4. Knowledge Engineering Basics for Accounting Professionals

Computers sometimes seem to perform magic. But computers are simply machines. Computers do not create that magic. Skilled craftsmen who wield their tools effectively is how the magic is created. Computers simply follow instructions. Computer science is to domain in which information technologies operate. Notice the "science" part of the term computer science. Ultimately making a computer perform work distills down to logic and mathematics.

Digital has positive characteristics, but just like anything else it also has negative or less favorable characteristics. What does *digital* mean? How does *digital* work? How does the financial reporting supply chain want *digital* to work? Professional accountants cannot afford to be ignorant of how to make *digital* work the way they want digital to work for them. Becoming skilled in the science and art of digital is crucial for professional accountants.

It is important to define the term engineering. Engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of something." Building a bridge and engineering a bridge are different things.

Bridges are engineered when they are constructed. Engineering is the skillful construction or creation of something leveraging known laws of how things interact with one another. A civil engineer does not simply throw concrete and steel together to construct a bridge. The bridge is engineered to balance cost, strength, likelihood that the bridge remains standing during high winds or an earthquake, etc. Likewise when we work with information using a computer, how we achieve our goals and objectives is an engineering process, not simply throwing a few things together. That process is often referred to as knowledge engineering.

In order to understand how to get computers to perform work, you need to understand how computers work. You have to understand some basics of knowledge engineering.

4.1. Understanding cognitive computing

The article, *What is cognitive computing? IBM Watson as an example*[38], points out what it claims to be the three eras of computing:

- First era, tabulating systems (1900),
- Second era, programmable systems (1950),
- Third era, cognitive computing.

Many people tend to miss a really important point when it comes to XBRL. They look at XBRL as "tagging" and they equate XBRL to the "barcode". But, turn the equation around. What would it mean if all financial information were properly bar coded? What could you do? What would that enable?

[38] *What is cognitive computing? IBM Watson as an example*, http://www.duperrin.com/english/2014/05/27/whats-cognitive-computing-ibm-watson-example/

Think about it: What would happen if you took financial reports and rather than reporting information in an unstructured format that only machines could understand (because the information is unstructured or more accurately structured for presentation); but rather reported information was in a structured format both humans and machines could read and understand? What exactly would that mean?

Cognitive computing is the simulation of human thought processes in a computerized model.

Notice the word simulation. Computers cannot think; they are dumb beasts. But these dumb beasts can be made to mimic the human thought process, if you understand how to harness the power of a computer. If you know how to harness the power of a computer, you can make a computer seem to perform magic.

Computers do not create the magic. Skilled craftsmen who wield their tools effectively are what create the magic. Commuters simply follow instructions.

This article, *Cognitive Computing and Semantic Technology: When Worlds Connect*[39], points out something that is really important in two key statements in that article:

First:

> For cognitive computing to achieve its promise we need a thick metadata layer that incorporate semantic tagging formats.

Second:

> A lot of the focus is on machine learning, especially as things move to really analyzing and building explicit knowledge models, but other areas that should be included in the cognitive computing mix include constructive ontologies and constructive knowledge modeling, whether it's done by groups or individuals or crowd-sourced in the case of the semantic web.

So, what is not in dispute is the need for a "thick metadata layer" in order for the computer to be able to perform useful work. But what is sometimes disputed, it seems, is HOW to get that thick metadata layer. There are two basic approaches:

- **Have the computer figure out what the metadata is**: This approach uses artificial intelligence, machine learning, and other high-tech approaches to detecting patterns and figuring out the metadata.
- **Tell the computer what the metadata is**: This approach leverages business domain experts and knowledge engineers to piece together the metadata so that the metadata becomes available.

As a professional accountant, I understand that the probability of a computer starting from scratch and using the most sophisticated technologies and approaches available today and creating any useful metadata is very close to zero. However, the more manually created metadata that a computer has to work with, the higher the probability that the computer would be helpful in correctly figuring out financial reporting metadata.

So what I am saying is that humans are going to have to prime the pump and get quite a lot of metadata pieced together. Then at some point and for some things, computers can effectively be used to contribute more metadata. And so, this is not

39 *Cognitive Computing and Semantic Technology: When Worlds Connect*, http://www.dataversity.net/cognitive-computing-semantic-technology-worlds-connect/

an either-or question. Both approaches can be used effectively and contribute to what is needed to realize the potential offered by cognitive computing.

Computers assisting professional accountants in correctly representing financial reports digitally will cause high-quality financial information to be available for analysis by investors and regulators. Everyone in the financial reporting supply chain will benefit from the meaningful exchange of financial information in machine-readable formats.

Concept computing will contribute to changing how financial reports are created similar to how CAD/CAM contributed to how blueprints are created and how the design supply chain interacts.

So how exactly do we get computers to perform work for use? Knowledge engineering, the science of making computer perform magic.

4.2. Understanding expert systems

Basically, financial report creation software will become what amounts to an expert system for creating financial reports. In his book, *Systematic Introduction to Expert Systems*[40], Frank Puppe describes what an expert system is, how they work, and what they can achieve.

First, a good definition of an expert system[41] is the following:

> **Expert systems** are computer programs that are built to mimic or simulate or emulate human behavior and knowledge; expert systems are computer application that performs a task that would otherwise be performed by a human expert.

Expert systems solve problems by reasoning about knowledge represented in machine-readable form as "IF…THEN" rules that the machine simply follows. Computers really are not thinking, they are only mimicking or simulating or emulating human though by following a clearly laid out set of machine-readable instructions to perform some task.

Frank Puppe explains in his book 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.
- **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.

A digital financial report creation tool is basically an expert system that helps its user, a professional accountant, assemble and generate an external financial report. The final product, the financial report, could be generated in human-readable form like the HTML, PDF, or word processing document-type outputs; and/or in machine-readable form such as XBRL or RDF/OWL.

[40] *Systematic Introduction to Expert Systems*, Frank Puppe, http://www.amazon.com/Systematic-Introduction-Expert-Systems-Representations/dp/3642779735/
[41] Expert systems, https://en.wikipedia.org/wiki/Expert_system

Chapter 21, *Review of the Problem-Solving Type Construction*, of Frank Puppe's book describes in detail how this would work. That chapter proves an excellent explanation and enough details to help you get your head around how expert systems work.

4.3. Reasoner

A reasoner is software that is able to infer logical consequences from a set of asserted facts. Every reasoner uses some sort of logic. For example, first-order predicate logic is a type of logic. Every reasoner works with some set of axioms. An axiom describes some logical fact. The capabilities of a reasoner depend on the expressiveness of the kind of logic that the reasoner uses and the axioms provided for the reasoner and logic to work against.

Reasoners are sometimes referred to as inference engines because while, as stated above, reasoners work with asserted facts; reasoners can also use the rule of logic to deduce theorems. Theorems are indirectly deduced facts. Theorems are deductions which can be proven by constructing a chain of reasoning by applying axioms. Basically, a reasoned and an inference engine is the same thing.

Clearly a human's capacity to apply logic is greater than a computer's capacity to apply logic[42]. Care has to be taken in order to express facts in a form that is safe, reliable, predictable, and repeatable. There are four catastrophic problems[43] that a computer can run up against;

- **Undecidability**: If a question cannot be resolved to a TRUE or FALSE answer; for example if the computer returns UNKNOWN then unpredictable results can be returned. Logic used by a computer must be decidable.
- **Infinite loops**: If a computer somehow enters an infinite loop from which it cannot return because of a logic error or because the logic is too complex for the machine to work with; the machine will simply stop working or return nonsense.
- **Unbounded structures or pieces**: Systems need boundaries for them to work correctly. If a system does not have the proper boundaries, then a machine can become confused or not understand how to work with information that is provided.
- **Unspecific or imprecise logic**: Confusing precise results with the capabilities of a computer to provide a statistically created result can cause problems. It is not expected that the business system at the level of describing the things in the system be able to support "fuzzy logic" or "probabilistic reasoning" or other such functionality.

Correctly balancing the expressiveness of a logic and the safety, reliability and predictably of a piece of software to return useful information takes conscious, skillful effort and execution. Years of experimentation in the area of expert systems and artificial intelligence has yielded invaluable information in achieving this balance.

[42] Logic in computer science, https://en.wikipedia.org/wiki/Logic_in_computer_science#Logic_applications_for_computers

[43] Brainstorming Idea of Logical Catastrophes or Failure Points, http://xbrl.squarespace.com/journal/2015/7/25/brainstorming-idea-of-logical-catastrophes-or-failure-points.html

While first-order predicate logic is expressive and powerful in performing work, it is not decidable and other problems can occur. In creating PROLOG, this problem was partially addressed by limiting which first-order predict logic statements can be used to a Horn clause[44]. But even PROLOG had issues and so further restrictions were made to first-order logic expressed using Horn clauses and Datalog[45] was created.

State-of-the-art semantic web technologies such as OWL 2 DL[46] have been limited to solve the problem of decidability by limiting the logic to *SROIQ* description logic. However, *SROIQ* description logic does not support expressing mathematical relations.

And so the question is: what is the correct balance of expressive power and safety/reliability/predictability? Currently there is no global-standard answer to this question.

4.4. Strengths of computers

Computers have four fundamental strengths:

1. **Storage**: Computers can store tremendous amounts of information reliably and efficiently.
2. **Retrieval**: Computers can retrieve tremendous amounts of information reliably and efficiently.
3. **Processing**: Computers can process stored information reliably and efficiently, mechanically repeating the same process over and over.
4. **Ubiquitous information distribution**: Computers can make information instantly accessible to individuals and more importantly other machine-based processes[47] anywhere on the planet in real time via the internet, simultaneously to all individuals.

So how do you harness this power provided by computers?

4.5. Major obstacles to harnessing the power of computers

However, there are a number of major obstacles to harnessing the power of computers to perform work for business professionals within one department, in an organization or across an entire supply chain. These obstacles include:

1. **Business professional idiosyncrasies**: The first obstacle is that different business professionals use different terminologies to refer to exactly the same thing.
2. **Information technology idiosyncrasies**: The second obstacle is that information technology professionals use different technology options[48],

[44] Horn clause, https://en.wikipedia.org/wiki/Horn_clause
[45] Datalog, https://en.wikipedia.org/wiki/Datalog
[46] OWL 2 DL, http://www.w3.org/TR/owl2-overview/
[47] Wired: *The Web is Dead: Long Live the Internet*, http://xbrl.squarespace.com/journal/2010/9/24/wired-the-web-is-dead-long-live-the-internet.html
[48] See *Understanding Database and Query Options (Part 2)*, http://xbrl.squarespace.com/journal/2014/4/27/understanding-databasequery-options-part-2.html

techniques[49], and formats[50] to encode information and store exactly the same information.

3. **Inconsistent domain understanding of and technology's limitations in expressing interconnections**: A third obstacle is that information is not just a long list of facts, but rather these facts are logically interconnected and generally used within sets which can be dynamic and used one way by one business professional and some other way by another business professional or by the same business professional at some different point in time. These relations are many times more detailed and complex than the typical computer database can handle. Business professionals sometimes do not understand that certain relations exist.

4. **Computers are dumb beasts**: The forth obstacle is that computers don't understand themselves, the programs they run, or the information that they work with. Computers are dumb beasts. What computers do can sometimes seem magical. But in reality, computers are only as smart as the metadata they are given to work with, the programs that humans create, and the data that exists in databases that the computers work with.

Computers do not create the magic. Skilled craftsmen who wield their tools effectively are what create the magic. Computers simply follow instructions.

If two computers use the same information formats and other technology aspects but use different terminology or different information organization strategies, the two computers will find it difficult or even impossible to interoperate. If this is the case, the only way to cross the chasm between these two different computers is with human intervention. Often this involves re-keying information. Saying this another way, in order for two computers to interoperate it is essential that every aspect including terminology, world view, information formats, instructions and so forth necessary to translate from one computer to the second computer must be explicitly provided.

