XBRL: Understanding What Can Go Wrong

Understanding XBRL-based digital financial report by looking at the impediments that get in the way of creating a properly functioning XBRL-based digital financial report

by

Charles Hoffman, CPA

Golden Edition (DRAFT 2022-06-15)

http://xbrlsite.azurewebsites.net/2021/essentials/Index.html

About the author:

Charles Hoffman, CPA, is credited as being the *Father of XBRL*. He started his public accounting career as an auditor with the international firm then called Price Waterhouse, served various roles in industry and public accounting for over 25 years, and has worked with XBRL since its introduction by the AICPA in 1998. In 2006, he received the AICPA Special Recognition Award for his pioneering role in developing XBRL. He has authored numerous publications including *XBRL for Dummies*, a number of *Journal of Accountancy* articles, writes a blog relating to XBRL, and contributed to a number of XBRL related technical specification and best practices documents. Currently, Charlie works as a consultant to CPAs and software vendors who want to better understand the subtle details of this new digital medium.

Charlie was co-editor of the first ever US GAAP XBRL taxonomy, contributor to the *XBRL 2.1 Specification* and the *XBRL Dimensions* specification, editor of the *Financial Reporting Taxonomy Architecture* and *Financial Reporting Instance Standards*, co-author of the *US GAAP Taxonomy Architecture*, part of the project team which created the *US GAAP Taxonomy*, and a major contributor to the IFRS XBRL taxonomy for a five-year period, and consultant to numerous other XBRL taxonomy projects.

Acknowledgements:

While I did physically create the information in this resource, I could have not done so without the gracious help of a number of people, directly and indirectly, over the years. I see myself as merely a custodian of this important information, nurturing it along for the benefit of all, condensing countless discussions into something hopefully useful for the common good.

I would like to specifically thank these contributors: Rene van Egmond, Walter Hamscher, PhD, Geoff Shuetrim, PhD; David vun Kannon; Rene van Egmond; Thomas Egan, CPA; Josef Macdonald, CA; Jim Richards; Roger Debreceny; Jeff Naumann, CPA; David Prather, Alan Teixeira, CA; Hugh Wallis; Allyson Ugarte; Colm O hAonghusa; Giancarlo Pellizzari; Yossef Newman, CPA; Rob Blake; Mark Creemers; Marc van Hilvoorde; Herman Fischer; Ignacio Hernandez-Ros; Dean Ritz; Timothy Randle; Cliff Binstock; David Scott Stokes; Masatomo Goto; Paul Warren; Mark Goodhand; Campbell Pryde, CPA; Michele Romanelli; Maciej Piechocki, PhD; Victor Morilla; Mike Rowling; Joe Ryba, CPA; Matthias Brantner; Dennis Knochenwefel; Ghislain Fourny, PhD; Daniel Taylor, Chris Taylor, CPA, Thomas McKinney, CPA; Eric Cohen, CPA; Mike Willis, CPA; Louis Matherne, CPA; Dudley Gould; Elisa Kendall.

There are others which I probably left off and for this I apologize. I acknowledge and appreciate the thinking others contributed to this endeavor.

Copyright (full and complete release of copyright)

"I skate to where the puck is going to be, not where it has been." Wayne Gretzky, legendary Canadian hockey star

"Being ignorant is not so much a shame, as being unwilling to learn." Benjamin Franklin

1. Introduction

An excellent way of making sure things are right is by studying what can go wrong.

The following is a comparison of 9 states of the simple financial report logical system, the accounting equation[1]. The point of using such a simple financial report logical system is to explain specific things that can go wrong so that a reader can understand why each of the categories of rules are necessary to control a process to make sure things are in fact right. These 9 states can occur in any financial report with one fragment, two fragments, or 194 fragments like the Microsoft 10-K.

Readers of this document are encouraged to read the document *Essentials of XBRL-based Digital Financial Reporting*[2] and the Essence of Accounting[3] prior to reading this document.

Again, *Mastering XBRL-based Digital Financial Reports*[4] walks you through small and simple to large and complex. For more information about these impediments, please watch the video playlist Understanding the Financial Report Logical System[5], in particular the three videos related to impediments.

