1. Principles

This section provides a set of principles that helps the reader think about XBRL-based digital financial reports.

Principles help you think about something thoroughly and consistently. Overcoming disagreements between stakeholders and even within groups of stakeholders is important. Agreement between stakeholder groups and within stakeholder groups contributes to harmony. Lack of agreement contributes to dissonance. Principles help in the communications process.

I would argue that a first step, if not the first step, of arriving at harmony is **outlining the interests, perceptions, positions, and risks** of each constituency/stakeholder group[1].

A "stakeholder" is anyone that has a vested interest. Another term for stakeholder is "constituent". A "constituent" is a component part of something.

Foundational to arriving at harmony is having a common conceptual framework including a set of consistent principles or assumptions or world view for thinking about the system. For example, accounting and financial reporting have such a conceptual framework including principles/assumptions such as "materiality" and "going concern" and "conservatism".

This "framework for agreeing" helps the communications process which increases harmony and decreases dissonance. This is about bringing the system into balance, consciously creating the appropriate equilibrium/balance.

The following is a set of principles which professional accountants can use to understand their perceptions, positions, and risks when it comes to financial reports. None of these principles is technical, all should be easy to understand.

1.1. Prudence dictates that using financial information from an XBRL-based digital financial report should not be a guessing game.

Safe, reliable, predictable, automated reuse of reported financial information by machine-based processes is preferable to creating a guessing game. Imagine numerous different software developers creating algorithms to use XBRL-based financial information. What helps guarantee that the results returned by each software algorithm are the same where they should be the same? How useful is such an XBRL-based financial report to automated machine-based processes if the reports contain defects?

General purpose financial reports tell a story. Different business professionals using different software tools must derive the same meaning from the same financial report. While business professionals are free to interpret the conveyed meaning of financial information as they might choose; the conveyed meaning itself should be objective and not be subject to interpretation.

[1] Charles Hoffman, CPA, *Professional Accountant's Interests, Perspective, Position, and Risks*, http://www.xbrlsite.com/mastering/Part01_Chapter02.I_ProfessionalAccountantsPerspective.pdf

1.2. A near zero defect financial report is useful, a defective financial report is not.

It is difficult, perhaps even impossible, for humans to create things that don't have errors. But a conscious command of rigorous processes and standards of excellence can contribute to minimizing defects[2]. But what is an acceptable defect rate?

The Six Sigma[3] philosophy offers a target acceptable defect rate of 0.00034% or 99.99966% correct. This philosophy can be applied to the information contained within an XBRL-based digital financial report. Something along those lines is likely appropriate.

Defects can be identified by taking measurements. The extent to which something is correct can likewise be determined using measurements. But how do you distinguish between something that is correct (i.e. not a defect) and something that is a defect? The answer is rules.

1.3. Rules prevent anarchy.

Anarchy is defined as "a situation of confusion and wild behavior in which the people in a country, group, organization, etc., are not controlled by rules or laws." Rules[4] prevent anarchy.

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

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

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

The meaning of a business rule is separate from the level of enforcement someone might apply to the rule.

1.4. The only way to achieve a meaningful exchange of information without dispute is with the prior existence of and agreement as to a standard set of technical syntax rules, business semantics rules, and workflow rules.

Meaningful exchange[5] relates to exchange without disputes as to precise meaning, it means unambiguous interpretation, it means resolving conflicts and inconsistencies.

---

[2] Charles Hoffman, CPA, *Method*, http://www.xbrlsite.com/mastering/Part02_Chapter05.N1_Method.pdf

[3] Wikipedia, *Six Sigma, Sigma Levels*, retrieved November 25, 2016, https://en.wikipedia.org/wiki/Six_Sigma#Sigma_levels

[4] Charles Hoffman, CPA, *Rules*, http://www.xbrlsite.com/mastering/Part02_Chapter05.F_Rules.pdf

Consider this scenario: Two public companies, A and B, each have some knowledge about their financial position and financial condition. They must communicate their knowledge to an investor who is making investment decisions which will make use of the combined information so as to draw some conclusions. All three parties are using a common set of basic logical principles (facts known to be true, deductive reasoning, inductive reasoning, etc.) and common financial reporting standards (i.e. US GAAP, IFRS, etc.), so they should be able to communicate this information fully, so that any inferences which, say, the investor draws from public company A's input should also be derivable by public company A using basic logical principles and common financial reporting standards, and vice versa; and similarly for the investor and public company B.

What goes into a financial report can be subjective, subject to professional judgement. How the report itself functions is completely objective, subject to logical, mechanical, and mathematical rules. Creators and users of such information should understand the intended logical interpretation of the information.

1.5. Explicitly stated information or reliably derived information is preferable to implicit information.

The rules of logic are well developed and understood. Formal logic is used to precisely describe complex systems such as safety critical railway signaling, medical device functionality, and nuclear power systems, or our system of mathematics.

