IFRS reporting styles[63] follow a similar pattern.

1.5.3.Fundamental Accounting Concept Relations

What different reporting styles do is organized fundamental accounting concept relations differently. For example, a classified balance sheet and an unclassified balance sheet organize concepts differently. The fundamental accounting concept relations is a mechanism of testing these relationships.

Per my last measurement[64] of US GAAP XBRL-based financial reports,

- 88.9% of all XBRL-based reports were consistent with all fundamental accounting concept relations rules.
- 99.2% of all relations within XBRL-based reports where consistent with fundamental accounting concept relations rules.
- On an individual test basis[65], the majority of tests were 99% consistent or greater and the lowest consistency level was 96.5%.

---

[62] Making the Case for Reporting Styles, http://xbrlsite.azurewebsites.net/2017/library/MakingTheCaseForReportingStyles.pdf

[63] IFRS Reporting Styles, http://www.xbrlsite.com/2018/IFRS/IFRS-Reporting-Styles.pdf

[64] Quarterly XBRL-based Public Company Financial Report Quality Measurement (June 2018), http://xbrl.squarespace.com/journal/2018/6/29/quarterly-xbrl-based-public-company-financial-report-quality.html

[65] Individual test basis, http://xbrlsite.azurewebsites.net/2018/Library/Signals_2018-06-30.jpg

Basically, US GAAP XBRL-based reports are remarkably consistent.

IFRS is undergoing similar testing and already overall consistency on a test basis is 98.2% with a margin of error of about 1%.

1.5.4.Reporting Checklist and Disclosure Mechanics

Reporting checklist rules and disclosure mechanics rules are the least tested but still relatively consistent. At this point testing of 65 disclosure for US GAAP yields 90% consistency[66]. The range on a per disclosure basis goes from 100% consistent to 61% consistent[67].

1.5.5.Type/Class Relations

Type/class relations are currently not a priority but the framework of testing these relations has been created and several rules have been created to exercise the framework. For example, a type/class rule might state that the concept "us-gaap:CostOfRevenues" is not ever to be a part of "us-gaap:OperatingExpenses" rather it is a part of "us-gaap:CostsAndExpenses" and is used to help professional accountants not make the mistake of modeling this relation incorrectly.

1.5.6.Final Words about Testing

For more information related to testing of digital financial reports, please read *Blueprint for Creating Zero-defect XBRL-based Digital Financial Reports*[68]. Note that there are two different versions of machine-readable rules that have been implemented. Both implementations have been reconciled to one another. Each approach has pros and cons. The most superior version, in my view, is the pure XBRL version which uses the XBRL global standard to represent all machine-readable rules used. The primary downside of that version is the verbose nature of XBRL Formula used to represent rules.

Note that for creating reports is it is not necessary to support 100% of all reporting entities in the beginning. The volume of business rules that need to be created is high, the tools for creating such rules are scarce, and the industry coverage and industry expertise will be diverse and broad. The volume of business rules will ultimately grow; the most important concern right now is not volume, but rather quality.

1.6. Conclusion

For those willing to pay attention to details; they will recognize that the "I"s have been dotted and the "T"s crossed in terms of testing and proving that building an expert system for creating general purpose financial reports is possible[69].

---

[66] Disclosure mechanics by generator, http://xbrlsite.azurewebsites.net/2018/Library/DisclosureMechanics_2018-03-31.jpg

[67] Disclosure mechanics by disclosure, http://xbrlsite.azurewebsites.net/2018/Library/DisclosureMechanics_ByDisclosure_2018-03-31.jpg

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

[69] For those that are really paying attention, they will recognize that a financial report is simply a complex business report and that what is being learned from analyzing financial reports is likewise useful for creating business reports in general.

About 80% of the work related to building this expert system has been overcoming limitations and inconsistencies of XBRL-based reports submitted to the SEC. On the one hand, it may seem like a risk to reduce the alternatives allowed to potential customers of a product for creating digital financial reports. But on the other hand, taking that risk has significant rewards. Careful analysis and good choices can maximize benefits to both users of software applications being created and to the developers creating those applications. Having the correct knowledge as to how XBRL-based reports work helps you reduce the risk of making a bad implementation choice.

