1. Understanding Other Moving Parts of Digital

This section provides additional information foundational to understanding digital financial information. If the ideas in this section are not understood, then trying to understand why digital financial reporting will replace the current financial reporting paradigm is harder to understand. Also, if these ideas are not understood, information technology professionals could make poor architecture choices when implementing digital financial reporting in software products.

The following is a summary of ideas, concepts, and terminology you will need to understand in order to undertake the important journey of understanding model-based semantic-oriented XBRL-based digital financial reporting. This section is intended to help you fill in any gaps in understanding that you might have.

1.1. Interactive data

The SEC coined the term “interactive data”. Most business professionals have used or at least seen a Microsoft Excel pivot table. A pivot table is interactive, or dynamic, in that information you are working with can be pivoted to display information in different configurations.

Just because a reporting entity provides information one way does not mean that you desire to make use of that information using that one presentation of the information. You may want to configure the information differently, you may want to do cross period comparisons of the information reported by an entity, or you may want to do cross reporting entity comparisons of information.

Imagine a financial report which is interactive or has the dynamic characteristics similar to an Excel pivot table. That is what a model-based digital financial report will be like. Digital financial reports can be made interactive, or dynamic, because of the nature of XBRL. You can jump from one place in a report to another because the report is really thousands of individual structures which are understood by software and the software can leverage that structure. You can reorganize the information to suit your preferences, desires, goals, and information needs. You can search, sort, filter, and reconfigure the financial report to suit your preferences.

How does this ability to reorganize a financial statement impact how a financial statement is, or should be, created and how does it impact how the reader of the financial statement interacts with the report? There is a connection between creation and use.

A model-based digital financial report or financial filing is much more like an Excel pivot table than a piece of paper or an electronic piece of paper such as PDF or HTML. As such, professional accountants creating such financial reports may need to look at what they are creating differently, adjusting for the characteristics of this new medium. With the positive characteristics offered by the structured nature of XBRL-based digital financial reports, potentially negative characteristics also show their face and if not properly managed can have undesired affects.

> HINT: Take a look at the video on this web page titled “Quantrix Key Concepts”: http://goo.gl/qQ4Hx This video will help you understand the difference between logical models and semantic models.

1.2. *Unstructured versus structured information*

Simply put, digital information comes in three forms:

- **Unstructured** which means the information contains no identifiable structure and therefore it is unrecognizable and therefore not usable by computer software. Further, no controlled navigation within the pieces of the unstructured information is possible due it its lack of structure.
- **Structured** (or highly-structured) which means the information has identifiable structure which can be recognized and utilized by computer software. Further, because of the structure navigation within the pieces of structured information is possible because of the structure.
- **Semi-structured**[1] information is between structured and unstructured. Semi-structured information does not have sophisticated access structures but accessing information is possible[2].

Structuring information enables computer software applications to leverage that structure and work with the information. Unstructured information has not been organized into a format that makes it easy to access and process that information. Most information has some sort of basic structure.

Structured information, on the other hand, has be organized so that information can be addressed, accessed, and processed by machine-based processes such as computers.

Truth be known, everything that a computer works with has to be structured at some level and the level of structure determines what a computer can do with that digital information. The type of structure determines what you can, and cannot, do with that information.

1.3. *Approaches to structuring information*

People tend to agree that there are three primary formats for representing/modeling highly structured and semi-structured information and that there are different types of databases or other approaches to storing this information:

- **Table-type format (homogeneous, tabular, consistent)**: relational databases, CSV, spreadsheets, or tabular-type representations which allow only one level of hierarchy within each table; but hierarchies can be constructed by relating tables
- **Tree-type format (heterogeneous, arborescent)**: XML, XBRL (using tuples), JSON and other tree-hierarchy-type information which allow for the expression of one hierarchy; hierarchical type databases, object type databases (Note that a tree is a special case of graph. See here to understand the difference)
- **Open-type or Graph-type format (heterogeneous, arborescent)**: RDF, EAV, XBRL (using dimensions) and other open schema-type or graph-type representations which are more graph-oriented or network-oriented (many-

[1] See *Combining Unstructured, Fully Structured and Semi-Structured Information in Semantic Wikis*, http://goo.gl/TwUbs

[2] For more information about semi-structured information see this video, https://www.youtube.com/watch?v=5dk53PTK3g0#t=76

to-many) and allow for dynamically creating virtually any number of hierarchies; supports the notion of cycles; very flexible; network type databases; RDF triple stores.

People tend to agree that data formats are 100% interchangeable. A "tree" can be expressed in the form of a table and put into a relational database. A "graph" can be expressed in the form of a table and likewise be put into a relational database. A "table" can be expressed in the form of a tree or graph. These formats are syntax and any of these syntaxes can be used to store any type of information.

Now performance is important and different structuring formats have different pros and cons. Performance is created in any data format via the use of indexes. Relational databases have indexing capabilities, hierarchical databases have indexing capabilities, and network type databases have indexing capabilities.

People tend to agree that information is more interesting than data. It is not data that people are after; people are after information for the sake of knowledge. Relational databases are popular because of the 'relational' piece, not the 'data' piece. Relational databases are about, in part, organizing sets and relating one set with another set, getting the answers to questions back fast and easy across multiple sets. This is about using information with other information, comparing information. It is not about having a store of data. It is about making use of that data.