Facts can be true or facts can be false; but a fact cannot be both true and false in the same system[6]. The well-established rules of deductive reasoning and inductive reasoning can be used to reliably derive new facts from existing facts. Logical deduction and induction is a completely different process from implying information. Implying is basically making an educated guess based on incomplete explicit or derived facts. When information is implied, two different rational people can arrive at two different answers to the same question and both can be correct. The important point here is that explicitly provided facts, logically derived facts, and implying information are different processes.

Basically, if information is vague, ambiguous, contradictory, or unclear; a computer process working with such information can, at best, return something that is vague, ambiguous, contradictory, unclear, or nothing at all. It is really that straight forward.

1.6. Digital financial reports can be guaranteed to be defect free using automated processes to the extent that machine-readable business rules exist.

Point #4 above states that meaning can be exchange reliably only to the extent that business rules are provided. Those business rules can come in two forms: human-readable and machine-readable. It is only to the extent that machine-readable business rules are available to automated machine-based processes that those automated processes can guarantee an XBRL-based digital financial report to be

---

[5] Charles Hoffman, CPA, *Exchanging Complex Financial Information*, http://www.xbrlsite.com/mastering/Part02_Chapter05.A_ExchangingComplexFinancialInformation.pdf

[6] Charles Hoffman, CPA, *Logical Systems*, http://www.xbrlsite.com/mastering/Part02_Chapter05.A_LogicalSystems.pdf

defect free. Defect free is defined as the objectivize logical, mechanical, and mathematical relations between reported facts. Beyond those machine-readable business rules, manual processes are necessary to detect and correct defects.

1.7. When possible to effectively create, machine-based automated processes tend to be more desirable than human-based manual processes because machine processes are more reliable and cost less.

Machines are good at performing repetitive tasks. Humans are good at other things. Machines should do what machines are good at and can effectively do; humans should do what humans are good at and humans can effectively do and what machine-based automated processes cannot do.

Humans augmented by machine capabilities, much like an electronic calculator enabling a human to do math quicker, will empower knowledge workers who know how to leverage the use of those machines.

1.8. Computers have limited reasoning capacity.

Computers are machines[7]. Computers are good at performing repetitive tasks, over and over, reliably. Computers are not good at: intuition, creativity, innovation, improvisation, exploration, imagination, judgement, politics, law, unstructured problem solving, non-routine tasks, identifying and acquiring new relevant information, compassion. Machines should do things that machines are good at, humans should do things that humans are good at.

1.9. Business rules should be created by knowledgeable business professionals, not information technology professionals.

Article 9 of the *Business Rules Manifesto*[8] states, that business rules are of, by, and for business people, not information technology people. Business rules should arise from knowledgeable business people. Business people should have tools available to help them formulate, validate, maintain, and otherwise manage rules. Business people should have tools available to help them verify business rules against each other for consistency.

Business professionals need to learn how to create, debug, and maintain the business rules that drives the digital age.

In an interview with *Wired* magazine[9], Barak Obama, then president of the United States, discussing artificial intelligence made the following statement about self-driving cars:

---

[7] Charles Hoffman, CPA, *Computational Thinking*, http://www.xbrlsite.com/mastering/Part00_Chapter01.C_ComputerEmpathy.pdf

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

[9] Wired, *Barack Obama, Neural Nets, Self-driving Cars, and the Future of the World*, https://www.wired.com/2016/10/president-obama-mit-joi-ito-interview/

> "There are gonna be a bunch of choices that you have to make, the classic problem being: If the car is driving, you can swerve to avoid hitting a pedestrian, but then you might hit a wall and kill yourself. It's a moral decision, and who's setting up those rules?"

This example which relates to self-driving cars points out two things that accounting professionals need to consider when thinking about XBRL-based digital financial reports: (1) who writes the rules, the logic, which software follows, (2) how do you write those rules and put them into machine readable form? Do you want software developers creating your rules?

1.10. The stronger the problem solving logic, the more a machine can achieve.

Problem solving logic is basically the extent to which a business rules engine can solve problems. Other terms for problem solving logic are expressive power or reasoning capacity. There are two inputs to solving problems: (1) the rules which can be expressed in machine-readable form and (2) the ability of a business rules engine to process those rules. Business rules engines have a problem solving method the most common being forward chaining.

1.11. Catastrophic logical failures are to be avoided at all cost; they cause systems to completely fail.

If a system can break or cease to operate for unknown reasons or at any time, the system is not predictable and therefore not reliable. Computer systems tend to be implemented using a safe subset of first-order logic because classical higher-order logics cannot be safely and reliably implemented in the form of software programs. An easy way to understand this is to think of an infinite loop. If a computer program gets into an infinite loop from which it cannot escape, the program ceases to function. While the maximum problem solving logic is desirable, that must be balanced on the side of safety, predictability, and reliability; erroring on the side of safety.