Here is a summary of all nine states with the first state outlined in green being the only properly functioning logical system proven to be complete, consistent, and precise:

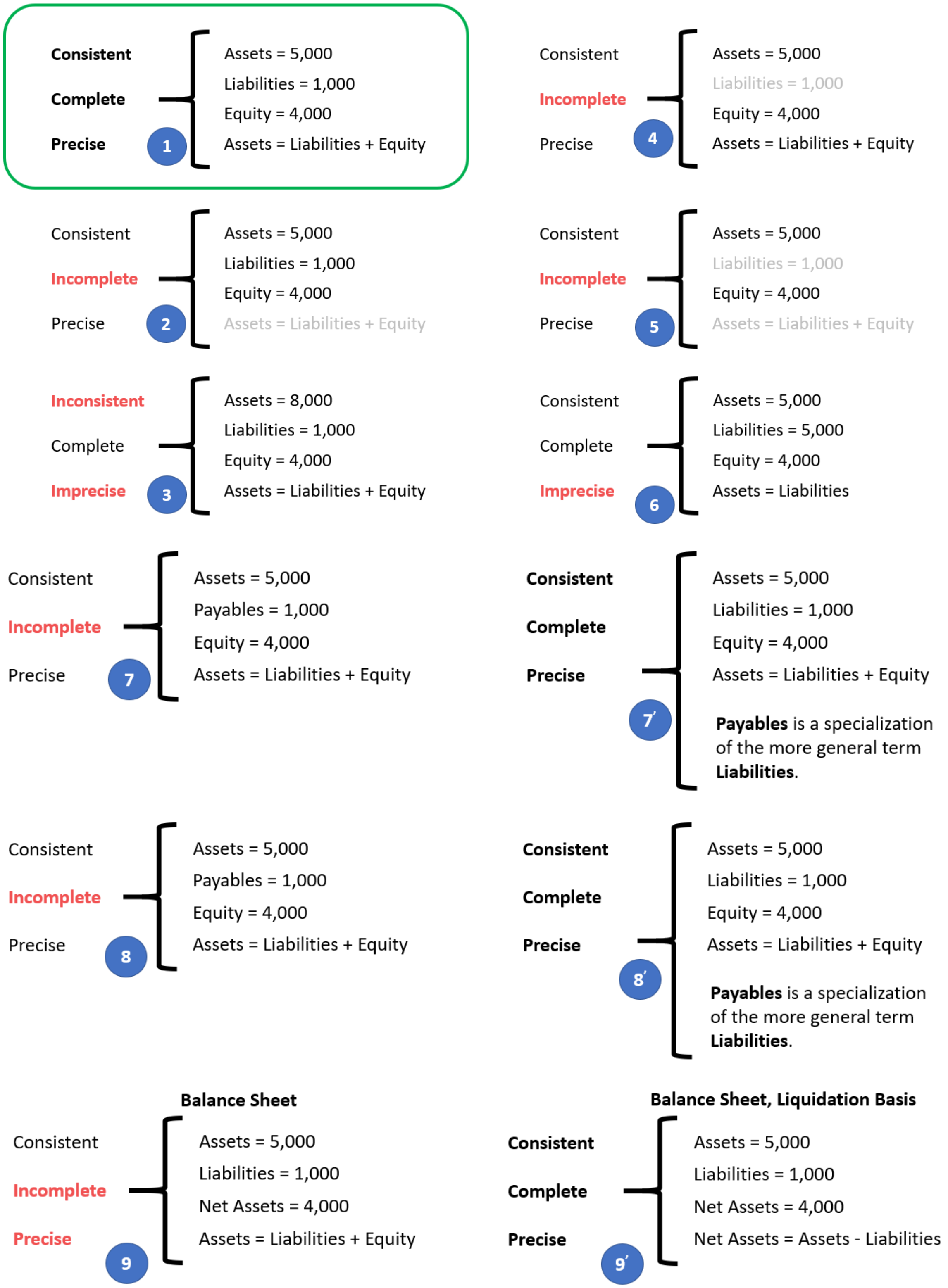
[1] Accounting equation, http://xbrlsite.azurewebsites.net/2020/master/ae/index.html

[2] Charles Hoffman, CPA, *Essentials of XBRL-based Digital Financial Reporting*, http://xbrlsite.azurewebsites.net/2021/essentials/EssentialsOfXBRLBasedDigitalFinancialReporting.pdf

[3] Charles Hoffman, CPA, *Essence of Accounting*, http://xbrlsite.azurewebsites.net/2020/Library/EssenceOfAccounting.pdf

[4] Mastering XBRL-based Digital Financial Reports, http://xbrlsite.azurewebsites.net/2020/master/

[5] Understanding the Financial Report Logical System, https://www.youtube.com/playlist?list=PLqMZRUzQ64B7EWamzDP-WaYbS_W0RL9nt



1
Consistent
Complete
Precise
Assets = 5,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity

4
Consistent
Incomplete
Precise
Assets = 5,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity

2
Consistent
Incomplete
Precise
Assets = 5,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity

5
Consistent
Incomplete
Precise
Assets = 5,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity

3
Inconsistent
Complete
Imprecise
Assets = 8,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity

6
Consistent
Complete
Imprecise
Assets = 5,000
Liabilities = 5,000
Equity = 4,000
Assets = Liabilities

7
Consistent
Incomplete
Precise
Assets = 5,000
Payables = 1,000
Equity = 4,000
Assets = Liabilities + Equity

7′
Consistent
Complete
Precise
Assets = 5,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity
Payables is a specialization of the more general term **Liabilities**.