Getting computers to perform work is straightforward science: The only way a meaningful exchange of information can occur is the prior existence and agreement on technical syntax rules, business domain semantics rules, workflow/process rules, and the information with which the computer will be working.

Computers are only able to reason with information that they have explicitly been given[51]. Remember, computers are dumb. Computers are incapable of implying meaning. This means that if the information is vague, inconsistent, logically incoherent, contradictory, ambiguous or in any other way unclear; the computer programmed to reason or use such information will produce either nothing at all or results which are likewise vague, inconsistent, logically incoherent, contradictory, ambiguous, or in some other way unclear.

It really is that straightforward: Nonsense-in-nonsense-out.

Computers cannot check the factual accuracy of information against reality. If the person who put the information into the computer made a mistake or intentionally

[49] See *Understanding Database and Query Options (Part 1)*, http://xbrl.squarespace.com/journal/2014/4/26/understanding-databasequery-options-part-1.html

[50] See *Understanding Syntax*, http://xbrl.squarespace.com/journal/2014/3/30/understanding-syntax.html

[51] Closed world assumption, http://en.wikipedia.org/wiki/Closed-world_assumption

entered the wrong information (i.e. fraud); that is exactly what the computer has to work with.

Finally there is setting the right expectations. Business professionals need to understand what computers can and cannot do. Computers cannot perform magic[52]. Computers fundamentally follow the rules of mathematics which follow the rules of formal logic. It really is that straight forward. Computers cannot effectively work with information such as the following:

- fuzzy expressions[53] - "It **often** rains in autumn."
- non-monotonicity[54] - "Birds fly, penguin is a bird, but penguin does not fly."
- propositional attitudes[55] - "Eve **thinks** that 2 is not a prime number." (It is true that she thinks it, but what she thinks is not true.)
- modal logic[56]
  - possibility and necessity - "It is **possible** that it will rain today."
  - epistemic modalities - "Eve **knows** that 2 is a prime number."
  - temporal logic - "I am **always** hungry."
  - deontic logic - "You **must** do this."

Computers can be provided with instructions in the form of explicit information which helps them mimic or seem to be able decipher such information; but it was really the business professional or information technology professional that created the instructions that made that happen.

At this time in history it is not possible for computers to think like human beings. Could it be possible in principle for computers to reason at some point in the future? Maybe. Artificial intelligence researchers have been working on this task for years but have been here-to-fore unsuccessful. IBM's Watson[57] is not intelligent. It only seems intelligent because the information used by Watson is clear, consistent, logically coherent, and unambiguous.

Overstating what a machine such as a computer can do is not wise. It is also not wise to either misunderstand the capabilities of a computer or to misinterpret what it takes to make a computer successful in performing the work that computers are capable of performing. Computers can perform specific types of work extremely well. Computers are machines that are very adept at reliably performing repetitive mindless tasks accurately. Even very sophisticated repetitive tasks can be performed by computers.

4.6. Computers are tools

In the hands of a skilled craftsman the right tools can produce quality results. In the wrong hands, the same tool might produce poor results.

---

[52] *Limitations of First-order logic expressiveness*, http://dior.ics.muni.cz/~makub/owl/

[53] Fuzzy logic, http://en.wikipedia.org/wiki/Fuzzy_logic

[54] Non-monotonicity, http://en.wikipedia.org/wiki/Non-monotonic_logic

[55] Propositional attitudes, http://en.wikipedia.org/wiki/Propositional_attitude

[56] Model logic, http://en.wikipedia.org/wiki/Modal_logic

[57] *IBM's Watson not as smart as you think*, http://www.computerworld.com/article/2507369/emerging-technology/ibm-s-watson-not-as-smart-as-you-think.html

And so as was pointed out there are a number of problems that need to be worked through in order to get computers to successfully perform the tasks that they are well suited to perform: reliably store a tremendous amount of information and reliably and automatically retrieve and work with that information by anyone from anywhere in real time. The idiosyncrasies of business professionals need to be worked through, the idiosyncrasies of programmers, the idiosyncrasies of database builders and other information technology professionals need to be overcome. Computer languages and programs with sufficient expressive power to handle the richness of business information from complex business structures and transactions, different legal and cultural structures and so forth need to be created and implemented by information technology professionals for business professionals. Substantial care needs to be taken to ensure that the things and relations between the things within a problem domain are clear, logically coherent, consistent, unambiguous, and otherwise well-defined, precise, and accurate to reflect the facts of reality as currently reflected and flexible enough to change as today's dynamic business environment changes.

Flexibility and inflexibility/rigidity need to be appropriately balanced. Equilibrium achieved. The answer to all of these challenges, the state-of-the-art solution to the real problems of getting many business systems to successfully interoperate with many other business systems in a distributed environment[58] such that a meaningful exchange of information can occur between business systems is "standards based ontology".

Such a system must be reliable, repeatable, predictable, safe, cost effective, easy to use, robust, scalable, secure as deemed required, auditable (track provenance) as deemed necessary.

4.7. Ontologies are tools

The term ontology has been used in philosophy for thousands of years going back to the father of formal logic, Aristotle[59] (400 B.C.). Ontology is defined as the study of the things and the relations between things that exist in reality. The goal of philosophical ontology is to provide deliberate, clear, coherent and rigorously worked out accounts of the basic structures found in reality.

In more current times, the term ontology has become prominent in the area of computer science and information science. In computer science the term ontology generally refers to the standardization of a terminology framework such that information repositories can be constructed. Ontologies used by philosophers like Aristotle were not machine-readable. Ontologies used by computers are machine-readable.

The problem that ontologies solve is not that of simply coming up with a set of terms such as a dictionary or creating basic relations between terms such as a thesaurus or even more complex relations between terms expressed by a taxonomy. Rather, an ontology defines terms, organizes the terms into categories or classes, and determines as many important universal relations as practical and necessary between the categories or classes within some business problem domain. Ontologies are the pinnacle of expressiveness.

[58] *Understanding Distributed Extensibility*, http://xbrl.squarespace.com/journal/2015/1/7/understanding-distributed-extensibility.html
[59] Aristotle's epistemology, http://en.wikipedia.org/wiki/Aristotle#Aristotle.27s_epistemology

To understand this, it is helpful to look at the differences between a dictionary, a classification, a taxonomy, and an ontology.

- A **dictionary** is much like a list, a dictionary had no hierarchy. A dictionary is simply a flat list of terms with no relations expressed between the terms.
- A **classification system** is a grouping of something based on some criteria. A classification tends to not have a hierarchy.
- A **taxonomy** is a classification system which does have a hierarchy, but the hierarchy tends to be less formal and less sophisticated. Taxonomies hierarchies tend to be trees of information.
- An **ontology** is a set of well-defined concepts which describes a specific domain. Ontologies are generally defined using class and subclass relations. Classes and subclasses have defined properties and relations. The goal of an ontology is to provide a formal, reference-able set of classes and relations which are used in communications within some domain. So, an ontology is also expressed as a hierarchy, but the hierarchy is more explicit and much richer in meaning than a taxonomy. Ontology hierarchies tend to be more like graphs[60] or networks[61].

4.8. Limitations of classification systems

There is a significant difference between a dictionary and an ontology. The focus should not be a definition[62] but rather the focus should be on the purpose of a concept and its relationships with other concepts in the knowledge domain. There is no single definition that is "the truth"; rather there are many definitions, depending on the context.

There is both a glossary and knowledge models reference that each other. A flexible graph of knowledge is the result rather than one rigid dictionary.

At the same time we say that ontologies are powerful tools, we also need to point out that every classification system ever devised by humans has deficiencies of some sort. David Wenberger's book *Everything Is Miscellaneous*[63] points out two important things:

- That every classification scheme ever devised inherently reflects the biases of those that constructed the classification system.
- The role metadata plays in allowing you to create your own custom classification system so you can have the view of something that you want.

As we move from "atoms" to "bits", people drag along the rules which apply to atoms and try to apply those rules to solve problems in the world of bits. This, of course, does not work. *Everything Is Miscellaneous* has countless examples

60 Graph theory, http://en.wikipedia.org/wiki/Graph_theory
61 Network theory, http://en.wikipedia.org/wiki/Network_theory
62 On the value of definitions, http://fadyart.com/en/index.php?option=com_content&view=article&id=140%3Aon-the-value-of-definitions
63 *Everything is Miscellaneous*, http://goo.gl/dZD3lq

contrasting the physical organization of atoms (such as books in a book store) and the organization of books digitally (like Amazon.com).

The second thing pointed out is the three orders of order and the power of the Third Order of Order. There are three orders of order:

- **First order of order**. Putting books on shelves is an example the first order of order.
- **Second order of order**. Creating a list of books on the shelves you have is an example of second order of order. This can be done on paper or it can be done in a database.
- **Third order of order**. Adding even more information to information is an example of third order of order. Using the book example, classifying books by genre, best sellers, featured books, bargin books, books which one of your friends has read; basically there are countless ways to organize something.

Third order removes the limitations which people seem to assume exist when it comes to organizing information. Weinberger says this about the third order of order:

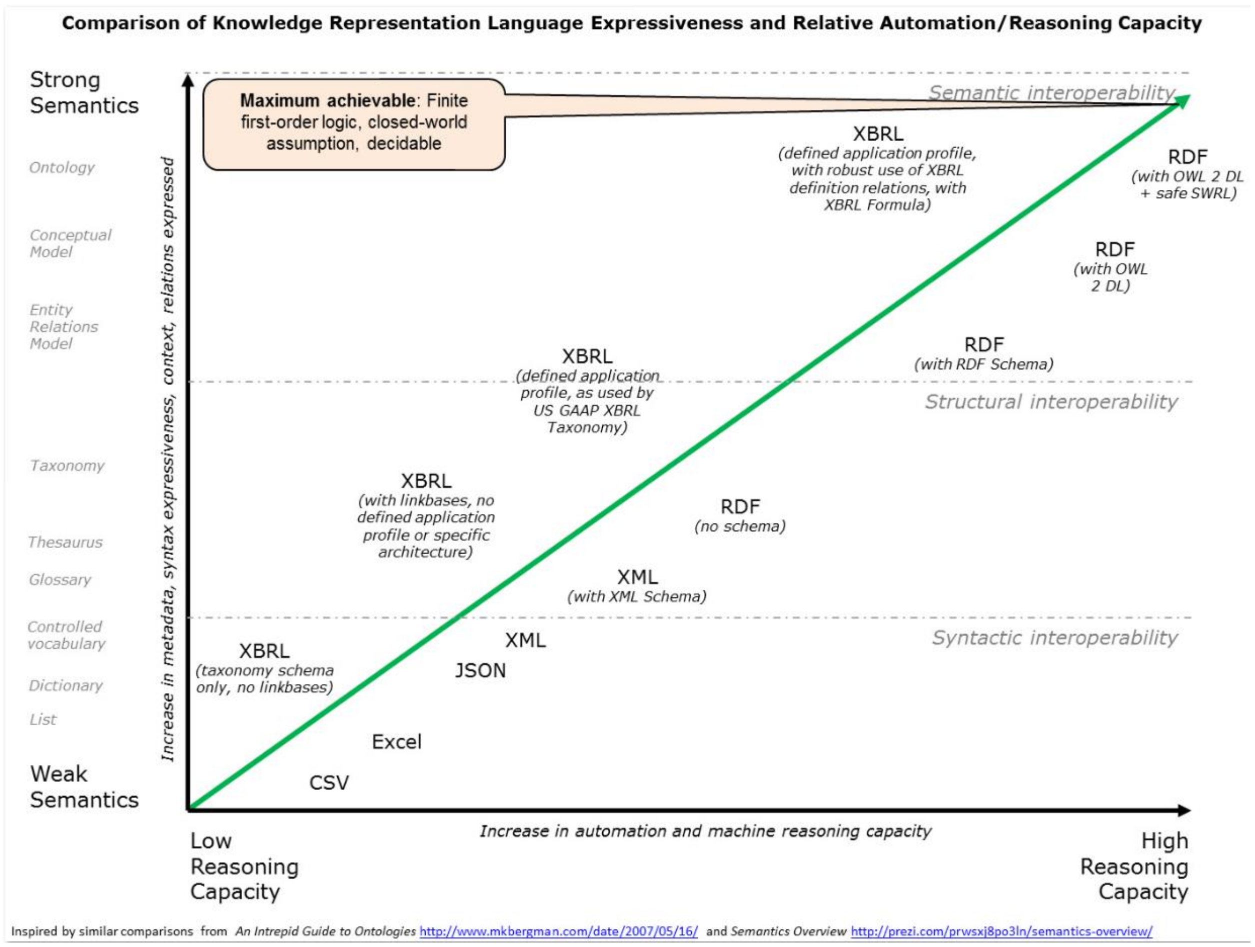
> *In fact, the third-order practices that make a company's existing assets more profitable, increase customer loyalty, and seriously reduce costs are the Trojan horse of the information age. As we all get used to them, third-order practices undermine some of our most deeply ingrained ways of thinking about the world and our knowledge of it.*

And so at the same time we say that ontologies are powerful tools and that every classification ever devised has deficiencies; we also point out that more metadata and relations are better than less. Again, creating an appropriate ontology is a balancing act, striking the appropriate equilibrium is the goal.