1.4. Structured for presentation versus structured for meaning

There are basically two manners or methods or protocols to structuring information digitally:

- **Structured for presentation**. An example of that is a Word processor document which is structured using headings, sub headings, paragraphs, tables and lists. An Excel spreadsheet is also an example of structuring for presentation, it uses worksheets, columns, rows, and cells. Or an HTML document is structured for presentation.
- **Structured for meaning**. An example of that is database or a taxonomy or other type of classification system. A database structures the presentation into rows and columns, but the rows and columns are associated with defined names which are contained in the database schema which have specific meaning.

XBRL structures information for meaning. That structured meaning can be used to help a business user make use of that information.

1.5. Differentiating syntax and semantics

Often confused are the two parts of structured information. Both parts are important, but for different reasons:

- **Syntax** describes the form of the information and is generally not relevant to a business person. This is syntax: `<Name>John Doe</Name>`. Syntax is important to technical people.
- **Semantics** communicates the meaning of the information. For example, “the director’s name is John Doe” communicates meaning as does “the balance sheet balances”. Both are semantics of the information. Business meaning is key to the digital world.

Syntax can be thought of as "how you say something". Semantics can be thought of as "the meaning behind what you said." The following two videos explain and differentiate syntax and semantics:

How XBRL Works: http://www.youtube.com/watch?v=nATJBPOiTxM

This video about semantics: http://www.youtube.com/watch?v=OGg8A2zfWKg

Business professionals need to work with the meaning of information, not the syntax. Software applications build to interact with something like the XBRL technical syntax effectively force business professionals, if they want to user that software, to work with the XBRL technical syntax. If a higher level semantic model is employed to effective mask the technical syntax exposing business professionals to a higher level semantic model, complex things become easier for business professionals.

1.6. Understanding "big data"

People tend to agree that the volume of information is growing rapidly. "Big data" is one of the new buzz words. Ask people what "big data" means and the average business professional probably could not tell you, but they will tell you that they need some because some software vendor says everyone needs big data! I have heard to good definitions of big data. The following are two good definitions that I have come across which explains what is meant by big data:

- Big data is data that is disparately located, varied in structure, voluminous in nature, and rapidly changing.
- Big data is data that is generated by machines. The data is "big" because the machines can generate the data faster than humans can consume the data. Humans really cannot create big data.

I would synthesize those two definitions and some other things that I know into the following explanation of what big data is:

BIG DATA is the notion that you no longer have the luxury of treating one database as 'the database' and putting all of the information you have into that one database. Data that you need exists within your organization and external to your organization. The data varies in the representation format (table, tree, graph). It varies by operating system (Windows, Mac, Linux, etc.). It varies by structure. The volume of information is high and it is getting higher. The velocity which data grows is increasing rapidly. Some of the information changes rapidly. Some of the data is generated by machines faster than humans can consume it. Welcome to the information age!

1.7. Information storage schemes

There are different ways to store information on a computer. You can store information in a file. Another alternative to storing information is to use a database. There are many types of databases. Another term for this is DBMS (database management system) or database model[3]. Now, keep in mind here that you have databases and you have modeling approaches used by databases. These are different things. For example, a relational database can use a multidimensional approach to representing information within that that relational database.

[3] Database model, http://en.wikipedia.org/wiki/Database_model

The following is a summary of database models:

- **RDBMS**: Relational database management system, which is a database based on the relational model or set theory. The relational model is a two dimensional structure: rows and columns. (Note that you can use a multidimensional structure in a relational database.)
- **Hierarchical database**: A hierarchical database management system is a system which follows the hierarchical model[4].
- **Object database**: An object database is a database management system in which information is represented in the form of objects (follows object model), similar in approach to how objects are used in object-oriented programming.
- **Network database**: A network database is a database management system which follows the network model[5].
- **Multidimensional database**: A multidimensional database or multidimensional engine is a system which is fundamentally to work using the multidimensional model[6]. (i.e. this means it is not a relational database which is then structured to mimic the multidimensional model[7], it inherently uses the multidimensional model[8])
- **NoSQL database**: A NoSQL[9] (not only SQL) database provides a system which is based on an open data structure (e.g. tree, graph, key-value, document) which is generally something other than tabular. Basically, a NoSQL database is very flexible and you have to manage the structure yourself[10].
- **Triplestore**: A triplestore[11] or RDF triplestore is a purpose-built database for the storage and retrieval of triples, such as RDF, which is a graph of subject-predicate-object relations.
- **Flat file database**: A flat file database[12] is a system where in essence one or more files are used to store data.
- **Graph database**: A graph database[13] uses the mathematics notion of a graph or directed graph to implement a database model. Linked data is basically seeing the entire internet as a database. So the "system" is the internet itself.

[4] Hierarchical database model, http://en.wikipedia.org/wiki/Hierarchical_database_model

[5] Network model, http://en.wikipedia.org/wiki/Network_model

[6] Multidimensional database model, http://web.stanford.edu/dept/itss/docs/oracle/10g/olap.101/b10333/multimodel.htm

[7] Multidimensional or relational? What's the right system for you?, http://quartetfs.com/blog/multidimensional-or-relational-whats-the-right-system-for-you/

[8] The Rebirth Of Multidimensional Analytics, https://www.youtube.com/watch?v=AjrByTsbzdg

[9] NoSQL, http://en.wikipedia.org/wiki/NoSQL

[10] What is a NoSQL database?, https://www.youtube.com/watch?v=pHAItWE7QMU

[11] Triplestore, http://en.wikipedia.org/wiki/Triplestore

[12] Flat file database, http://en.wikipedia.org/wiki/Flat_file_database

[13] Graph, http://en.wikipedia.org/wiki/Graph_(abstract_data_type)

Relational databases are popular information storage schemes and are very mature. NoSQL, network, graph, and RDF triplestores are gaining in popularity because they overcome many of the limitations of relational databases.