8
Consistent
Incomplete
Precise
Assets = 5,000
Payables = 1,000
Equity = 4,000
Assets = Liabilities + Equity

8′
Consistent
Complete
Precise
Assets = 5,000
Liabilities = 1,000
Equity = 4,000
Assets = Liabilities + Equity
Payables is a specialization of the more general term **Liabilities**.

Balance Sheet
9
Consistent
Incomplete
Precise
Assets = 5,000
Liabilities = 1,000
Net Assets = 4,000
Assets = Liabilities + Equity

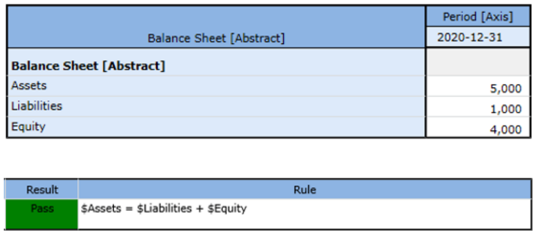
Balance Sheet, Liquidation Basis
9′
Consistent
Complete
Precise
Assets = 5,000
Liabilities = 1,000
Net Assets = 4,000
Net Assets = Assets - Liabilities

In the following sections I want to make some adjustments to the logical system which make the logical system either inconsistent, incomplete, or imprecise and explain why the system is then not a properly functioning logical system. I made videos that explain each of these impediments to a properly functioning logical system which you can see in this video playlist, *Understanding the Financial Report Logical System*[6].

Before we get to the improperly functioning logical systems, let's take one final look at the properly functioning logical system so that you can use that as a baseline for comparing and contrasting the properly functioning and improperly functioning logical systems so that you can understand the sorts of errors that can occur.

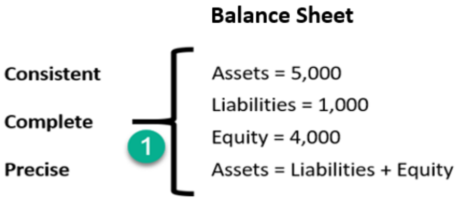
1.1. State 1: Properly Functioning Logical System

For completeness, I want to start by mentioning again our properly functioning logical system which is consistent, complete, and precise. It can be helpful to contrast other states to this state to understand the difference between properly functioning logical systems and improperly functioning systems.



Balance Sheet [Abstract]	Period [Axis] 2020-12-31
Balance Sheet [Abstract]	
Assets	5,000
Liabilities	1,000
Equity	4,000

Result	Rule
Pass	$Assets = $Liabilities + $Equity



Again, this is considered a properly functioning logical system because (a) all the statements within the system are **consistent**; (b) the set of statements that describe the system is **complete**; and (c) the information conveyed by the system is **precise** in its representation of reality. Further, we are formally declaring this "reality"[7] to be our base understanding.

Also, we need to be explicit. We defined three terms "Assets", "Liabilities", and "Equity".

Now, you may know what those three terms are; but a computer does not. You have to define what you work with relative to something that you know. Imagine our system defines four terms, "fac:Assets", "fac:Liabilities", "fac:Equity", and "fac:LiabilitiesAndEquity"[8]. You understand your system but you have to map every external system into your system[9]. Your internal system understands more that the accounting equation system (i.e. you have LiabilitiesAndEquity). You have to be able to compute that value based on some other system's information[10]. It is perfectly reasonable for our system to create a concept LiabilitiesAndEquity and compute that value even though the accounting equation logical system does not have that explicit value.

[6] Understanding the Financial Report Logical System, https://www.youtube.com/playlist?list=PLqMZRUzQ64B7EWamzDP-WaYbS_W0RL9nt

[7] YouTube, *Reality*, https://youtu.be/eq2Jw6waaCI

[8] Fundamental accounting concepts, http://xbrlsite.azurewebsites.net/2020/core/master-ae/fac.xsd

[9] Mapping from accounting equation to fundamental accounting concepts in our system, http://xbrlsite.azurewebsites.net/2020/core/master-ae/fac-mapping-definition.xml

[10] XBRL Formula to derive the value for LiabilitiesAndEquity, http://xbrlsite.azurewebsites.net/2020/core/master-ae/fac-ImputeRule-LiabilitiesAndEquity-formula.xml

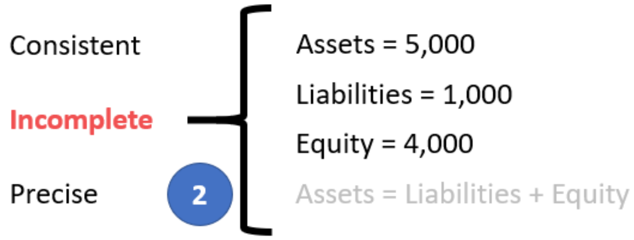
The point is that different economic entities have different models; but all models of a financial reporting scheme are reconcilable from/to one another in some manner[11].