1.12. Complexity cannot be removed from a system, but complexity can be moved.

The *Law of Conservation of Complexity*[10] states that every software application has an inherent amount of irreducible complexity. That complexity cannot be removed from the software application. However, complexity can be moved. The question is: Who will have to deal with the complexity? Will it be the application user, the application developer, or the platform developer which the application leverages? Poor choices mean hard to use software.

1.13. Part of a system is not really that useful.

Irreducible complexity is explained as follows: A single system which is composed of several interacting parts that contribute to the basic function, and where the removal of any one of the parts causes the system to effectively cease functioning.

[10] Larry Tesler, *Law of Conservation of Complexity*, http://www.nomodes.com/Larry_Tesler_Consulting/Complexity_Law.html

So for example, consider a simple mechanism such as a mousetrap. That mousetrap is composed of several different parts each of which is essential to the proper functioning of the mousetrap: a flat wooden base, a spring, a horizontal bar, a catch bar, the catch, and staples that hold the parts to the wooden base. If you have all the parts and the parts are assembled together properly, the mousetrap works as it was designed to work.

But say you remove one of the parts of the mousetrap. The mousetrap will no longer function as it was designed, it will not work. That is irreducible complexity: the complexity of the design requires that it can't be reduced any farther without losing functionality.

A non-functioning system is not useful. A partially functioning system is only partially useful.

1.14. Simplicity and simplistic are not the same thing.

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

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

Creating something that is complex is easy. Creating something that is simple is hard and requires more work.

A kluge, a term from the engineering and computer science world, refers to something that is convoluted and messy but gets the job done.

1.15. Apply double-entry bookkeeping procedures, processes, and techniques to digital financial reports.

Single-entry bookkeeping[11] is how 'everyone' would do accounting. In fact, that is how accounting was done before double-entry bookkeeping was invented[12].

Double-entry bookkeeping[13] adds an additional important property to the accounting system, that of a clear strategy to identify errors and to remove them from the system. Even better, it has a side effect of clearly firewalling errors as either accident or fraud. This then leads to an audit strategy. Double-entry bookkeeping is how professional accountants do accounting.

---

[11] Wikipedia, *Single-entry Bookkeeping System*, retrieved August 30, 2016, https://en.wikipedia.org/wiki/Single-entry_bookkeeping_system

[12] Charles Hoffman, CPA, *Essence of Accounting*, http://www.xbrlsite.com/mastering/Part00_Chapter01.D_EssenceOfAccounting.pdf

[13] Wikipedia, *Double-entry Bookkeeping System*, retrieved August 30, 2016, https://en.wikipedia.org/wiki/Double-entry_bookkeeping_system

Double-entry bookkeeping was the invention of medieval merchants and was first documented by the Italian mathematician and Franciscan Friar Luca Pacioli[14]. Double-entry bookkeeping is one of the greatest discoveries of commerce and its significance is difficult to overstate.

Which came first, double-entry bookkeeping or the enterprise[15]? Was it double-entry bookkeeping and what it offered that enable the large enterprise to exist; or did the large enterprise create the need for double-entry bookkeeping?

Accountants think differently than non-accountants, it is part of their training. Non-accountants don't realize this and accountants tend to forget or take this for granted. The quality difference between the set of facts that makes up a financial report and all the support for that financial report tends to be much higher than the quality level of non-financial information that is managed by a non-accountant. Why? Because double-entry bookkeeping is ingrained in the processes, procedures, and techniques of professional accountants.

What information technology professionals see as redundancies and opportunities for error are really more similar to a parity check[16] or a checksum[17] and opportunities for making certain that you are not making a mistake.

Every accountant learns that when analyzing an account: beginning balance + additions – subtractions = ending balance. If you know any three values, you can always find the fourth value. But if you know all four values then you can prove that all the values are accurate. The same is true about the facts contained within a financial report. Say *Revenues*, *Cost of Revenues*, and *Gross Profit* are reported in a financial report. If you know those three facts and you know that there is a business rule that specifies that *Gross Profit* = *Revenues* – *Cost of Revenues* and the facts and the business rule are consistent with your expectation; you can rely on the information as being accurate. Apply this technique to all the facts of an XBRL-based digital financial report and you get a near zero defect report.

Accountants, don't under estimate the value of double-entry bookkeeping and the other processes, procedures, and techniques employed to make sure that everything "ticks and ties" and "cross casts and foots". These useful techniques, even perhaps better referred to as ingrained medieval traditions, should make their way into XBRL-based digital financial reports. These medieval techniques are still very relevant even in the digital age.

[14] Wikipedia, *Luca Pacioli*, retrieved August 30, 2016, https://en.wikipedia.org/wiki/Luca_Pacioli

[15] Ian Grigg, *Triple Entry Accounting, A Very Brief History of Accounting, Which Came First - Double Entry or the Enterprise?,* http://iang.org/papers/triple_entry.html

[16] Wikipedia, *Parity check*, retrieved December 6, 2016, https://en.wikipedia.org/wiki/Parity_bit

[17] Wikipedia, *Checksum*, retrieved December 6, 2016, https://en.wikipedia.org/wiki/Checksum