People tend to agree that relational databases are a very mature, tested, stable, well understood, popular, robust, sophisticated tools. There are a lot of people who understand how to administer relational databases, develop relational databases, maintain relational databases, etc. There are a lot of incredibly useful features which relational databases have such as fault tolerance, commit-rollback, replication, etc. However, relational databases do have their weaknesses. No tool can do everything.

1.8. Information retrieval or query schemes

One you put information into a database of some sort, the next thing you want to do is get the information out of the database effectively and efficiently. There are many different information retrieval or query schemes or query languages[14]:

- **SQL**: Structured Query Language or SQL[15], a global standard query and functional programming language used by relational databases. SQL queries table-type data.
- **XQuery**: XQuery[16] is a global standard query and functional programming language that is designed to query and transform collections of structured and unstructured data, usually in the form of XML or text. XQuery queries tree-type data.
- **JSONiq**: JSONiq[17] is an open, third-party extension of XQuery which, among other things, expands XQuery to be used with JSON formatted information.
- **SPARQL**: SPARQL[18] Protocol and RDF Query Language is a global standard RDF query language, it is used to query graph-type or open-type data formatted in RDF.
- **MDX**: Multidimensional eXpressions or MDX[19], a query language for OLAP databases. This was first developed by Microsoft but it seems to be sort of a standard, it does seem to be popular.

People tend to agree that SQL is a fantastic query tool. Personally I love SQL. And, because we said above that any information can be expressed using any data format, clearly we can express any kind of information within a relational database. You can create foreign keys and relate any relational database table to any other relational database table. No problem there. Information technology professionals can do that for you or you can do that yourself.

However, the queries you have to create to get information out of that relational database get increasingly complex with more and more of these types of relations. Further, the database schema becomes increasingly hard to understand because how the information is structured is really not that intuitive.

[14] Query languages, http://en.wikipedia.org/wiki/Query_language
[15] SQL, http://en.wikipedia.org/wiki/SQL
[16] XQuery, http://en.wikipedia.org/wiki/XQuery
[17] JSONiq, http://en.wikipedia.org/wiki/JSONiq
[18] SPARQL, http://en.wikipedia.org/wiki/SPARQL
[19] MDX, http://en.wikipedia.org/wiki/MultiDimensional_eXpressions

Some people believe that you can convert SQL and SPARQL queries to XQuery[20]. I don't totally grasp this presentation. Theoretically, it makes sense that this is possible because information can be stored and transferred between any database format.

People don't tend to realize that trying to make a relational database do things that it was not really built do causes complexity which can make things harder to understand, harder to create, and harder to maintain. Basically, it is best to use the right tool for the right job.

1.9. Balancing a system, arriving at equilibrium

Life is a tradeoff. It is rare that something has only positive characteristics and something else has only negative characteristics. You get to choose a “basket” of characteristics. You pick the basket with the best set of characteristics which meets your needs and requirements. Maybe business professionals don't care about some characteristics of these things but someone who pays the bills cares about all of these. Business professionals need to understand their real needs and requirements so that the proper balance can be achieve. This decision process generally involves information technology professionals. Business professionals need to understand the tradeoffs so that proper discussions can take place between business professionals and information technology professionals. The correct equilibrium or balance must be achieved. All these need to be in balance:

- **Easy for business professional to use (intuitive)**: Something should be EASY to use as opposed to HARD to use.
- **Query power and query sophistication**: Queries should be POWERFUL rather than UNSOPHISTICATED. (The more you can do, the better, as long as what you can do is useful to you.)
- **Performance, query speed**: Performance should be FAST rather than SLOW.
- **Expressive power**: The expressiveness of the system should be EXPRESSIVE as compared to INEXPRESSIVE. (The more you can do, the better, as long as what you can do is useful to you.)
- **System flexibility, agility**: A system should be FLEXIBLE as compared to INFLEXIBLE. (Flexibility should be judged by where the user needs the flexibility. Flexibility in the wrong places causes a system to be harder to use than necessary. Unnecessary options are a bug, not a feature.)
- **System scalability**: A system might need to SCALE as compared to DOES NOT SCALE.
- **Global standard**: A system might be better if it is more STANDARD than PROPRIETARY.
- **Cost effective**: A system could either EXPENSIVE or INEXPENSIVE.
- **Maintainability**: A system could be either HARD TO MAINTAIN or EASY TO MAINTAIN.

[20] Translating SPARQL and SQL to XQuerySee, http://archive.xmlprague.cz/2011/presentations/sparql-sql-xquery.pdf