1.2. State 2: Incomplete Coverage by Rules

The logical system #2 below is intended to show exactly the same information as our #1 properly functioning logical system, except that #2 leaves out the rule "Assets = Liabilities and Equity" which is showed as grayed out (i.e. because it is assumed to be missing from the logical system.

Coverage is a measure of how well you do or can represent a domain of information within a logical system. "Do" is about using the tools you have correctly and effectively. "Can" is about the capabilities of the tools you are using to represent the rule.

For example, if your logical system neglects to include the rule "Assets = Liabilities + Equity" or if your tools don't provide the capabilities to allow you to represent that rule; then there is the possibility that the facts being represented to be represented incorrectly and the system will not detect the inconsistency. As such, that logical system has **incomplete coverage**.

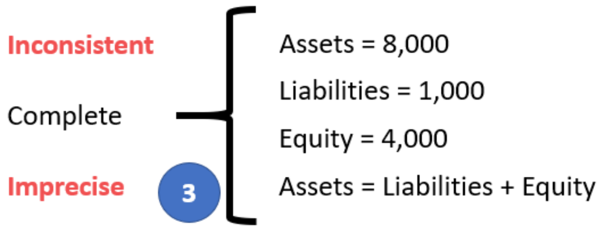


While this specific state #2 does have the Assets, Liabilities, and Equity facts consistent with the absent rule; the system is still incomplete because the coverage can be improved by adding the missing rule. If that missing rule is added, then the logical system can be considered complete again.

1.3. State 3: Inconsistent and Imprecise

All the statements in the system must be consistent for the logical system to be considered properly functioning. If statements are inconsistent, the logical system is not is not properly functioning. In this system #3, the values for Assets, Liabilities, and Equity are inconsistent with the rule "Assets = Liabilities + Equity". From looking at the information provided, it is impossible to know exactly which of the three facts are incorrect; it is only possible to understand that the statements made within the logical system is inconsistent. It could be the case that the rule is incorrect.

[11] Charles Hoffman, CPA, *Special Theory of Machine-based Automated Communication of Semantic Information of Financial Statements*, http://xbrl.squarespace.com/journal/2019/12/30/special-theory-of-machine-based-automated-communication-of-s.html

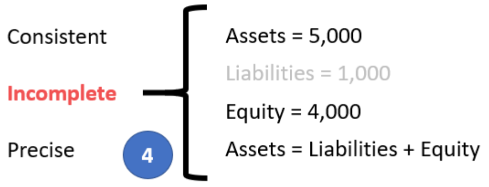


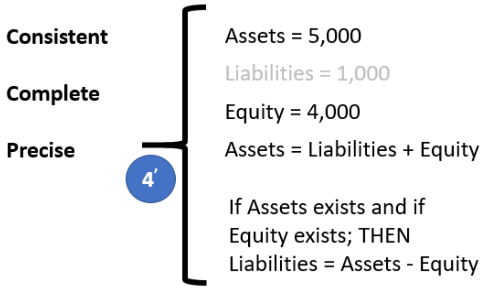
However, given that we know from state #1 that the value for Assets is 5,000 and not 8,000; the facts in this system is imprecise because the fact for Assets does not reflect reality.

1.4. State 4: Unreported Facts

In state #4, the situation is that the economic entity representing information in their report neglected to include the fact for Liabilities. Whether it is the case that a fact can, or cannot, be left unreported is a decision that can be made by the stakeholders of the system.

If it is the case that it is decided that the fact "Liabilities" can be omitted if both Assets and Equity are reported; then you must provide a rule to derive the value of Liabilities when that fact is not reported. Below you see that the system has been adjusted in state #4' to add the rule "IF Assets exists and if Equity exists; THEN Liabilities = Assets - Equity"[12]. *(NOTE that this rule should actually be "IF Assets exists and if Equity exists and if not(exists) Liabilities; THEN Liabilities = Assets – Equity")*





If it were likewise true that either Assets[13] or Equity[14] could also be left unreported, similarly derivation rules could be created for each of those facts. Note that XBRL Formula chaining[15] can be used to physically derive unreported facts if any one of these three facts remain unreported. Note that it is impossible to derive missing information if any two of the facts remain unreported. Adding the derivation rule makes the system complete.