Here is one example. The debate over whether [Table]s should be required has gone on ad nausea. What many uninformed people don't yet understand is that it is easier to require the use of a [Table] if software is created correctly because the choice professional accountants have to make between whether to use a [Table] or not becomes moot. Basically, the software can decide. Already, certain filing agents put [Table]s in 100% of their filings. Basically, some filing agents and software vendors have reached the obvious conclusion. So the risk of following the lead of other software vendors and filing agents of decreasing options to accounting professionals is going down. This trend will continue. Ultimately, professional accountants will not care when (a) the software is easy to use and (b) quality results can be shown.

Basically, a "sweet spot" exists. In the past technical decisions and business domain decisions were intermingled, generally misunderstood due to lack of understanding and poor communication, and the result were bad software implementation choices. Those bad choices where expensive for software companies and resulted in software that is not up to the task of creating high-quality financial reports reliably, every time. Different choices are possible, even preferable.

A significant ramification of these decisions is software that serves only the market of companies that are compelled to create XBRL-based reports due to a regulator mandate; about perhaps 10,000 companies in the US that are compelled to report to the SEC and perhaps another 10,000 that are compelled to report to ESMA.

What is missed is the market of 26,000,000 private companies in the US alone and probably similar numbers in Europe and the rest of the world. No private company in their right mind would purchase software that does what they are doing today, but it is harder to use and more expensive. That would be a foolish choice.

But what if the software was easier to use, the quality of the financial reports was higher than it is today, total cost for creating reports was lower, and financial reports could be created faster. Would private companies, and perhaps public companies, buy that software? I will let you contemplate that and answer the question yourself.

It is hoped that the ideas I am sharing in this document will start a discussion and that the collective wisdom of the institution of accountancy will correct what I may have gotten incorrect.

1.6.1.Issues with Pesseract

Although Pesseract proves the concept that software can be built to enable professional accountants to create financial reports using an expert system, Pesseract does have issues which other such systems might choose to address. From my perspective, the following is a summary of those issues which should be perhaps avoided by other software engineers building similar software:

1. Pesseract is "batch" oriented currently in some cases. What that means is that validation is not real time as report fragments are being created; rather some validation is run after-the-fact basically in batch mode. Real time would be preferable.
2. Pesseract is locally installable software. It may, or may not, be preferable to create a browser-based version of this sort of software.
3. Pesseract is single user. Many companies require multi-user software because they have multiple people creating financial reports.
4. Business rules used by Pesseract might not be "world-class" architecture. The schemes for articulating machine-readable business rules using XBRL definition relations and XBRL Formula works, but that approach might not be the best approach to use. The jury is still out.
5. The power or expressiveness of business rules and the ability of the Pesseract business rules engine to process rules is questionable. It is simply hard to believe that the Pesseract system can be as robust as something like TopBraid or ErgoAI. Again, the jury is still out.
6. Pesseract can be used to create structural models and gather facts from other systems or input facts manually; however, it cannot be used present facts precisely using Inline XBRL. But, Pesseract can auto-generate Inline XBRL.

These issues may, or may not, be addressed as Pesseract transitions from a working proof of concept to commercial software.

1.6.2.Digital Financial Report Creation using the Semantic Web Stack

Intuitively, it seems to me that the most viable option for creating general purpose financial reports is to: (a) recast the model of the report more cleanly and succinctly using RDF; OWL; and RIF, SPIN, SWRL, or SHACL. The reason is that those standards are better thought through in terms of semantics and the reasoning engines are more complete in terms of the necessary functionality. One day, conversion to/from XBRL technical syntax to/from the semantic web stack of technologies syntax one day will be a trivial process.

But admittedly, understanding the best technical approach to use is not my strength because I am a professional accountant, not an information technology professional. That said, I do understand the issues more than the average software developer who knows nothing of the semantic web stack of technologies.

As such, I defer to technical professionals to make this decision.

1.6.3.Next steps

If you want more details, here is where you can get those details. That next level can be found in ***Intelligent XBRL-based Digital Financial Reporting***[70]. This document is an accumulation of almost 20 years of effort to construct an expert system for creating financial reports. The information has been organized and synthesized to a degree. The document is far from perfect, but all the information that you need is very likely there. I will be rewriting that document at some point in the future to improve it, but I am not sure when I will begin that project.

[70] Charles Hoffman and Rene van Egmond, *Intelligent XBRL-based Digital Financial Reporting*, http://xbrl.squarespace.com/intelligent-xbrl/