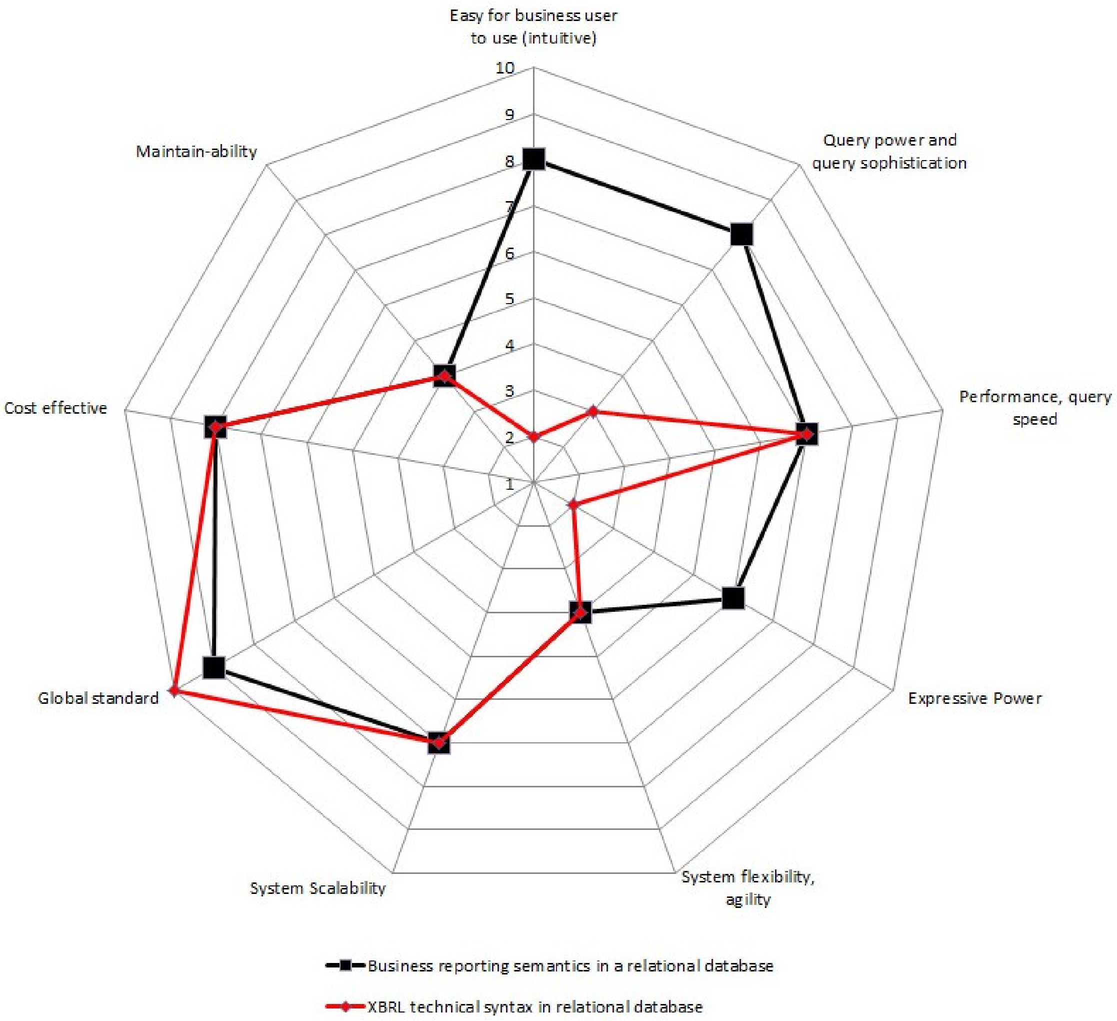
The following shows two radar charts which compare different implementation alternatives for storing XBRL-based information[21].

The first radar chart shows information related to storing the XBRL technical syntax in a relational database.

The second radar chart shows information related to storing not the XBRL technical syntax, but rather the meaning of information within a NoSQL database.

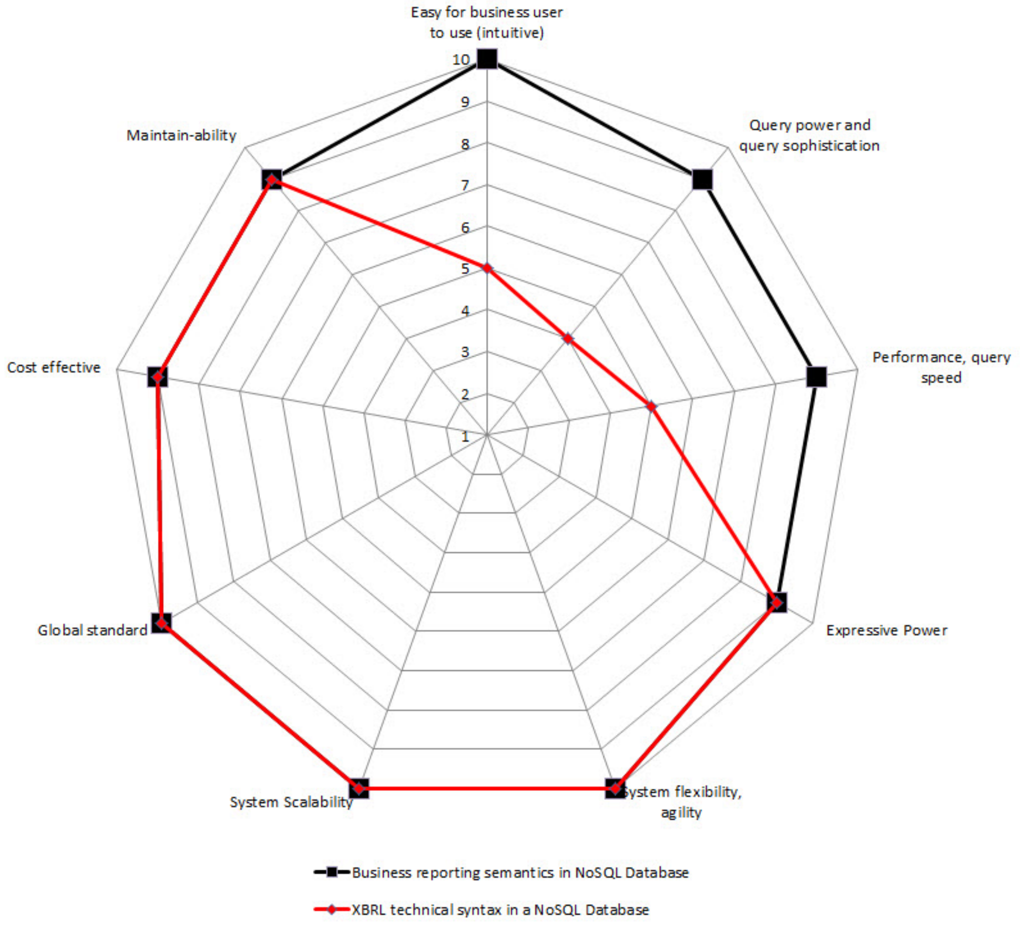
Data about the two implementation alternatives is not the focus here, rather the process of comparing different alternatives is the focus. Using testing and benchmarking anyone can accumulate their own information about alternatives which they see.