[12] Here is the impute or derivation rule that would be added to the accounting equation logical system for this situation, http://xbrlsite.azurewebsites.net/2020/core/master-ae/ImputeRule-Key-1-Code-BS-Impute-01-formula.xml

[13] XBRL Formula rule for deriving Assets, http://xbrlsite.azurewebsites.net/2020/core/master-ae/ImputeRule-Key-3-Code-BS-Impute-03-formula.xml

[14] XBRL Formula rule for deriving Equity, http://xbrlsite.azurewebsites.net/2020/core/master-ae/ImputeRule-Key-2-Code-BS-Impute-02-formula.xml

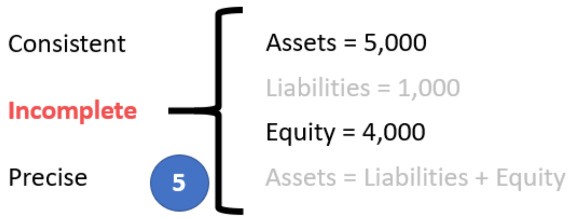
[15] Deriving Facts Using XBRL Formula Chaining (Example), http://xbrl.squarespace.com/journal/2019/4/24/deriving-information-using-xbrl-formula-chaining-example.html

Allowing certain line items of a report to go unreported specifies the need to create rules to derive missing information. Or saying this another way, omitting the possibility of unreported facts negates the need for creating derivation rules.

A second downside of allowing unreported facts is that you lose the parity check or cross check if facts can go unreported. Said another way, it would be considered best practice to not leave important high-level financial report line items to go unreported.

1.5. State 5: Incomplete

Similar to state #4, in state #5 the logical system is incomplete because both (a) the fact Liabilities is unreported and also (b) the consistency rule "Assets = Liabilities + Equity" is missing from the logical system. Because both a fact and the rule are missing from the logical system, it would be impossible to deduce the value of Liabilities in this case. There is not enough information in the logical system to allow Liabilities to be derived. At a minimum, a consistency crosscheck rule[16] plus the derivation rule to impute Liabilities[17] would be necessary.

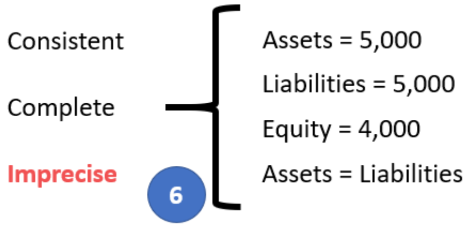


Again, consistent with state #4; Assets and Equity would require similar rules and there is no parity check of reported information.

1.6. State 6: Imprecise

A logical system is a true and fair representation of some agreed upon realism. **Precision** is a measure of how precisely you do or can represent the information of a domain within a logical theory. The reality that we formalized in state #1 indicates that "Assets = Liabilities + Equity". Yet, in the state #6 example, the rule "Assets = Liabilities" was provided. Further, the values of Assets and Liabilities are, in fact, consistent with the rule that has been provided.

Remember that in state #1 we formalized our truth to be that "Assets = Liabilities + Equity". As such, this logical system can be described as being imprecise. To make this logical system precise, all that needs to be done is to fix the rule.

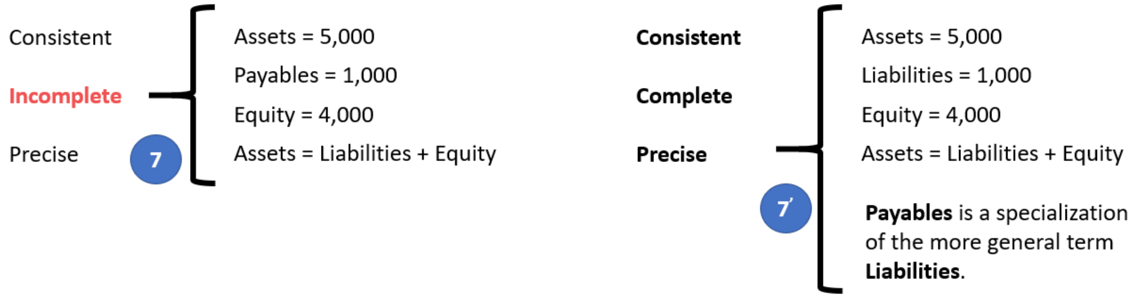


[16] XBRL Formula consistency crosscheck rule Assets = Liabilities + Equity, http://xbrlsite.azurewebsites.net/2020/core/master-ae/Consistency-5-Code-BS01-formula.xml