4.9. Relation between expressiveness and reasoning capacity

The diagram below compares the relative reasoning capacity which is achievable give the semantic power or expressiveness of some language. The goal is to maximize the reasoning capacity that can be achieve, or said another way the ability of a computer to automate work[64].

[64] This diagram is inspired by two other diagrams. First, from *An Intrepid Guide to Ontologies*, http://www.mkbergman.com/date/2007/05/16/; Second, from the presentation *Semantics Overview*, https://prezi.com/prwsxj8po3ln/semantics-overview/


Comparison of Knowledge Representation Language Expressiveness and Relative Automation/Reasoning Capacity
Strong Semantics
Weak Semantics
Ontology
Conceptual Model
Entity Relations Model
Taxonomy
Thesaurus
Glossary
Controlled vocabulary
Dictionary
List
Increase in metadata, syntax expressiveness, context, relations expressed
Maximum achievable: Finite first-order logic, closed-world assumption, decidable
Semantic interoperability
Structural interoperability
Syntactic interoperability
XBRL (defined application profile, with robust use of XBRL definition relations, with XBRL Formula)
RDF (with OWL 2 DL + safe SWRL)
RDF (with OWL 2 DL)
RDF (with RDF Schema)
XBRL (defined application profile, as used by US GAAP XBRL Taxonomy)
XBRL (with linkbases, no defined application profile or specific architecture)
RDF (no schema)
XML (with XML Schema)
XML
JSON
XBRL (taxonomy schema only, no linkbases)
Excel
CSV
Low Reasoning Capacity
Increase in automation and machine reasoning capacity
High Reasoning Capacity
Inspired by similar comparisons from An Intrepid Guide to Ontologies http://www.mkbergman.com/date/2007/05/16/ and Semantics Overview http://prezi.com/prwsxj8po3ln/semantics-overview/

We are not trying to represent data with ontologies; we are trying to represent information for the purpose of gaining knowledge[65]. Keep in mind that we are consciously using the term information and not data. Don't think "data" when we say "information". This summary helps you to understand the difference:

- **Data**: The basic compound for Intelligence is data -- measures and representations of the world around us, presented as external signals and picked up by various sensory instruments and organs. Simplified: raw facts and numbers.
- **Information**: Information is produced by assigning relevant meaning to data. Simplified: information is data in context.
- **Knowledge**: Knowledge is the subjective interpretation of information and approach to act upon the information in the mind of the perceiver. Simplified: knowledge is the interpretation of information.
- **Wisdom (or Intelligence)**: Intelligence or wisdom embodies awareness, insight, moral judgments, and principles to construct new knowledge and

[65] *Understanding Knowledge Modeling*, http://xbrl.squarespace.com/journal/2014/3/24/understanding-knowledge-modeling.html

improve upon existing understanding. Simplified: wisdom is the creation of new knowledge.

Data that is not useable is simply noise. Data without context is not actionable[66]. Information is actionable.

4.10. Understanding what ontologies do

Ontologies overcome the four major obstacles of getting a computer system to perform work discussed previously. Remember the goal: reliable, repeatable, predictable, safe, cost effective, easy to use, robust, scalable, secure when necessary, auditable (track provenance) when necessary.

Ontologies both describe the information being worked with and verify information consistency against that description to avoid information quality problems or inconsistencies. When creating information it is important to verify that what has been created is consistent with the expected description. When consuming information it is important to understand that the information being consumed is consistent with the expected description. Remember: nonsense-in-nonsense-out.

The first two obstacles which related to the problem of business professional idiosyncrasy and technical idiosyncrasy are overcome by using an ontology to standardize terminology. Rather than using arbitrary[67] terminology to express information about some business domain, standard terminology is used. This includes selecting the appropriate important terms and defining the terms in a deliberate, rigorous, clear, logically coherent, consistent, and unambiguous manner. Care is taken to precisely and accurately reflect reality using standard terms.

The third obstacle of expressing the rich logical interconnectedness of facts within and across business systems can be overcome by using general ontological theories disciplined, methodical, and rigorous approach to structuring the relations between terms. Meaning, expressed using machine-readable ontologies, must be exchangeable between business systems not just used within your one system.

Beginning with a rigorous and logically coherent specification of the theoretical information to be implemented makes it possible to address the problems of human idiosyncrasy.

Given the idiosyncratic tendencies of business professionals; interpretations which reflect the arbitrary peculiarities of individuals can sometimes slip in or mistakes can be made when expressing such terminology. Further, parts of our understanding of a business domain can be incorrect and even evolve, improve, or simply change over time.

If different groups of business professional use different terminology for the same concepts and ideas to express the exact same truths about a business domain; those business professionals should be able to inquire as to why these arbitrary terms are used, identify the specific reasoning for this, and specifically identify concepts and ideas which are exactly the same as other concepts and ideas but use different terminology or labels to describe what is in fact exactly the same thing. But to also

[66] Understanding the Term Actionable Information, http://xbrl.squarespace.com/journal/2012/1/18/understanding-the-term-actionable-information.html
[67] Understanding the Difference between Standard and Arbitrary, http://xbrl.squarespace.com/journal/2014/8/22/understanding-the-difference-between-standard-and-arbitrary.html

understand the subtleties and nuances of concepts and ideas which are truly different from other concepts and ideas.

If idiosyncrasies result only in different terms and labels which are used to express the exact same concepts and ideas; then mappings can be created to point out these different terms used to express those same concepts and ideas. Such mappings make dialogue more intelligible and could get groups to accept a single standardized term or set of terminology for the purpose of interacting with common repositories of business information.

If the difference in terminology and expression are rooted in true and real theoretical differences between business professionals, and the different terms express and point out real and important subtleties and nuances between what seemed to be the same terms; then these differences can be made conscious, explicit, clear, and therefore they can discussed, in a rigorous and deliberate fashion because the differences are consciously recognized.

4.11. Knowledge engineering

Explaining exactly how to create an ontology is both beyond the scope of this document and not something that the average business professional needs to concern themselves with. There is a significant difference between the skills needed to create an ontology and use an ontology. Most business professionals will use ontologies rather than create ontologies from scratch. Business professionals will highly-likely append ontologies.

Further, there is a significant difference between creating ontologies for any business domain and creating ontologies for one specific business domain. Most business professionals will work within one business domain or perhaps interact with a handful of other business domains.

This is not to say that business professionals have no role or responsibility in creating ontologies. They do have a role. The first role is to understand knowledge engineering[68] enough in order to grasp the moving pieces. The next role is to communicate with information technology professionals to create tools which abstract away as much of the complexity related to creating and using ontologies away so that business professionals can focus on their business domain. Business professional don't need to concern themselves with the details of exactly how everything works, but they need to have some grasp of the big picture and moving pieces.

While complexity can never be eliminated, complexity can be moved. Using the correct software development approaches, complexity can be buried deep within software that exposes simple to understand ideas to business professionals working with the technology in their business domain.

This is much like software developers creating software using higher-level languages and integrated software development environments rather than programming in assembly language using a text editor.

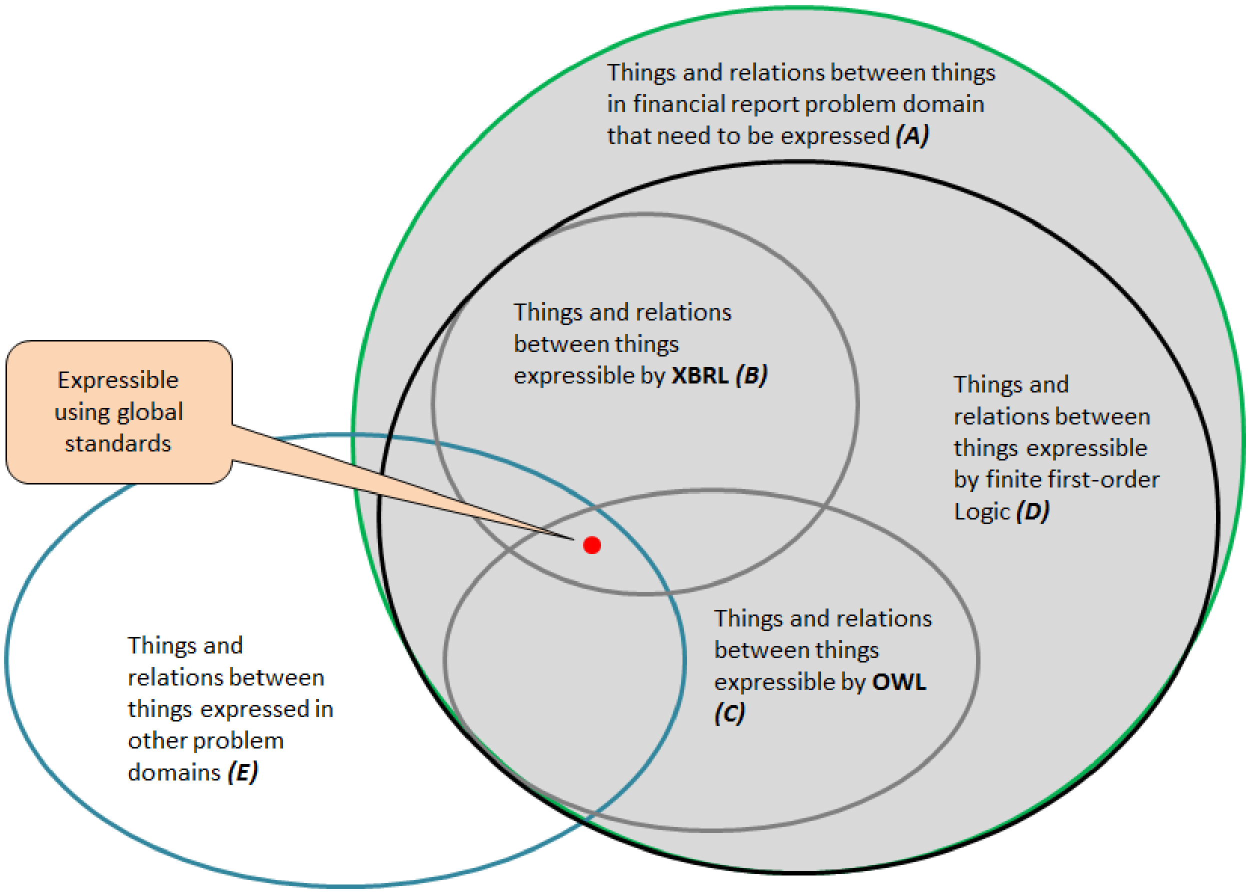
Some business professionals will become skilled knowledge engineers; specialists which help other business professionals create and work with high-quality ontologies to solve specific business domain problems or automate work.