Comparison of requirements when XBRL technical syntax (red) and business report meaning (black) is stored in a relational database:



[21] Comparing XBRL Implementation Alternatives, http://xbrl.squarespace.com/journal/2014/6/14/comparing-xbrl-implementation-alternatives.html

Comparison of requirements when XBRL technical syntax (red) and business report meaning (black) is stored in a NoSQL database:



Business professionals need to understand certain aspects of how information technology works in order to make the best decisions and pick the best alternatives, all things considered. That is the point. The comparison above is our observations of the empirical evidence. Gather your own evidence.

1.10. Business information is inherently dimensional

Business information, and particularly financial information, is inherently multidimensional. To understand what dimensional or multidimensional means and to understand why this is important, consider the following brief explanation:

- A **value** such as the numeric value for *pi* is a **scalar**. The value of *pi* which is 3.14 is the same, no matter where it is used. Scalars have no dimensions or other characteristics, they stand alone.
- A **list** can be thought of as having one dimension. Dimensions are a model for expressing characteristics of information. Dimensions effectively contextualized for unambiguous interpretation. For example, the name of a company and its state of incorporation can be thought of as a list.

- A **table** can be thought of as having two dimensions; one dimension represented by the columns of the table, the other by the rows of a table. Other terms used for table are matrix and array.
- A **cube** can be thought of as a three dimensional matrix/array. For example, think of the "x", the "y" and the "z" axis of a three dimensional chart you may have worked with.
- A **hypercube** is an "*n*-dimensional" matrix/array, meaning that it can have from one to any number of dimensions. Hypercubes can be hard to articulate in two dimensions, such as paper. But computers are good at working with hypercubes. You can think of a pivot table data as a hypercube.

The fundamental building block of the multidimensional model is the hypercube. A hypercube is a set of dimensions used to represent information.

Walking through this in another way, consider the number 1,000. What does that number mean? What if we told you that the number related to Cash and Cash Equivalents for the current fiscal period of December 31, 2010, reported by the consolidated entity which has the SEC CIK number 0123456789 whose value is $1,000,000 reported in thousands of US Dollars. Each of those descriptive characteristics of the number 1,000 is a different dimension of that number.

In order for financial information to be usable the information must be unambiguous to be interpreted appropriately.

The multidimensional model is simply a logical model for organizing information. The multidimensional model is flexible in that it does not specify presentation information related to the information expressed by the model. Presentation of that information is a different problem than unambiguously expressing the information. Users of the model are free to present the information as they deem appropriate, leveraging the dimensional information or other helpful information. What the multidimensional model does provide is enough agreement to express information so that it can be unambiguously understood by a computer software application, including applications which can render the financial information in a format appropriate for human consumption.

1.11. Role of software

Complexity can never be removed from a process but it can be moved. Software can assume the complexity of things like the XBRL technical syntax by leveraging things like a logical model or a semantic model. Software can leverage ideas such as the multidimensional model in pursuit of that task.

Software can turn the complex physical implementation of technology into a significantly easier to use logical model and/or semantic model; hiding and taking care of the complexity of the technology for the user in the background. Most software today which tries to help business professionals make us of XBRL is still maturing and does not leverage a logical model or semantic model; therefore they have to work at the level of the XBRL technical syntax. Software will mature and move to a more semantic approach, hiding the technical syntax from business professionals.

1.12. Understanding the multidimensional model

The multidimensional model is a model used to represent information. Other popular models for representing information include the relational model and hierarchical model. There are other models. Each models has its strengths and weaknesses, it pros and cons.

Multidimensional views of information provide what many people refer to as the ability to “slice and dice” information. Another way of stating this is that the multidimensional model provides flexible access to information.

People often confuse the multidimensional model with OLAP (online analytical processing), BI (business intelligence) and other such implementations of the multidimensional model.

Transaction processing systems such as accounting systems tend to use the relational model or a relational database management system (RDBMS).

Data warehouses or sometimes called data marts is an approach to creating an enterprise wide data store. A data warehouse basically helps tie transaction processing systems together so the data can be access as if it were one set. Business intelligence systems are used to report information to those who use that information. But data warehouses and business intelligence software tends to be focused on the internal use of information within one organization. Much information which one might use can be external to an organization.

As we said, each of these models has its pros (strengths) and cons (weaknesses); each has different needs. Business information comes from these different systems and goes into these different systems.

Yet there is no one standard multidimensional model used by all systems which use that model. The relational model has SQL (structured query language) and ODBC (open database connectivity). Connecting systems which use the multidimensional model can be more challenging. The white paper *Getting Started with ADAPT™, OLAP Database Design*[22] discusses these issues.

This section helps sheds light on why the multidimensional model is used, it separates the multidimensional model, OLAP, BI, and XBRL Dimensions.

1.12.1. Strength of the multidimensional model

The greatest strength of the multidimensional model is the flexibility it provides to slice and dice and otherwise reformat information to fit the preference of the consumer of the information. Relational databases can be made to express information using a multidimensional type of an approach using fact tables, star schemas to mimic the multidimensional model, but a multidimensional database is optimized for the multidimensional model.