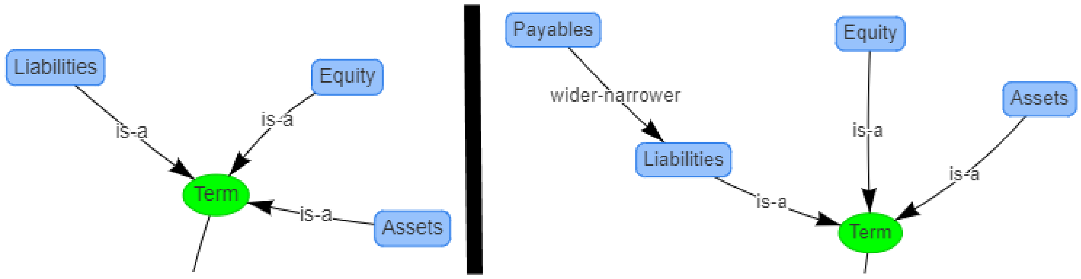
[17] XBRL Formula derivation rule to impute Liabilities, http://xbrlsite.azurewebsites.net/2020/core/master-ae/ImputeRule-Key-1-Code-BS-Impute-01-formula.xml

1.7. State 7: Extension Concept

In state #7 on the left, what we are trying to convey is that the economic entity reported the fact for Liabilities using the extension concept "Payables" that it had created. If a fact is represented using an extension concept created by a reporting entity; then a "general-special" or "wider-narrower" or "class-equivalentClass" association must be created to indicate to software applications of the relationship so that information can be used correctly. State #7' on the right, the rule "Payables is a specialization of the more general term Liabilities" has been added to the logical system which allows the system to operate effectively[18].



And so, the graphic below shows a fragment of the knowledge graph on page 9 above before and after the information that "Payables is a specialization of the more general term Liabilities," was added. On the left you see State 7, the taxonomy before the information was added and on the right you see "Payable" being added as an extension concept indicating that there is a "wider-narrower" relationship between Payables and Liabilities. Therefore, a machine based process can utilize the information per State 7' because the process understands Liabilities in the taxonomy, understands the "wider-narrower" relationship therefore knowing that "Payables" is a type of Liability.



1.8. State 8: Base Taxonomy Wider/Narrower Concept Use

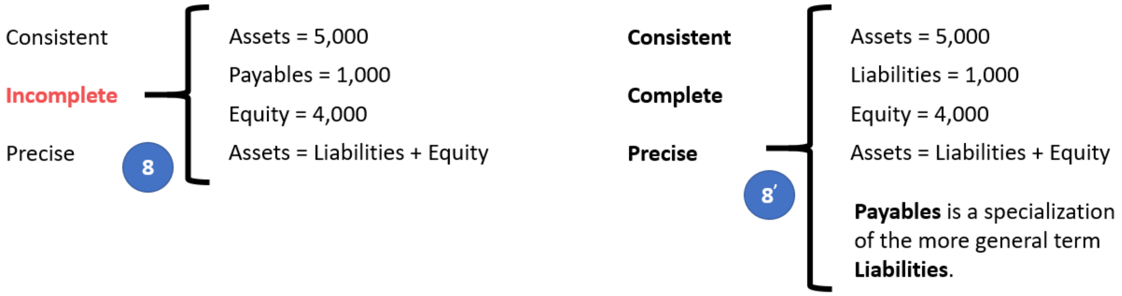
State #8 on the left below is similar to state #7 in that a different concept is used to report a fact; but while state #7 focuses on using an extension concept; state #8 points out that using a wider or narrower base taxonomy concept gives exactly the same result.

Now, our base state #1 does not have the concept "Payables"; but let's assume for a moment that it does have the concept "Payables". Also suppose that there was no

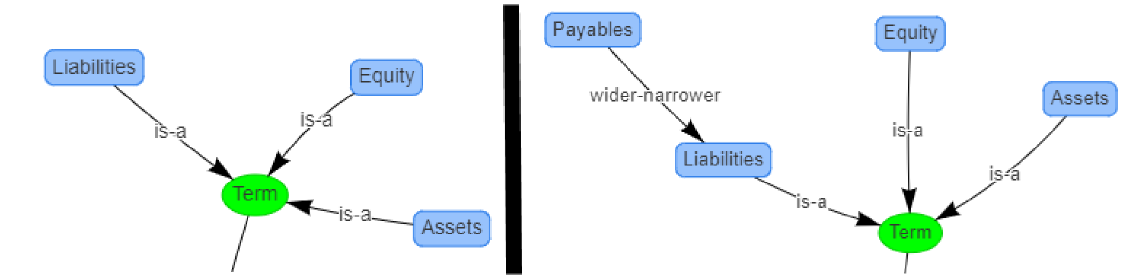
[18] XBRL Definition relations showing example of a mapping rule, http://xbrlsite.azurewebsites.net/2020/core/master-ae/fac-mapping-definition.xml

information in the base logical system indicating the relationship between "Payables" and any other concept. If a fact is represented using a BASE TAXONOMY CONCEPT by a reporting entity; then a "general-special" or "wider-narrower" or "type-subtype" association must exist in that base taxonomy to indicate that some concept is a permissible alternative for some other concept.