[68] *Knowledge Engineering 101 for Business Professionals*, http://www.xbrlsite.com/2015/Library/IssuesAndConsiderationsInCreatingDigitalFinancialReporting.pdf

4.12. The matter of technical syntax

Lastly is the matter of technical syntax. Ultimately, some technology needs to be used to implement a software-based solution. There are two global standard technical syntax options which are very useful in expressing information about a problem domain in the form of an ontology: XBRL[69] and OWL 2 DL[70]. Both global standard technical syntax options have pros and cons. Neither global standard technical syntax option has the full spectrum of expressiveness necessary to articulate what is necessary for most business problem domains.

The graphic below makes this point using the domain of financial reporting which is a business domain with which I am very familiar. I have been trying to figure out how to create an ontology for a financial report[71] for over 15 years. I have used XBRL, OWL, and proprietary approaches. This is what I have discovered:


Things and relations between things in financial report problem domain that need to be expressed (A)
Things and relations between things expressible by XBRL (B)
Expressible using global standards
Things and relations between things expressible by finite first-order Logic (D)
Things and relations between things expressible by OWL (C)
Things and relations between things expressed in other problem domains (E)

This explains the graphic (note that the size of the circles have no meaning):

- **Theoretical goal**: The green circle with the label "(A)" represents the theoretical goal of expressiveness desirable for the business domain of a financial report. It represents every business rule for every relation anyone would ever want to express related to a financial report. This is a theoretical

69 Extensible Business Reporting Language (XBRL), XBRL International, https://www.xbrl.org/
70 Web Ontology Language (OWL), W3C, http://www.w3.org/standards/semanticweb/ontology
71 Financial Report Ontology, http://xbrl.squarespace.com/financial-report-ontology/

goal because it is highly unlikely that this objective will ever be met because of limitations of technology or the ability of the business domain of a financial report to ever discover and express this information.

- **Achievable using technology today**: The black circle with the label "(D)" indicates what is technically possible to implement today given the current state of technology. The best "bucket" that I can use to express the circle is the notion of finite first-order logic. There could be a better bucket and I cannot articulate the boundaries of the bucket, but it seems like the closest correct bucket because it meets two crucial needs and makes one crucial assumption. The assumption is the closed world assumption[72]. The two needs are the notion of "finite[73]" as contrast to infinite for which systems cannot be built and the notion of "decidability[74]" which eliminates other problems and system blowups.
- **XBRL**: The lighter gray circle with the label "(B)" represents things that are expressible using XBRL currently using global standard approaches. The most important piece of XBRL is what XBRL can do but OWL cannot do. XBRL has two strengths: (1) the ability to articulate information about dimensional relations using the multidimensional model, (2) the ability to articulate mathematical relations. Clearly dimensional relations and mathematical relations are use cases for financial reporting in particular and business reporting generally. XBRL also has the power to express terms and relations between terms, but in this regard OWL is better equipped than XBRL. However, XBRL does have some expressive power here and it also has the architecture which enables richer expression of relations to be created.
- **OWL**: The other lighter gray circle labeled "(C)" represents things that are expressible using OWL 2 DL and Description Logic SROIQ[75]. OWL 2 DL and Description Logic SROIQ are state-of-the-art technologies which are W3C global standards for expressing ontologies. They meet the two needs of "finite" and "decidability" and also make the closed world assumption. These technologies surpass XBRL's current ability to express relations between terms. However, OWL 2 DL and Description Logic have two major limitations: (1) they do not have a dimensional model and (2) they don't support expressing mathematical relations. There is one additional drawback of OWL which is on the one hand a feature, but on the other hand something that is undesirable. OWL 2 DL is so low-level that it has the flexibility to represent any business domain, scientific domain, or any domain for that matter. But this flexibility comes with a price. The price is that OWL 2 DL is so low-level that it is like working in assembly language and therefore it is extremely difficult for even information technology professionals to make use of, and virtually impossible for business professionals to use.
- **Interoperability with other business domains**: The blue circle labeled "(E)" represents other domains which some business domain must interact and interoperate with. Using my example of a financial report which I am creating, other business domains creating ontologies interact with financial

[72] For details, see the document *Knowledge Engineering 101 for Business Professionals*
[73] For details, see the document *Knowledge Engineering 101 for Business Professionals*
[74] For details, see the document *Knowledge Engineering 101 for Business Professionals*
[75] See Understanding Description Logic, http://xbrl.squarespace.com/journal/2015/1/8/understanding-the-importance-of-description-logic.html

reports. For example, the Financial Industry Business Ontology (FIBO)[76] is likely one of those business domains. FIBO is expressed using OWL 2 DL. Public company financial reports filed with the SEC are XBRL-based. These different technical syntax are not a problem, as long as the business meaning, the semantics, are properly synchronized as pointed out earlier in this document. Other global standard technical syntaxes might be used by other business domains. Certainly proprietary formats will also be used internally by reporting entities.

At this point it is worth refreshing our memories of two things: the goal and how to achieve the goal. The goal is the reliable, repeatable, predictable, safe, cost effective, easy to use, robust, scalable, secure when necessary, auditable (track provenance) when necessary and *meaningful exchange of information between business systems*.

The only way a meaningful exchange of information can be achieved is with the *prior existence and agreement on technical syntax rules, business domain semantics rules, workflow/process rules, and the information with which the computer will be working*.

Both XBRL and OWL 2 DL are global standard technical syntaxes. Both will likely progress to be able to serve the needs of business professionals, eventually. But what do we do today? What do we do now? One common denominator for both syntax is finite first-order logic. A partial solution is no solution, it leaves holes which cause problems. These are the complete solution alternatives that I see:

- **XBRL global standard + proprietary**: Using XBRL and supplementing XBRL with proprietary solutions which fill the gap will work. One problem with this is that the OWL-type functionality would need to be recreated in any proprietary solution.
- **OWL 2 DL global standard + proprietary**: Using OWL 2 DL could work, but then you would need to build the multidimensional functionality and the mathematical relations functionality. As I understand it safe SWRL[77] could be used to express mathematical relations. Others say SPIN[78] is a better choice than SWRL. Neither SWRL nor SPIN are W3C recommendations as of yet. The RDF Data Cube Vocabulary[79] could be used to express multidimensional relations. But then, since XBRL is used for actual financial reports, one needs to ultimately serialize information into and likely also read it from XBRL. If this approach is taken, things like open source OWL reasoners[80] can be leveraged.
- **Composite XBRL global standard + OWL 2 DL global standard + proprietary**: It has been suggested before that a composite solution could be built to move things between syntax to leverage both the power of XBRL and the power of OWL 2 DL. That means that any proprietary implementation which fills any gaps that exist would be minimized.

[76] Financial Industry Business Ontology (FIBO), http://www.omg.org/hot-topics/finance.htm
[77] Semantic Web Rules Language (SWRL), http://www.w3.org/Submission/SWRL/
[78] SPARQL Inferencing Notation (SPIN), http://www.w3.org/Submission/spin-overview/ and http://spinrdf.org/
[79] The RDF Data Cube Vocabulary, http://www.w3.org/TR/2014/REC-vocab-data-cube-20140116/
[80] JAVA-based open source OWL reasoner, http://code.google.com/p/owlreasoner/

- **XBRL global standard + XBRL-based proprietary**: Another possible solution is to build proprietary, but read the handwriting on the wall and realize where XBRL has to go next and build an XBRL-based proprietary solution. For example, to provide the expressive semantics that OWL provides in XBRL. This can be done using XBRL definition relations[81]. I created arcroles to express all of the types of relations that I see are necessary for what I need to do to make sure a financial report is created correctly. What if someone implemented a semantic reasoner[82] tailored for XBRL-based digital financial reports or other digital business reports?

It is hard to say exactly what the best implementation approach might be all things considered. This is a decision for information technology professionals. But information technology professionals need to clearly understand the needs of business professionals in order to implement a complete solution.

4.13. Software usable by business professionals

As much as possible proprietary solutions should be avoided in favor of a solution which is based on global standard technical syntax. But that still does not provide business professionals with what they need. Business professionals will want to mainly do things like extend XBRL taxonomies (really they are ontologies). Some business professionals will create big base taxonomies such as the US GAAP XBRL Taxonomy or IFRS XBRL Taxonomy which exist for financial reporting. While those architectures need to be correct and it take more skilled professionals to design an architecture than to simply use an architecture; you will always have professional accountants needing to maintain those taxonomies (ontologies).

But most business professionals will be using taxonomies/ontologies created by other perhaps more highly skilled professionals in the area of knowledge engineering.

Software developers can leverage patterns to make software easier for business professionals to use. Patterns can be combined into composite patterns[83] make working with the technology less like working with low-level assembly language and more like working with Lego blocks.

There are three examples that help you understand what I mean by Lego blocks:

- **Blockly**[84]: This shows the abstract concept of how blocks can be used to work with syntax. Look at the visual, but also note that you can look at the same visual in the JAVA syntax, Python syntax, DART syntax, and XML syntax.
- **Scratch**[85]: This is a tool to help teach elementary school age children about programming. Imagine that the pieces of a financial report or other business report could be put together in this manner.

[81] State-of-the-Art Use of XBRL Definition Relations to Express Business Rules, http://xbrl.squarespace.com/journal/2015/2/18/state-of-the-art-use-of-xbrl-definition-relations-to-express.html

[82] Semantic Reasoner, http://xbrl.squarespace.com/journal/2013/5/28/semantic-reasoner.html

[83] A Vision for Diagrammatic Ontology Engineering, see Patterns on page 5 and Merging patterns on page 7, http://ceur-ws.org/Vol-1299/paper1.pdf

[84] Blockly, https://blockly-demo.appspot.com/static/demos/code/index.html#5ge5sh

[85] Scratch, created by MIT, watch the video in the upper right hand corner of the web page, https://scratch.mit.edu/

- **Quatrix**[86]: This is an application which while does not support XBRL, it works very similar to how I would expect an XBRL-based digital financial report creation tool to work.

There are three technically oriented tools that I have worked with to create OWL ontologies:

- **Protégé**[87]: (free download) This is a free software application which is very hard to use to create ontologies. Business professionals would never be able to use this type of tool.
- **Fluent Editor**[88]: (free download) This tool is a little easier to use because the user can simply create an ontology using a controlled natural language. However, you still need to understand how to create a correct ontology. Again, the typical business professional would never be able to effectively use this tool.
- **Top braid composer**[89]: (free download) This is probably the most complex tool that I have used to create ontologies, much too hard for business professionals to relate to.

Imagine the power of the technically oriented tools, an easy to user interface which hides complexity within well designed software which enables business professionals to only create things correctly. Business professionals would work with things that they understand from their business domain and deal with logic. If things act the way they expect, the logic of what they expect and the logic of what the software does are consistent; business professionals could very successfully make use of semantic technologies.

Too much to ask? I don't think so. Creating something that is complex is easy; anyone can do that. Creating something that is simple is hard work. Creating something simple takes thought, creativity, effort, etc. It takes a skilled craftsman. Such a tool will be elegant. Such tools can, and I believe will, be created.

XBRL International has created the Open Information Model working group[90] to develop a syntax-independent model of a business report. That shows that the understanding that syntax matters less and semantics matters more.

4.14. Understanding the critical importance of decidability

In order to understand critical aspects that make a system work we need to take a brief but important fork in this discussion to make the reader conscious of the notion of decidability. To understand the notion of decidability, we must also discuss the closed world assumption. We do that here.