1.12.2. Strength of the OLAP

OLAP (On-Line Analytical Processing[23]) is an approach to swiftly answer a query.

[22] Getting Started with ADAPT™, OLAP Database Design, http://www.symcorp.com/downloads/ADAPT_white_paper.pdf

[23] OLAP Council Whitepaper, http://www.olapcouncil.org/research/whtpaply.htm

OLAP and the multidimensional model are two different things. OLAP uses the multidimensional model to achieve its goals. OLAP tends to focus on numbers only, is optimized to enable the aggregation of information. Also, OLAP sometimes even pre-aggregates numbers to make queries faster. Further, OLAP is for providing information, it is not generally "read-write". OLAP tends to be less useful with reporting textual type information and in situations where you do not want aggregation.

OLAP tends to be internally focused within an entity and not that adept at working with information which is external to an entity.

You can think of OLAP as if it were a three dimensional spreadsheet (or more precisely an "*N*" dimensional spreadsheet meaning any number of dimensions). This is called an OLAP cube. An Excel pivot table is a very basic example of an OLAP cube[24].

1.12.3. Business intelligence systems

Business intelligence[25] (BI) is a type of decision support system which transforms and organized raw information and transforms that information so that it can be used to make business decisions. BI systems are organized to present information in such a way as to guide a business toward some desired goal.

BI systems tend to use OLAP and therefore likewise tend to use the multidimensional model. BI systems are implemented within software.

BI systems have pros and cons:

- There is no one global standard BI system or one standard multidimensional model used by BI systems. As such, BI systems are not generally interoperable. They can be made to interoperate, but they are not inherently interoperable. BI systems tend to work well with the internal information of an enterprise, but less well with information external to an enterprise.
- BI systems generally use OLAP. And as such they have the strengths and limitations of OLAP. As such, BI systems tend to work best with numbers and tend to force you to aggregate numbers.
- BI systems tend to be read only, you can use information from a BI system but you cannot put information into a BI system. Generally, BI information is put into a transaction processing system which then goes into a data warehouse which the BI system then uses.
- BI systems focus on numbers and work with numbers extremely well; however they work less well with textual type information or narratives.
- BI systems don't tend to allow you to import schemas or other metadata which is used to work with the information, the tools tend to provide you mechanisms within the tools to create this metadata.

Two of these limitations are critically important when it comes to XBRL. The first is that BI applications tend to focus more on numbers, rather than text and numbers and therefore BI systems are limited in working with XBRL information which can contain both numbers and text. The second is that BI systems tend to focus on

[24] OLAP Cube, http://en.wikipedia.org/wiki/OLAP_cube

[25] Business Intelligence, http://en.wikipedia.org/wiki/Business_intelligence

numbers and like to help you aggregate those numbers because that is what OLAP does and in XBRL reports you don't want aggregation many times.

For example, if you ever tried to use an Excel pivot table which is basically a simple BI-type tool, you can see how a pivot table cannot quite do what you want to do in terms of rendering financial reporting information which has been expressed in XBRL.

A third important thing to realize is that BI system don't tend to provide easy ways to import metadata such as the information which is contained within an XBRL taxonomy which provides the schema for information contained within an XBRL instance.

BI systems are quite useful, but they need to go to the next level. Currently, BI systems seem to be focused on internal analytics within an organization or many times within a department of an organization which cannot work with the internal analytics of systems within the same organization. BI needs to be more externally oriented, bringing in information from whatever source, from whatever entity, internal or external.

1.12.4. Model based reporting and the multidimensional model

Model-based reporting is catching on in the financial reporting space. Enterprise software vendors such as IBM (IBM Cognos Financial Statement Reporting (FSR) External Reporting), SAP (SAP BusinessObjects Disclosure Management), and Oracle (Oracle Hyperion Disclosure Management) have model-based reporting software applications which support the creation of financial statements. Financial reporting can be seen as leading the way in model-based reporting.

But many other software companies are jumping into the model-based financial reporting arena.

Two companies which I will mention here are Quatrix and A3 Modeling because they have great videos which help understand what model-based financial reporting looks like. Here are those videos:

Quantrix Modeler: http://www.quantrix.com/tour/Concepts2.htm

A3 Modeling: http://a3solutions.com/advantages-of-a3-modeling/

Although, many of these model-based financial reporting solutions are tied too tightly to OLAP which means they are focused on numbers and not both numbers and textual information such as narratives found in financial reports.

1.12.5. Reconciling multidimensional terminology

The multidimensional model terminology associate with it. Unfortunately, there is not one standard, precise set of terms that everyone agrees on. But most models are fairly close. Symmetry Corp, a business intelligence consulting firm, has created a common model that it uses to reconcile all the different multidimensional model terminology used by the major software vendors they support. You can see this reconciliation here:

http://www.symcorp.com/downloads/ADAPT_white_paper.pdf

XBRL Dimensions terminology is yet another variation of multidimensional terminology. The US GAAP Taxonomy uses yet another set of terms in an attempt to make the multidimensional model easier for business professionals to make use of. The table below provides a reconciliation between this terminology:

Common BI or Multidimensional Model Term	**XBRL Dimensions Term**	**US GAAP Term**	**Description**
Scalar			Data that has no dimensions. For example, the value for pi (3.14) has no dimensions.
Cube, data cube, hypercube, pivot table, array, matrix, info cube	Hypercube	[Table]	Connection between a set of dimensions.
Dimension, characteristic, measure, axis	Dimension	[Axis]	A characteristic of the information. For example, "Geographic Area" may be a characteristic of the information and therefore a dimension.
Domain	Domain	[Domain]	Set of members of a dimension.
Member	Member	[Member]	A possible values of a dimension. For example, "Asia", "Europe", "North America", "South America" might be members of the "Geographic Area" dimension.
Measure	Primary item	[Line Items], Concept	Generally, in XBRL terms, the XBRL taxonomy concept dimension of information. For example the taxonomy concept "Sales" may be a primary item. NOTE: In BI, concepts are simply another dimension.
	Network	[Network]	Hypercubes exist within XBRL networks. A network may have one or more hypercubes within it. Networks are a way of physically separating sets of relations.
Navigational attribute, Flow		Number and category of network	Order or sequence of hypercubes
Fact, key figure	Fact	Fact	A fact is reported piece of information which could be numeric, non-numeric (i.e. strings), or narrative (i.e. TextBlock).
Fact table			Set of facts associated with a hypercube
Slice			A portion of a hypercube, somewhat like a filter, which allows information with more than two dimensions to be presented on a two-dimensional surface.
Formatting information, display attributes		Presentation relations	Information related to formatting, presenting, and/or rendering information from a hypercube.

If you are confused as to what a term means, the table above can be helpful in figuring out the definition of the term.

1.13. Problems with OLAP

OLAP, like any other tool, is not perfect. While some point out micro-level issues with OLAP[26], there are also macro level issues with OLAP. Here is a summary of issues with OLAP[27] that I have accumulated trying to understand and use OLAP tools to make use of XBRL:

- There is no global standard for OLAP
- Cube rigidity
- Limited computation support, mainly supports only roll ups
- Limited business rule support and inability to exchange business rules between implementations
- Inability to transfer cubes between systems, each system is a "silo" which cannot communicate with other silos
- Inability to articulate metadata which can be shared between OLAP systems

[26] The Problems with OLAP, http://www.information-management.com/issues/20070301/1076555-1.html

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

- Focus on numeric-type information and inconsistent support for text data types
- OLAP systems tend to be internally focused within an organization and do not work well externally, for example across a supply chain
- OLAP tends to be read only

1.14. Problems with electronic spreadsheets

While electronic spreadsheets are wonderful tools, electronic spreadsheets are not perfect tools. People point out the flaws of the electronic spreadsheet including[28]:

- Vulnerable to fraud
- Susceptible to trivial human errors
- Difficult to troubleshoot or test
- Obstructive to regulatory compliance
- Unfit for agile business practices
- Not designed for collaborative work
- Hard to consolidate
- Incapable of supporting quick decision making
- Unsuited for business continuity
- Scales poorly

Business professionals tend to love spreadsheets, information technology departments tend to loath electronic spreadsheets for the problems they cause.

1.15. Sweet-spot of XBRL-based business system to business system information exchange

When we think of financial reporting on usually thinks of word processor documents or electronic spreadsheets exchanged between business professionals. But financial reporting is actually much broader in scope than these work processor documents and electronic spreadsheets.

Many times the word processor documents or electronic spreadsheets end up being “cut and pasted” into other documents, spreadsheets, or systems. One case in point is how information from a financial statement is many times put into the system of a bank, regulator, or analyst to reuse that financial information in some manner many times over many years.

For contrast, look at the other end of the spectrum and what many people refer to as transactions. Be these accounting transactions or operating system transactions, transactions tend to be smaller in nature, while the information within the transaction may change, the form of the transaction generally does not change. While transactions are not considered model-based digital financial reports, the

[28] Top 10 Disadvantages of Spreadsheets, http://www.denizon.com/spreadsheets/top-10-disadvantages-of-spreadsheets/

difference between these two offer an opportunity to understand the difference between the two.

Model-based digital financial reporting allows for formal agreement and therefore the opportunity to automate financial information exchanges of many types. While this approach is not generally appropriate for high volume, small, unchanging transactions; it does offer an opportunity to automate a number of information exchanges used within a business. The "sweet spot" of model-based digital financial reporting can be articulated as:

- **Larger transactions** which tend to change (i.e. such as a 50 or 100 page regulatory report with perhaps thousands of facts exchanged, as opposed to a small transaction with 10 data points)
- **Ad hoc exchanges** which seem to appear, all one needs to do is look at the electronic spreadsheets which you exchange today.
- **Business people changing the metadata**, no information technology department involvement required.
- **Information which needs to be reconfigured**, rather than a "form" (i.e. financial reports are not a form)
- **Zero (or low) tolerance for errors** in the information (i.e. everything must tick and tie and if things don't add up, bad things happen)
- **Business report focused** exchanges means that XBRL does not have to represent everything, it focuses on fact-based information exchanges and is therefore easier to use because of the higher-level a business professional has to work with[29]

While other technical syntaxes are inflexible or too flexible and therefore too hard to use, XBRL offers a unique mix of characteristics which is balanced for business professionals to use.

1.16. Understanding the semantic spreadsheet

Imagine an improved electronic spreadsheet, a semantic spreadsheet[30] which overcomes many of the problems of how spreadsheets work today. Imagine an improved OLAP, or NOLAP (not only OLAP)[31]; where the spreadsheet is inherently a dynamic pivot table. Imagine a new take on spreadsheets[32]. Imagine an end to what is called "spreadsheet hell"[33].