State #8′ on the right adds the rule "Payables is a specialization of the more general term Liabilities"[19].



And so, the graphic below shows a fragment of the knowledge graph on page 9 above before and after the information that "Payables is a specialization of the more general term Liabilities," was added. On the left you see State 8, the base taxonomy before the information was added and on the right you see "Payable" being added as an extension concept indicating that there is a "wider-narrower" relationship between Payables and Liabilities. Therefore, a machine based process can utilize the information per State 8′ because the process understands Liabilities in the base taxonomy, understands the "wider-narrower" relationship therefore knowing that "Payables" is a type of Liability.



1.9. State 9: Defining a Completely New Structure

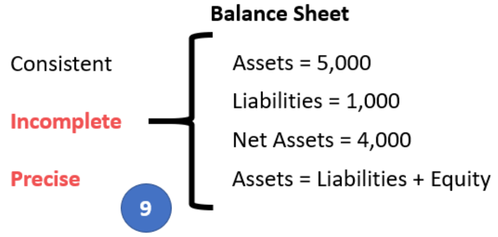
State #9 below on the left focuses on the structure as contrast all the prior examples which focused on the terms and rules. If a new structure is created, the new structure must be referenced to the base taxonomy and the new structure needs to be explained using machine-readable rules[20]. Even base taxonomy structures need to be defined in order to be referred to[21]. When you say "Balance Sheet" you know what that means. But a machine does not know.

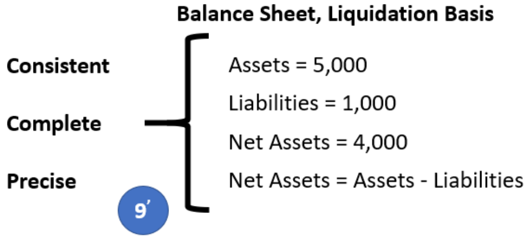
[19] XBRL Definition relations showing example of a mapping rule, http://xbrlsite.azurewebsites.net/2020/core/master-ae/fac-mapping-definition.xml

[20] XBRL Definition relations used to represent structure rules, http://xbrlsite.azurewebsites.net/2020/core/master-ae/dm-1355-rules-def.xml

[21] XBRL taxonomy schema used to define "Balance Sheet", http://xbrlsite.azurewebsites.net/2020/core/master-ae/disclosures.xsd

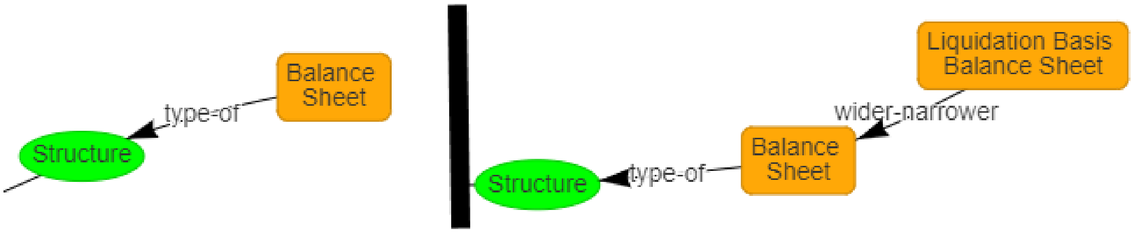
A base taxonomy should (a) provide all necessary structures separately, not intermingle different models in the same set of associations and (b) define what each structure must look like. Remember, computers are like babies and need to be led by the hand in order to understand the details you need them to understand.





Finally, in our case we have only one disclosure, the Balance Sheet. In our case, the Balance Sheet is always required to be reported per this logical system. As such, that rule is stated in a machine-readable reporting checklist[22]. Other logical systems with more disclosures will have more rules relating to when a disclosure is required to be provided in a report.

Similar to how "Payables" was added as an extension of the terms in the logical system; we can extend the structures to include a "Liquidation Basis Balance Sheet" structure which is a specialization of a Balance Sheet:



And such, an automated process will be able to understand the new structure because it is related to an existing structure. Other structures could be added and only identified as a type of structure.

2. Summary Narrative

The following summarizes in the form of a narrative what has been described by the different states of the accounting equation representation. Each of these states can be shown when using the Auditchain Pacioli logic engine to verify each of the representations in the above states[23].