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

[86] Quantrix videos, watch the Quantrix Key Concepts video, http://www.quantrix.com/en/community/videos/
[87] Protégé, http://protege.stanford.edu/
[88] Fluent Editor, http://xbrl.squarespace.com/journal/2015/1/29/fluent-editor-helps-accountants-see-where-financial-reportin.html
[89] Top braid composer, standard edition, http://www.topquadrant.com/tools/modeling-topbraid-composer-standard-edition/
[90] See https://www.xbrl.org/news/open-information-model-call-for-participation/

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

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

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

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

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

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

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

At the risk of being redundant we point out again the critical distinction between the mechanical aspects of a financial report and the subjective aspects which require or judgmental by a skilled accountant. A conclusion about the correctness or incorrectness of the mechanical aspects in no way suggests or implies that a computer will ever be able to determine the overall appropriateness of a financial report. Such determination always involves professional judgment and therefore always involves a skilled professional accountant.

4.15. Understanding the importance and limitations of first-order logic

First-order logic might seem hard to understand but in reality it is very a straight forward idea. First-order logic is simply an approach or language for describing things and relations between things. Again, different languages have different syntaxes, different levels of expressive power, they are good for some things and not as good as other things. Description logics[91] are a family of representational languages. *SROIQ* Description Logic[92] is one such language which is based on a

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

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

fragment of first-order logic. There are two reasons *SROIQ* Description Logic is important: (a) it is decidable, (b) OWL 2 DL and *SROIQ* Description Logic have consciously equivalent expressive power. Meaning, they were consciously and specifically built to be equivalent for a reason and use the same fragment of first-order logic. While different syntaxes, semantically they are equivalent. While OWL and Description Logic were initially created independently, wise people realized that there are significant advantages to making them equivalent and with specific functionality[93]. The result was OWL 2 DL and *SROIQ* Description Logic which have different syntaxes but equivalent semantics.

Another form of first-order logic is PROLOG which is a programming language[94]. PROLOG a general purpose logic programming language that is also declarative. PROLOG is based on first-order logic. The syntax of PROLOG is derived from Horn clauses which is a subset of first-order logic. Because PROLOG is declarative, program logic is expressed represented by facts, relations, and rules. Questions are asked and then answers are provided based on the facts, relations, and rules.

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

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

A theory is a communications tool. A theory explains, using first-order logic, a theory explains real world things and relations between those things. A theory can be right or wrong, but it is characteristic by its intent: the discovery of essence. The purpose of a theory is to correctly describe the essence of some real world problem domain. Theories can be proven right or wrong. Theories can be tested. Things like OWL 2 DL, *SROIQ* Description Logic, and PROLOG can be used to test and tune theories.

For example, *Financial Report Semantics and Dynamics Theory*[95] is a theory that explains the mechanics of how a financial report works.

4.16. Understanding the importance boundaries

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

[93] *From SHIQ and RDF to OWL: The Making of a Web Ontology Language* helps you understand important ideas and concepts, see http://www.cs.man.ac.uk/~horrocks/Publications/download/2003/HoPH03a.pdf
[94] Understanding the Importance of PROLOG to Digital Financial Reporting, http://xbrl.squarespace.com/journal/2015/7/23/understanding-the-importance-of-prolog-to-digital-financial.html
[95] *Financial Report Semantics and Dynamics Theory*, http://xbrl.squarespace.com/fin-report-sem-dyn-theory/

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

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

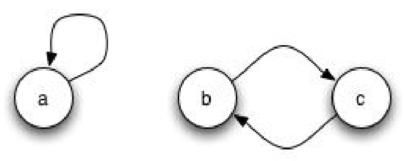
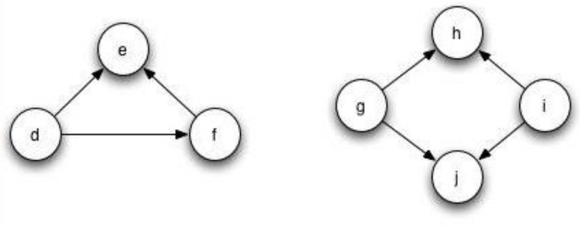
Extensibility is the ability to add things to a system. *Local extensibility* is extensibility that is "inside the walls" of one organization and all extensibility is explicitly coordinated and controlled within and by one organization. For example, a chart of accounts of an organization is an example of local extensibility. You have a framework for adding accounts and you can add whatever accounts you need to the systems.

Distributed extensibility is extensibility that is not explicitly controlled and coordinated by one specific organization but rather using standards-based mechanisms and rules. For example, XBRL-based financial reports submitted to the U.S. SEC is a type of distributed extensibility because while the entire system is controlled by some standard set of rules, each reporting entity has control and can extend the system; but they must stay within a set of rules which coordinates the extensibility.

The point is that one must correctly understand the notion of finite and boundaries. Even an distributed system which is extensible can have solid boundaries if the system is engineered correctly.

4.17. Understanding the problem of recursion

Another catastrophic problem that can occur is when a computer program goes into an infinite loop from which it cannot escape. In network theory[96] there is a relation type called a directed cycle[97] which can cause infinite loops. The following graphics of a directed and undirected cycle helps you understand the potential problems of directed cycles and infinite loops:

Directed cycle	Undirected cycle
	

Directed cycles should be avoided and don't generally exist in many areas of reality. Tools for representing reality should not allow their users to unintentionally create directed cycles. Business professionals should be conscious of the difference between directed and undirected cycles.

[96] Network Theory, https://en.wikipedia.org/wiki/Network_theory

[97] Directed cycle described by the XBRL Technical Specification, http://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html#Directed-cycles

4.18. Choice

There are many choices involved in the process of making digital financial reporting work. Digital financial reports need to be engineered. Computer science is just that; science.

The first choice is whether creating the machine-readable digital financial report is a desirable goal in the first place. Next, issues related to how a digital financial report might work need to be resolved.

Initial use of digital financial reports, mandated public company reporting to the SEC, has not gone perfectly and it can be hard to quantify the relative success or failure of that implementation of digital financial reports. But by all accounts, the following seems to be true:

- Software used to create XBRL-based digital financial reports is hard for public companies to use, there is no way private companies would tolerate that level of usability
- Data quality is not good enough to allow safe and reliable use of XBRL-based financial information which is reported.

This section is not about evangelizing XBRL-based digital financial reporting. Rather, this section explains general aspects of knowledge engineering that should be considered by accounting professionals in order to make digital financial reporting work effectively however one might define "work effectively". If creating and using a digital general purpose financial report is not simple, cost effective, and effective in creating and exchanging financial information of a financial report, then digital financial reporting can never be adopted by the masses.

This section summarizes important general issues, outlines important considerations, and points out the opportunities which could be provided by digital financial reporting if it works as deemed appropriate by the financial reporting supply chain. The essence can be summarized in one concise reality:

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

4.19. Understanding the goal

As Stephen R. Covey pointed out in is seminal work *Seven Habits of Highly Effective People*[98], "Begin with the End in Mind." We begin with the end.

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

The goal is a system that works safely, reliably, predictably, repeatedly, effectively, and efficiently.

Information technology professionals creating software must be able to create software which yields the same result when it would seem obvious to a business

[98] *Seven Habits of Highly Effective People, Habit 2*, http://www.amazon.com/The-Habits-Highly-Effective-People/dp/1455892823

professional using software that the result, such as a query of basic information from a financial report, should be exactly the same even if different software applications are used.

Conscious and skillful execution using this approach can create digital financial reporting which is simple and elegant; and yet a sophisticated and powerful tool. Information in a digital financial reports must be deliberately created to be clear, consistent, logically coherent, and otherwise unambiguous to make sure the guessing game never takes place.

- *Complete solutions* are better than *incomplete solutions*
- *Less expensive solutions* are better than *more expensive solutions*
- *Powerful solutions* are better than *simplistic solutions*
- *Easy to maintain solutions* are better than *hard to maintain solutions*
- *Easy to use solutions* are better than *hard to use solutions*
- *Good solution performance* is better than *poor solution performance*
- *More scalable solutions* are better than *less scalable solutions*
- *Standard solutions* are better than *proprietary solutions*

4.20. Power of agreement

It is only through deliberate, methodical, rigorous and conscious collaboration, cooperation and coordination by the participants of the financial reporting supply chain that XBRL-based digital financial reporting will work safely, reliably, predictably, repeatedly, effectively, and efficiently. That is the goal. This goal will not be achieved by accident.

Consider the definitions of arbitrary and standard:

- **Arbitrary**: based on random choice or personal whim, rather than any reason or system; depending on individual discretion (as of a judge) and not fixed by law

- **Standard**: used or accepted as normal; something established by authority, custom, or general consent as a model or example

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

Agreement is what creates the possibility of enabling machines to perform certain tasks. Business professionals are practical people. If business professionals wanted to have endless theoretical debates they would have become theologians, academics or philosophers. The goal is not to persist the debate; the goal is to agree in order to achieve a specific purpose.

4.21. Basic mechanics of a digital financial report

The basic mechanics of a digital financial report are consistent[99]. XBRL-based public company financial reports filed with the U.S. Securities and Exchange Commission are empirical evidence of this consistency. This consistency is caused by clarity as to these fundamental mechanics articulated by the technical specifications which describe how XBRL works. These mechanics are not open to interpretation.

And yet while 99.9%[100] of these relations are consistent, professional accountants, software vendors, and others do interpret these fundamental mechanics slightly differently.

At the highest level the financial information which is reported is likewise consistent. Overall consistency of a set of 22 basic relations such as "Assets = Liabilities and Equity" is about 98%[101]. Consistency of that specific relation, the accounting equation, is 99.75[102]%. By consistency we mean that every financial report universally follows a specific rule. This does not mean that every report follows exactly the same rules. For example, not every economic entity provides a classified balance sheet; some provide an unclassified balance sheet. But every entity provides either a classified or unclassified balance sheet. Classified and unclassified balance sheets have different rules. Liquidity basis statements of financial position are simply another class of statement.

And so it is the mechanical aspects of a financial report provides the frame of the report and are completely objective and not requiring judgment. What requires judgment is deciding *what* should go into the financial report; what gets disclosed.

4.22. Differentiating objective mechanical aspects from subjective aspects which require professional judgment

Digital financial reports contain thousands and sometimes many thousands of individual pieces or structures[103]. These structures, commonly formatted in machine-readable form using XBRL, are used to represent the information contained in the digital financial report. There are two distinct aspects of these pieces or structures that are important to recognize and be conscious of:

- **objective aspects** which are mechanical and do not require judgment and therefore can be managed using automated machine-based processes.

[99] *Understanding the Basic Mechanics of a Digital Financial Report*, http://www.xbrlsite.com/2015/Library/UnderstandingTheMechanicsOfDigitalFinancialReport.pdf
[100] See *Arriving at Digital Financial Reporting All Stars: Summary Information*, page 4, http://www.xbrlsite.com/2014/Library/AnalysisSummary_ArrivingAtDigitalFinancialReportingAllStars.pdf
[101] See *Summary Information about Conformance with Fundamental Accounting Concept Relations*, http://www.xbrlsite.com/2014/Library/SummaryInformationAboutConformanceWithFundamentalAccountingConceptRelations.pdf
[102] See *Arriving at 2014 Digital Financial Reporting All Stars: Summary*, http://www.xbrlsite.com/2015/Library/AnalysisSummary2014_ArrivingAtDigitalFinancialReportingAllStars.pdf
[103] *Understanding that XBRL-based Digital Financial Reports are made up of Distinct Identifiable Pieces*, http://xbrl.squarespace.com/journal/2015/5/3/understanding-that-xbrl-based-digital-financial-reports-are.html

- **subjective aspects** which require the professional judgment of a skilled accountant, therefore they must be managed by humans.