This is my take on what is wrong with current electronic spreadsheets is this list of 5 fundamental problems that I see and how to fix those problems:

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

[30] See *Semantic spreadsheets*, http://xbrl.squarespace.com/journal/2013/4/18/semantic-spreadsheets.html

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

[32] See *Time for a New Take on the Electronic Spreadsheet*, http://xbrl.squarespace.com/journal/2013/8/2/time-for-a-new-take-on-the-electronic-spreadsheet.html

[33] See *XBRL Ends Spreadsheet Hell*, http://xbrl.squarespace.com/journal/2009/5/2/xbrl-ends-spreadsheet-hell.html

1. **Information is presentation oriented rather than meaning oriented**: Today's electronic spreadsheets, all of them, are made up of sheets which contain rows and columns which intersect to form cells. Information is entered into cells. All these rows, columns, and cells are presentation oriented. What if the information was meaning oriented instead? What is spreadsheet information was glued together by the meaning of the information?

2. **Business rules combined with spreadsheet information**: Spreadsheets today have the data within the spreadsheet combined with the business rules such as formulas for how information adds up, tests to make sure there are no errors, and other information mixed within the data of the spreadsheet. This can make it very hard to check a spreadsheet for errors or missing business rules. To look at this another way, imagine a spreadsheet which is verified using an *external set of business rules*. Sometimes the business rules could be publically available, other times the business rules would be securely available to a select group of users of the spreadsheet. The basic premise is that you can separate the business rules used to check the spreadsheet from the actual information which provides more control over both the business rules and the information. Plus, this means that the same set of business rules can be used across multiple spreadsheets to verify that the spreadsheets do not contain errors. Considering #1 above, the information, the business rules, and how the information is presented all really need to be separated to make the spreadsheet more flexible. So, what if business rules could be external to the spreadsheet?

3. **Multiple copies of the same spreadsheet**: A big problem is multiple versions of the same spreadsheet and you lose track of which version is the correct version to be using. Many people refer to this issue as spreadsheet hell. More and more people are addressing this by storing spreadsheet information in a database and exposing the information view Excel, but saving the information into a database. The problem with this is see #1 above, the information stored is still presentation oriented and not meaning oriented. What if you addressed information by the meaning of the information, the characteristics of the information is how you identify the information?[34]

4. **Comparing information between spreadsheets can be a challenge**: If you have ever given a spreadsheet to two or more different people, had each person put information into the spreadsheet, and then tried to compare spreadsheet information you understand this situation. Reusing information contained in spreadsheets effectively can be a big challenge. What if you could compare meaning?

5. **Proprietary format, forced to use one software application**: Excel is a great software application for working with spreadsheets. But if you don't have Excel or someone you want to share information with does not have Excel and you want to exchange information, this can be problematic. The interoperability between Excel, Google Spreadsheets, and Apple Numbers spreadsheets is OK some times, but other times problematic. Standard formats such as Open Documents helps, but the standards focus on formatting of information, not the semantics of the information. Also,

[34] One software vendor calls this a "cell store", I believe a better term is a "fact store".

> business rules are still embedded within the application. Further, Excel is a very "heavy" client. With tablet PCs and mobile devices growing in popularity, that becomes more and more of a problem. What if a spreadsheet was a global standard format, rather than a proprietary format of one software vendor?

Here are my requirements for a better spreadsheet[35]. This new improved version of a spreadsheet is not intended to replace 100% of all existing spreadsheets. Rather, this is intended to be a new alternative, a new category of spreadsheet. An alternative which could be used in 20% of the cases where more control is needed over spreadsheets (but I suspect the spreadsheet would be use in 80% of cases).

1. **Readable by both humans and machines**: A spreadsheet should be readable by both humans and machines. Information provided within a spreadsheet should be more a representation of information than presentation oriented. The representation can be presented in sheets, rows, columns, and cells but this is done leveraging information metadata and commonly understood patterns. 100% pixel perfect renderings are specifically not a requirement.

2. **Global standard format**: The format of the spreadsheet should be a global standard, not controlled by one software vendor.

3. **Agreed upon level of semantics**: The creators and consumers can agree on the level of semantic clarity they will make use of for a spreadsheet. The spectrum can range from no semantics at all (which is similar to today's spreadsheet) or a high level of semantics expressed by a highly controlled representation model.

4. **Separation of representation and presentation**: The "representation" and the "presentation of the representation" should not be intermingled.

5. **Business rules separable from spreadsheet**: Business rules should be separated from the information when desired, integrated with the spreadsheet when necessary. Business rules which are external to the spreadsheet can be used to "watch over" the things and relations within the spreadsheet. The business rules can be made available publicly via a URL, privately via a private URL, etc.

6. **Managed global standard**: The better spreadsheet should be a global standard under the control of someone like OMG, XBRL International, ISO, Apache OpenOffice, or some other such organization.

7. **Provide a formal shape but be domain neutral filler**: One formal shape should be agreed to, for example the multidimensional model, but the pieces which fit into that shape or "fill" the shape are domain neutral, controlled by the business domain.

8. **Format should allow for versioning, collaboration, etc.**: The syntax format should allow for ease of versioning, constructing systems which are collaborative in nature (multi-user).

[35] See, *Need for New Global Standard Spreadsheet Alternative*, http://xbrl.squarespace.com/journal/2014/5/3/need-for-new-global-standard-spreadsheet-alternative.html

9. **Straightforwardly usable over the Internet**: The format should be compliant with internet standards.
10. **Support a wide variety of common business use cases**: A wide variety of common business use cases would be served, but it is not a goal to solve every business problem which exists.
11. **Highly limited options**: The number of optional features is to be kept to the absolute minimum, ideally zero. Multiple approaches to solving a problem are not necessary when one will do.
12. **Formal and concise design**: The design must be formal, concise, well designed and well-engineered.