These objective mechanical aspects are distinct from the subjective aspects which require professional judgment. The mechanical aspects relate to the things and relations between the things in a digital financial report. These mechanical aspects are governed by logic, common sense, and the rules of math. These mechanical aspects are what make up the structure or substrate of a financial report. Everything else fits into this frame or skeleton. This is much like the keystones of a building.

4.23. Representing the financial report problem domain in machine-readable form

A **problem domain**, such as the domain of financial reports, can be broken down into distinct, identifiable elements. This can happen on two different levels: first, on the level of individuals, when we break down this specific financial report into these specific elements unique to that specific financial report. And second, on the level of classes, when we distinguish classes of elements common to all financial reports and therefore universal to all financial reports. Another term for problem domain is area of concern.

Historically, information technology professionals and knowledge engineers have used different terminologies and schemes for describing these identifiable elements of a problem domain (concept maps, UML, UML+ OCL, entity relationship diagrams, semantic data model[104], conceptual model[105], and now what is commonly referred to as the "Semantic Web"). Different schemes often use different terms to refer to exactly the same thing or they use the same term to refer to different things. Different approaches have different expressive power[106] which may, or may not, meet the needs of a business domain. This adds to the confusion of how to best represent real world problem domains in machine-readable form and get the results a business professional expects and desires.

Therefore, we created one common set of terms based on global standard and current state-of-the-art technology. That standard is OWL 2 DL[107] and SROIQ Description Logic which have different technical syntaxes but equivalent semantics.

XBRL should remain consistent with this W3C global standard.

4.24. Machine-readable representations, Taxonomy/Ontology 101

As mentioned, different terms are used to describe a machine-readable representation including taxonomy, ontology, and vocabulary[108]. Although it might

[104] Semantic data model, http://en.wikipedia.org/wiki/Semantic_data_model

[105] Conceptual model, http://en.wikipedia.org/wiki/Conceptual_model

[106] OWL and OCL for Semantic Integration, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.2.7683&rep=rep1&type=pdf

[107] See OWL 2 Web Ontology Language Primer (Second Edition), http://www.w3.org/TR/owl2-primer/#OWL_2_DL_and_OWL_2_Full and the OWL 2 Overview, Semantics section, http://www.w3.org/TR/owl2-overview/#Semantics

[108] Interestingly, the W3C page http://www.w3.org/standards/semanticweb/ontology (notice ontology at the end of the URL) uses the term "Vocabularies".

seem scary, we will standardize on the term ontology and state that an XBRL taxonomy is, and ought to be, an ontology.

An ontology is a salient collection of the classes and subclasses of a problem domain or area of concern. An ontology accurately represents reality. The goal of an ontology is to provide a deliberate, rigorously and methodically worked out, description of the important things and relations between things which is clear, consistent, logically coherent, and unambiguous.

Ontologies identify and describe sets of basic categories of things and universal similarities that these sets have. Universals can be instantiated by more than one object at more than one time. Particulars are non-repeatable and can exist in only one place at any one given time. So, things can be universals or things can be particulars. Relations can exist between universals and particulars:

- **Universal to universal relations**, "is-a" or "is a subtype of" type relations
- **Universal to particular relations**, "has-property" type relations
- **Particular to particular relations**, "part-of" type relations

An ontology should fit the needs of the some specific community, such as a supply chain. Ontologies describe or explain how the collection of things within the problem domain can be represented.

Ontologies have no concern with computational efficiency of a software application.

An ontology should be tractable rather than intractable.

We have distilled the key terminology down to its essence, focusing on terms important to business professionals, information technology professionals, and knowledge engineering professionals who need to communicate in order to articulate information about a problem domain in machine-readable terms. We use the Semantic Web language OWL to capture what we see as the most important elements in the domain of financial reports. OWL is a state-of-the art global standard approach to describing a problem domain.

The following are the key high-level definitions of terms:

- **Thing**: A thing is something that exists in the real world, in the problem domain, in the area of concern. A thing is just a class that all classes and individuals of the problem domain must belong to. All classes are subclasses of thing. Every individual must be of some class. Every class is a thing. Therefore since all classes are subclasses of thing; then all individuals are likewise ultimately a thing. "Nothing" is the opposite of thing.
- **Class**: A class is a set of individuals that have one or more similar distinguishing features in common. Classes are universals. For example *person* is the class consisting of all persons of which *Bill Gates III* is a member. Each problem domain can be captured in terms of a family of classes, together with a set of relations. The most important relation is the subclass relation (also called *is-a*) which organizes the classes in a taxonomic tree. Other key types of relations are *whole-part* and *has-part*.
- **Individual**: An individual is some specific item that exists in reality. Individuals are particulars. For example, a specific person such as *Bill Gates III*, a specific report such as *Fiscal year 2014 financial statement*, a

specific economic entity such as *Microsoft Corp*. An individual exists only once.

- **Property**: (universal to particular, has property) A property is a trait, quality, feature, attribute of an individual, for example the property of *being male* of a person, of *being filed* of a report, and so on.
- **Relations between individuals**: (particular to particular, part of) one individual can be related to another individual, as when *Bill* is *brother-of Dave*, *Bill* is *owner-of the building at 1835 73rd Ave NE, Medina*, and so on.
- **Relations between classes**: (universal to universal, is subtype of) when every member of a certain class stands in a certain relation to some member of another class, then the relation is universal and we can formulate this as a relation between classes. So for example because every brother *is identical to* some male person, we can assert this as a *relation* between the classes *brother* and *male person* to the effect that *brother is-a male person* – in other words the class *brother* is included as a subclass in the class *male person*. If every *financial report* has some *statement* as part, then we can assert *financial report has-part statement*. Relations between classes are universal and apply to every member of that class.

The result of the above rules is a system which always has a single root class at the very top called 'Thing' and a single leaf class at the bottom called 'Nothing'. Thing is the universal class to which all other classes are subclasses must ultimately belong in the financial report domain. All individuals are ultimately members of the universal class. Nothing is an empty class which has no members at all. And so, every such system has Thing at the top, Nothing at the bottom, and business problem domain classes in the middle.

This is a crucial distinction because that resulting organization allows for a conclusion to be reached as to the consistency of some human-readable or machine-readable representation of the problem domain with the description of the problem domain provided by the system. Basically, this system organization is finite rather than infinite.

Having a finite system organization is crucial because if a conclusion cannot be reached as to the consistency of some representation with the description then the system is infinite. Infinite systems are unsafe. Unsafe means that unexpected results, ambiguous results, complexity results which can lead to a machine entering an infinite loop from which it cannot escape could possibly occur.

The fact that the system can be completely described, to the extent of the expressive power of the language of the statements/axioms, by a given set of statements/axioms is provable using formal logic. As such, the finite system a useful tool: it is safe, predictable, reliable, results are repeatable, and no unexpected complexity-caused blowups will occur.

4.25. Representing reality

The ontology uses lower-level terms which fit into the higher-level terms we just described.

The central function of an ontology is to represent reality of the problem domain comprehensively, precisely, and accurately. The quality of an ontology is a function of the comprehensiveness and accuracy of the representation of things and relations between things which make up the problem domain. An ontology is a machine-readable "window" into reality.

There are two approaches to viewing "reality".

One approach is to believe that reality (the world) exists objectively in-and-of itself; reality is independent of any one person. Therefore, reality is knowable; the world exists and its properties are there to be discovered. This view implies that reality is objective and knowable and therefore constraints can exist as to what can be said about reality. In other words, ontologies which provide representations of the world could get things wrong. Therefore, an ontology is right insofar as it accurately reflects the way the world is.

A second approach is to believe that there is no one reality, that every individual perceives the world and that individual perception is reality. This view implies that reality is subjective. This view does not imply that reality is not knowable because there are so many realities that it is impossible to agree on one reality. Rather, it implies that there are "reality camps" or groups of individuals with common beliefs about reality. Therefore, an ontology can represent one "reality camp". Which implies that an ontology can be created for each camp. Therefore, the second approach becomes equivalent to the first approach.

The following terms help one understand the difference between an important nuance and an unimportant negligible difference.

- **Nuance**: a subtle difference in or shade of meaning, expression, or sound; a subtle distinction or variation
- **Subtle**: so delicate or precise as to be difficult to analyze or describe; hard to notice or see; not obvious
- **Negligible**: so small or unimportant as to be not worth considering; insignificant; so small or unimportant or of so little consequence as to warrant little or no attention

Business professionals can best differentiate important nuances from unimportant negligible differences. They do not do it perfectly and the only real way to make sure things are right is testing and experimentation at times.

Ontologies are about getting the salient aspects of a problem domain right. One needs to take a pragmatic view of the world because it is impossible to describe every single aspect of the world. Ontologies only need to represent the important things. An ontology is therefore more like a "wireframe" or a "substrate".

Central to the idea of representing the things in reality is the notion of fidelity. Fidelity means to be a faithful representation or expression of reality relevant to the domain experts who explain the problem domain. Fidelity is the correspondence between or quality of the ontology's representation of the problem domain and the real world.

One final set of terms is important to make clear:

- **Policy**: a course or principle of action adopted or proposed by a government, party, business, or individual; definite course or method of action selected from among alternatives or options and in light of given conditions to guide and determine present and future decisions or choices

- **Requirement**: a thing that is needed or wanted; something that is needed or that must be done
- **Choice**: an act of selecting or making a decision when faced with two or more possibilities or options; the act of choosing; the act of picking or deciding between two or more possibilities or options
- **Option**: a thing that is or may be chosen; the opportunity or ability to choose something or to choose between two or more things

The reason these terms are important is because if options exist and therefore a choice exists, but then a policy is established that no longer allows certain options; then an option can be turned into a requirement.

4.26. Difference between "simple" and "simplistic"

Anyone can create something that is sophisticated and complex. It is much harder to create something that is sophisticated and simple. Simple is not the same thing as simplistic. "Simple" is not about doing simple things. Simple is the ultimate sophistication. Simple is elegant.

Simplicity is "dumbing down" a problem to make the problem easier to solve. That is not what simple is about. Simple is about beating down complexity in order to make something simple and elegant; to make sophisticated things simple to use rather than complex to use.

4.27. Challenges of representing a problem domain

The creation of an ontology is an engineering process, the specialty of knowledge engineers.

An ontology is created about some problem domain, the specialty of domain experts. Financial reporting is a problem domain and professional accountants are experts in that problem domain.

The creation of the machines, the software applications, which leverage the machine-readable ontology is likewise an engineering process, the specialty of software engineers.

It is important to define the term engineering. Engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of something." Building a bridge and engineering a bridge are different things.

Software engineering and knowledge engineering lives in their own little worlds, silos[109]. Professional accountants live in a completely different world.

And so to summarize this situation succinctly: software engineers generally don't understand knowledge engineering; knowledge engineers generally don't understand software engineering; neither software engineers nor knowledge engineers generally understand financial reporting; and professional accountants generally have no idea what knowledge engineering is and are only a little more adept at communicating with software engineers.

[109] *Applications of Ontologies in Software Engineering*, https://km.aifb.kit.edu/ws/swese2006/final/happel_full.pdf

Yet, properly enabled functionality, when properly implemented in software, could provide professional accountants with an ability to automate certain specific mundane tasks.

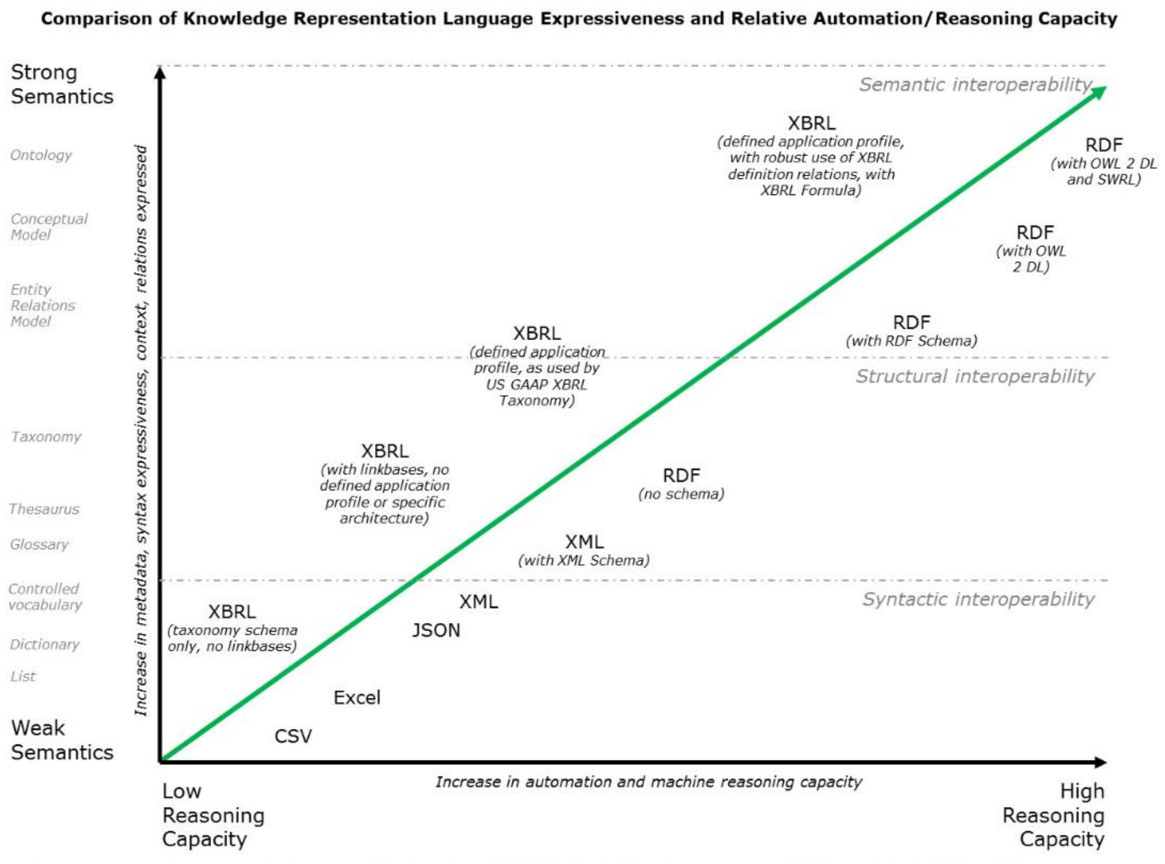
Add to that differences in the interests of participants in the financial reporting supply chain. Professional accountants don't all have the same fundamental interests. Some professional accountants create financial reports. Other professional accountants, financial analysts, analyze the information reported within the financial reports. Other professional accountants work for the FASB and have to create the financial reporting standards necessary for economic entities to report and satisfy the information needs of financial analysts and other users of such information.

Each of these subgroups of professional accountants has a different take on reality because they have different fundamental interests.

Ontologies must be engineered (systematic, disciplined, qualified, etc.) by professionals who communicate effectively. All too often, a software engineer listens to a business professional, takes notes, and then implements what was written down in the notes. That is not engineering. Often, one needs to have the professional skills to "read between the lines" of what a business professional is saying in order to distill the true meaning from what was said.

4.28. Overcoming Limitations of Knowledge Representation Languages

No knowledge representation language is 100% complete. Each has limitations. One must be conscious of such limitations when creating a representation of some problem domain in machine readable form. The graphic below compares knowledge representation language expressive power with the achievable relative level of automation and/or reasoning capacity which can be achieved with that knowledge representation language.



spired by similar comparisons from *An Intrepid Guide to Ontologies* http://www.mkbergman.com/date/2007/05/16/ and *Semantics Overview* http://prezi.com/prwsxj8po3ln/semantics-overview/

Neither XBRL nor OWL 2 DL + SAFE SWRL has 100% of what is necessary to represent 100% of what is necessary for digital financial reporting. Which is the best alternative is unknown at this point in time. The specific gap between the two in terms of expressive power is unknown at this time.

4.29. Pitfalls of knowledge engineering

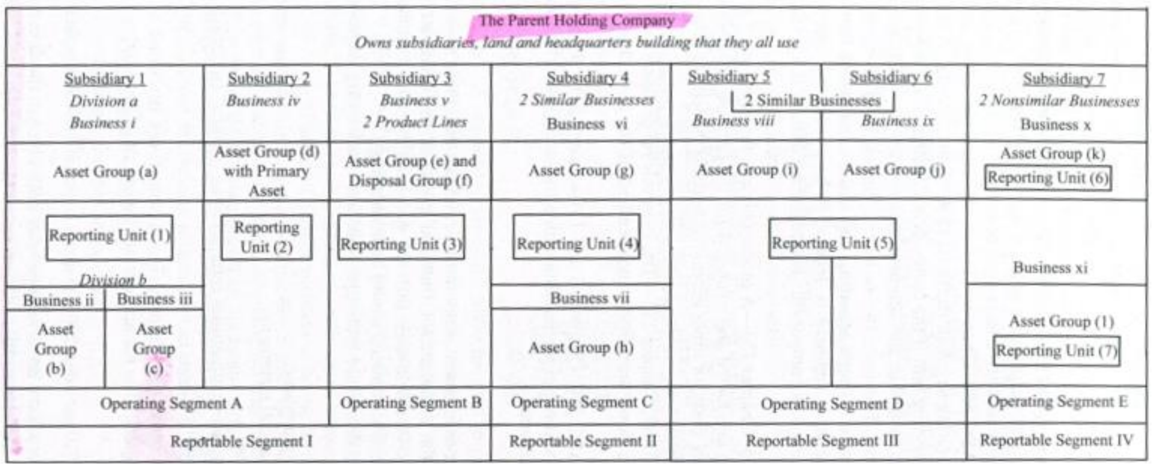
There are many different ways to stumble when trying to represent the knowledge of a problem domain. The following is a summary of many common pitfalls which should be recognized and then avoided.

4.29.1. One rigid reality

Many of the things in a business problem domain are the invention of humans: the foot or meter, currency such as the US Dollar or the Euro, laws, regulations, accounting rules, concept of a legal entity. As such, to a large extent these things that are the creation of humans are malleable. At times there cannot be one single “correct” ontology for things in a problem domain because of inconsistencies in these human inventions. And so it can be the case that there is no single objectively correct answer, but possibly some set of pragmatically-based set of correct answers of some set of groups of users with clearly defined goals but with different sets of interests or self-interest of the specific set or group.

Fundamentally, excessive commitment to reality can lead to and inappropriate level of flexibility or inflexibility.

To make this point clear we use the following example pointed out in the Wiley GAAP 2011, *Interpretations and Applications of Generally Accepted Accounting Principles*, Bragg, page 46:



The segments into which a reporting entity can be broken down are defined inconsistently in the financial reporting literature. From FASB Accounting Standards Codifications, ASC 280 relates to the classification of assets and sometimes liabilities uses the terms operating segments and reportable segments of the business. ASC 350 which relates to impairment uses the term reporting unit. ASC 860 which relates to special-purpose entities and the master glossary uses the term business. ASC 360 which relates to long-lived assets uses the term asset groups and disposal groups. Are all of these different sets of terminology necessary? Perhaps yes, perhaps no.

The following standard terminology is proposed by the Wiley GAAP Guide:

- Consolidated entity
- Parent holding company
- Operating segment (ASC 280)
- Reportable segment (ASC 280)
- Reporting unit (ASC 350)
- Business (ASC 805)
- Asset group (ASC 360)
- Disposal group (ASC 360)

There are two approaches to dealing with this issue: (a) get the FASB to fix the problem or (b) do something to address the symptoms of the problem because the FASB won't or can't address this issue.

Again, note that this is one specific example provided to show that reality is sometimes malleable. At other times reality is less malleable. This specific example is representative of a more general situation.

4.29.2. Overly complicated representation

On the one hand, one must be careful of the illusion of clarity and apparent rigor where, in fact, there is little or no rigor or clarity. These illusions mask problems definitions of things which can be exceedingly difficult and even problematic to correctly characterize or how things interact with one another. Some problem domain things can be untenable regardless if one attempts to articulate the things in machine-readable form. Not recognizing such issues provides a false sense of meaningful information exchange.

Overly complicated representations are spots where the illusion of clarity can hide. Making things obscure by adding unnecessary and perhaps inaccurate details. This also adds to complexity which is simply not necessary.

4.29.3. Blind trust of domain experts

Knowledge engineering calls for careful attention being paid to domain experts characterization of a domain by skilled knowledge engineers. But giving blind trust to domain experts is not appropriate. Knowledge engineers must have a critical side, analyzing and challenging representations for consistency and adequacy. Domain experts are not always right. Blind trust can lead to inappropriate tolerances and otherwise poorly constructed knowledge representations and ultimately an unworkable machine-readable representation.

One of the best ways to overcome this pitfall is to use deliberate and rigorous testing in order to check understanding.

4.29.4. Misuse of highly-expressive languages

Using a highly-expressive language is no guarantee against sloppiness or process deficiencies. Highly-expressive languages offer the power and ability to articulate rich and precise rules for important classes and relations between classes. A weakly-expressive language encourages sloppiness and commonly leads to inaccuracies due to the deficiencies in ability of the weakly-expressive languages to articulate important classes and relations between classes. Where only weak-expressivity is available rich expressiveness is not even available to the knowledge engineer; the result can be a superficial representation which is not useable by the problem domain.

4.30. Recognize that pitfalls are avoidable

Pitfalls are avoidable. Limitations are many times unavoidable and must be worked around. While the real world is malleable and there are always options for representing classes and relations between classes in various ways; this does not mean that everything can be created in any way one pleases. Using one approach in one specific area can mean that options are constrained for some other area of the representation. Dysfunctional, irrational, nonsensical, illogical, inconsistencies, and other issues which cause problems must be discovered and dealt with.

There is a difference between conscious inconsistencies and unconscious inconsistencies. Conscious inconsistencies are generally choices which are made because things are truly different, perhaps only subtle differences or nuances. Unconscious inconsistencies are generally due to sloppiness and lack of attention to detail and cannot be explained which pointed out and questioned.

4.31. Rigorous testing maximizes communication and quality

The best way of assuring that a machine-readable representation is not dysfunctional, irrational, nonsensical, illogical, inconsistent or has some other issue is comprehensive, thorough, deliberate, rigorous testing. Another is examining empirical evidence. Testing is s robust and pragmatic approach to checking understanding and determining if communication has taken place between domain experts, knowledge engineers, and software engineers who ultimately must implement software.

4.32. Representational framework.

A framework which cannot be measured for simplicity is a recipe for unnecessary complexity. Conscientious knowledge engineers are compelled to express a problem domain's classes and relations as richly as possible. With a highly-expressive language at a knowledge engineer's disposal it is possible to think through different representational options at a level of detail that is impossible with a weaker-expressive language. Stronger frameworks push one more than one using a weaker framework. Testing pushes one more than not using testing toward greater accuracy and comprehensiveness. As is said, "Ignorance is bliss." Limitations of expressivity of the representation language used should be exposed so that the limitations become conscious.

4.33. Global standard knowledge engineering framework

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

At a minimum, there will be software vendors and others who desire to convert from OWL 2 DL + SAFE SWRL to XBRL, and from XBRL to OWL 2 DL + SAFE SWRL. No matter what the representation language, the meaning expressed should be equivalent as the "reality" being represented by the domain is the same. It is only the representation language which changes. While different representation languages have different limitations in terms of what can be expressed, what can be expressed should mean the same in each representation language.

Today, neither XBRL nor OWL 2 DL + safe SWRL or safe SPIN can represent 100% of what the domain of financial reporting needs to express. Finite first-order logic can represent this information[110]. Eventually XBRL and/or OWL 2 DL will catch up to the needs of financial reporting. Until then, proprietary approaches likely need to be used. XBRL-based proprietary approaches are the best, all things considered.

4.34. Relations between things are business rules which should be managed by business professionals

As we pointed out earlier, an ontology provides a machine-readable representation of the important things and relations between the things of some problem domain. People refer to these relations in many different ways.

[110] Terminology of a Financial Report, http://www.xbrlsite.com/2015/Library/TerminologyOfFinancialReport.pdf

Some people use the terms "TBox[111]" and "ABox[112]". TBox statements describe the things, the terminology component or the controlled vocabulary of a problem domain. ABox statements describe the relations between the things, the assertions component of the problem domain.

Another term used to describe relations is business rule. The Business Rules Manifesto[113] does a good job of describing what a business rule is. Article 9; Of, By, and For Business People, Not IT People; points out the need for these business rules to be managed by business professionals:

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

Business professionals are the ones who understand the problem domain. As such, business professionals are the ones who understand the business rules or relations between the things in their problem domain.

4.35. One global standard digital financial report or multiple global standards?

Unless someone consciously and explicitly creates one global standard digital financial report specification then there is a risk that multiple digital financial report specifications will exist. While consciously and explicitly creating one global standard digital financial report specification is no guarantee that only one such specification will exist; if no one specification is created it is at least highly likely that multiple specifications will come into existence and those digital financial report specifications may or may not be interoperable. Further, if one global standard specification is not created it opens up the possibility of multiple proprietary standards which are even less likely to be interoperable.

While it is not the end of the world if there are two or perhaps even a few more global standards for digital financial reporting it is the business professional who will ultimately pay the price for unnecessary standards. And this is not to say that if two global standards exist for conscious reasons and with explicit differences in functionality which someone can point to and explain. There is nothing wrong with two global standards if business professionals require two global standards.

What would be a travesty is if there were 10 global standards when 1 global standard would have done and business professionals pay for the inattention which caused that problem to occur with higher priced software.

Imagine a bank trying to implement digital financial reporting in order to reduce the costs of collecting and managing financial information in support of a commercial loan. Say that digital financial reporting was adopted and that for one reason or another 10 different standards for creating a digital financial report existed. Say the bank had 10,000 customers who had loans and who now must submit digital financial reports to the bank. Say the 10 different standards where used equally,

[111] T Box, http://en.wikipedia.org/wiki/Tbox

[112] A Box, http://en.wikipedia.org/wiki/Abox

[113] *Business Rules Manifesto*, http://www.businessrulesgroup.org/brmanifesto.htm

1,000 customers used each of the 10 different formats. How would that work out for the bank which needed to deal with 10 different formats?

4.36. Understanding the taxonomy/ontology life cycle

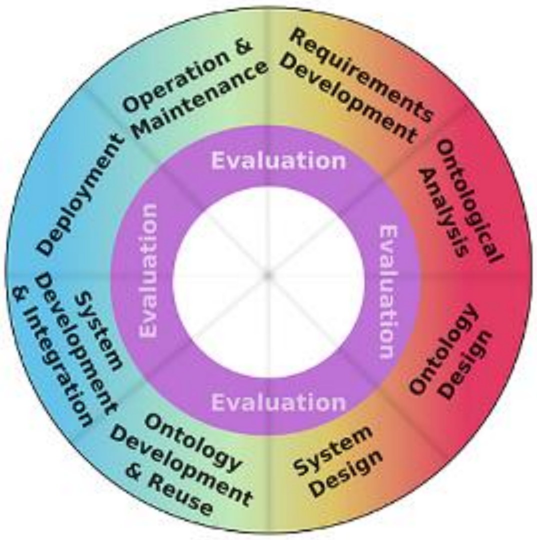
Just like many other things a taxonomy or ontology has a life cycle. The paper *Towards ontology evaluation across the life cycle*[114] explains the problem of not understanding that life cycle and not being able to evaluate the quality of an ontology:

> Problem: Currently, there is no agreed on methodology for development of ontologies, and there is no consensus on how ontologies should be evaluated. Consequently, evaluation techniques and tools are not widely utilized in the development of ontologies. This can lead to ontologies of poor quality and is an obstacle to the successful deployment of ontologies as a technology.

The paper points out that there are five aspects to the quality of ontologies which need to be evaluated:

- intelligibility
- fidelity
- craftsmanship
- fitness
- deployability

The paper provides this diagram of the different stages of the taxonomy/ontology life cycle:



This is a list of the stages which are explained in the document:

- System design
- Ontology design

[114] Towards ontology evaluation across the life cycle, http://www.researchgate.net/publication/260834360_Toward_Ontology_Evaluation_across_the_lifecycle

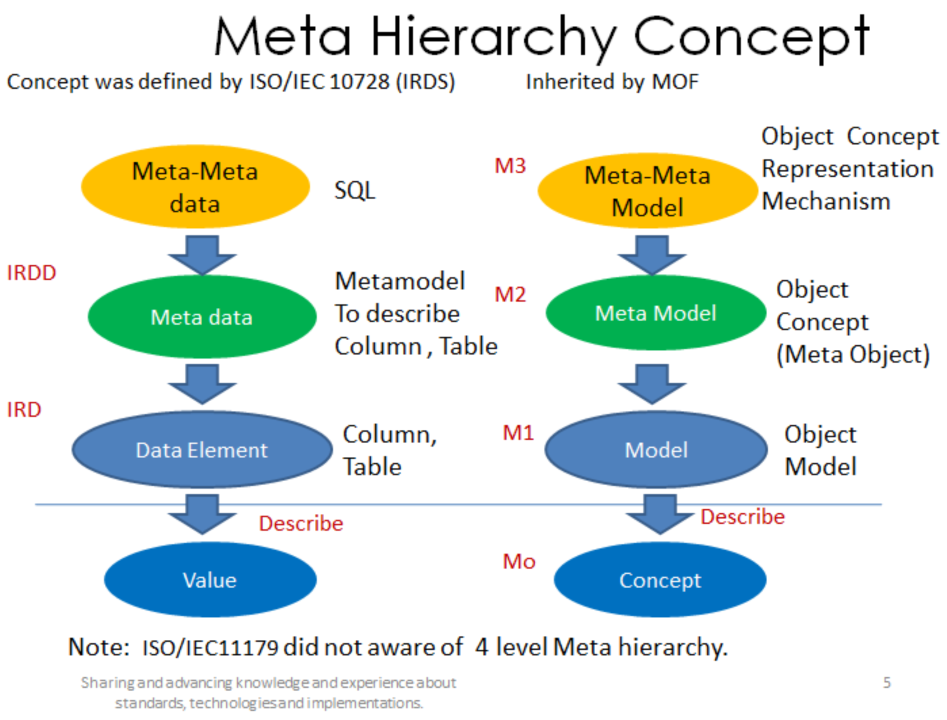
- Ontological analysis
- Requirements definition
- Operations/maintenance
- Deployment
- System development and integration
- Ontology development and reuse

As we mentioned in a previous section, testing helps to maximize communication and therefore quality of an ontology or taxonomy. Evaluation and testing are the same thing.

4.37. Ontology interoperability

Different business domains and even different people in the same domain can create ontologies differently. Yet, many times ontologies need to interoperate. OMG[115] and ISO[116] have created a meta-meta model or hierarchy of concepts which are used to express ontologies. This meta-meta model is intended to maximize interoperability between different ontologies.

This diagram provides an overview of that model.


Meta Hierarchy Concept
Concept was defined by ISO/IEC 10728 (IRDS)
Inherited by MOF
Meta-Meta data
SQL
M3
Meta-Meta Model
Object Concept Representation Mechanism
IRDD
Meta data
Metamodel To describe Column , Table
M2
Meta Model
Object Concept (Meta Object)
IRD
Data Element
Column, Table
M1
Model
Object Model
Describe
Describe
Value
Mo
Concept
Note: ISO/IEC11179 did not aware of 4 level Meta hierarchy.
Sharing and advancing knowledge and experience about standards, technologies and implementations.
5

[115] OMG Meta Object Facility, https://en.wikipedia.org/wiki/Meta-Object_Facility

[116] ISO/IEC/IEEE 42010 Model and Metamodel, http://www.iso-architecture.org/ieee-1471/meta/

The Basic Formal Ontology[117] likewise is intended to maximize interoperability between different ontologies and to promote ontology creation best practices.

4.38. Reading list

The following books are extremely helpful in trying to understand digital financial reporting. We strongly recommend that for anyone who wants to understand digital financial reporting well or who want to build rock-solid products/solutions to read the following books:

Data and Reality[118], by William Kent: (162 pages) While the first and last chapters of this book are the best, the entire book is very useful. The primary message of the Data and Reality book is in the last chapter, Chapter 9: Philosophy. The rest of the book is excellent for anyone creating a taxonomy/ontology and it is good to understand, but what you don't want to do is get discouraged by the detail and then miss the primary point of the book. The goal is not to have endless theoretical/philosophical debates about how things could be. The goal is to create something that works and is useful. A shared view of reality. That enable us to create a common enough shared reality to achieve some working purpose.

Everything is Miscellaneous[119], by David Wenberger: (277 pages) This entire book is useful. This is very easy to read book that has two primary messages: (1) Every classification system has problems. The best thing to do is create a flexible enough classification system to let people classify things how they might want to classify them, usually in ways unanticipated by the creators of the classification system. (2) The big thing is that this book explains the power of metadata. First order of order, second order of order, and third order of order.

Models. Behaving. Badly.[120], by Emanual Derman: (231 pages) The first 100 pages of this book is the most useful. If you read the *Financial Report Semantics and Dynamics Theory*, you got most of what you need to understand from this book. But the book is still worth reading. It explains extremely well how it is generally one person who puts in a ton of work, figures something out, then expresses extremely complex stuff in terms of a very simple model and then thousands or millions of people can understand that otherwise complex phenomenon.

Semantic Web for the Working Ontologist[121], by Dean Allenmang and Jim Hendler: (354 pages) The first to chapters are the most useful. This is an extremely technical book, but the first chapter (only 11 pages) explains the big picture of "smart applications". It also explains the difference between the power of a query language like SQL (relational database) and a graph pattern matching language (like XQuery). Querying can be an order of magnitude more powerful if the information is organized correctly.

---

[117] See, *Ontology for the Twenty First Century: An Introduction with Recommendations*; http://ifomis.uni-saarland.de/bfo/documents/manual.pdf

[118] See, http://xbrl.squarespace.com/journal/2014/7/28/data-and-reality-what-is-the-purpose-of-sec-xbrl-financial-f.html

[119] See, http://xbrl.squarespace.com/journal/2011/1/31/us-gaap-taxonomy-build-it-to-allow-reoganization.html

[120] See, http://xbrl.squarespace.com/journal/2014/7/20/updated-financial-report-semantics-and-dynamics-theory.html

[121] See, http://www.amazon.com/Semantic-Web-Working-Ontologist-Effective/dp/0123735564

Ontology for the Twenty First Century: An Introduction with Recommendations[122], by Andrew D. Spear: (132 pages) The introduction first 45 pages are the best. This chapter is highly influenced by this resource. This can be challenging to make your way through but if you really want to understand all of the issues in creating useful ontologies; reading this is worth the effort.

Systematic Introduction to Expert Systems: Knowledge Representation and Problem Solving Methods[123], by Frank Puppe: (350 pages) The first three chapters of this book are an excellent introduction to expert systems, about 25 pages, and is easily understandable to a business professional. The second section of this book explains how expert systems work and the moving pieces of expert systems. The last to sections get technical, but are still understandable, and provide what amounts to an inventory of problem solving approaches and how to best implement those approaches in software.

[122] See, http://ifomis.uni-saarland.de/bfo/documents/manual.pdf

[123] See, http://xbrl.squarespace.com/journal/2015/7/15/understanding-expert-systems-applicability-to-financial-repo.html