- **State 1**: A report can be provably properly functioning per a set of rules. The State 1 report is an example of a properly functioning financial report. **All the logical statements are provided for per a set of machine-readable rules, the statements are consistent with one another, and the information is precise per the area of knowledge (in this case the accounting equation).** The Pacioli logic engine verifies all this.
- **State 2**: If you REMOVE a logical statement, such as if you REMOVE the rule "Assets = Liabilities + Equity"; and three facts are reported; a machine-based

[22] XBRL Definition relations used to represent a reporting checklist or disclosure rules, http://xbrlsite.azurewebsites.net/2020/core/master-ae/reporting-checklist-rules-def.xml

[23] Testing Impediments, http://xbrlsite.com/seattlemethod/golden/reports/ae/Testing_Impediments.pdf

process can have NO IDEA whether those three facts are CORRECT or INCORRECT using automated processes because the RULE IS MISSING. NO MACHINE-READABLE RULE, MACHINE CANNOT VERIFY.

- **State 3**: If you intentionally put information in the report to simulate an error; Pacioli DETECTS that error and a human can SEE that from the verification results. But for inconsistencies to be detected, the machine-readable rules must exist.
- **State 4**: If a line item is not reported, this causes increased complexity in processing reports. For example, if you leave out the line item "Liabilities" BUT you don't provide a machine-readable derivation rule, there is NO WAY a computer can UNDERSTAND THE information.
- **State 4'**: But as State 4' shows, if you DO provide the derivation rules, the system will still be able to automate effectively because the missing information can be derived. It just calls for additional work (i.e. creating the derivation rule BECAUSE not reporting certain line items is allowed). Don't what to cause additional work? Don't want to cause additional RISK of misunderstanding? Then DON'T ALLOW unreported high-level line items.
- **State 5**: If a line item is NOT reported AND the consistency rule is not provided; THEN there is no way a machine-based process can effectively use the report. Remember, computers are INCREDIBLY DUMB. They need to be led by the hand; that is what machine-readable rules do.
- **State 6**: It is possible to represent a report, get 100% of the verification checks to be GREEN; but the report is STILL WRONG. If (a) a FACT is wrong and (b) if a RULE is wrong; the two can work together to make a report seem CORRECT. However, this situation can be detected by using high-level crosschecks of the continuity of a report.
- **State 7**: If a report uses an extension concept to report a fact, the MACHINE will not understand how to process the extension information, so humans MUST step in to sort things out. As such, a process cannot be automated.
- **State 7'**: "Anchoring" helps OVERCOME State 7; the machine-readable anchoring information enables a process (a) to be automated and (b) the use of extensions to make the system more flexible which maximizes system "information richness".
- **State 8**: The EXACT SAME PROBLEM that anchoring solves with extension concepts exists with BASE TAXONOMY CONCEPTS!!! Again, computers are dumb, dumb, dumb. If anchoring is not also used in the base taxonomy, then you have EXACTLY the same problem you have with extensions.
- **State 9**: The EXACT SAME PROBLEM that anchoring solves exists with DISCLOSURES (i.e. SETS of concepts). Whether a disclosure (structure) is in a report model or base taxonomy model; if an automated process cannot sort out what that structure/disclosure is; then it WILL NOT KNOW WHAT TO DO WITH IT. Anchoring applies to structures/disclosures as it does to concepts, report or base models.
- **State 10**: The report models need to be represented logically. What does "indentation" of a concept mean? Whatever you thing it means, it means something else to someone else. Every Tom, Dick, and Harry has their own

personal interpretation. Don't use "indentation" (parent-child associations) to carry ambiguous meaning. Remember, computers are REALLY dumb.

- **State 11**: Fundamentally, the "payload" of information must be provided by a standard technical syntax. It an improper XBRL technical syntax is used, software may not work correctly. XBRL technical syntax tends to be pretty good because XBRL International publishes a conformance suite which can be used to test software to see if the software is detecting mistakes correctly in XBRL technical syntax.

3. Examining Errors in Actual Submitted Reports

As explained, a very good way to understand how to create reports correctly is to examine errors in reports that others have created. Here are three documents that provide well documented examples of undisputed errors:

- Issues in XBRL-based Digital Financial Reports[24]
- More Issues in XBRL-based Digital Financial Reports[25]
- Accounting errors[26]

Finally, this blog post[27] (see toward the bottom) provides 24 sets of well documented errors in XBRL-based reports.

Note that all of these errors were discovered using automated processes which leverage this method. If reports were verified using this method prior to submitting the report to a regulator, all such errors could have been detected and corrected. The result would be higher quality XBRL-based reports.

4. Best Practices for Preventing Errors

A **best practice** is a method or technique that has been generally accepted as superior to any other known alternatives because it produces results that are superior to those achieved by other means or because it has become a standard way of doing things.

Best practices (or good practices) are techniques that have produced outstanding results in other situations, inside or outside of a particular organization and which can be validated, codified, and shared with others and recommended as models to follow[28].

[24] Issues in XBRL-based Digital Financial Reports, http://xbrlsite.azurewebsites.net/2020/master/100IssuesOfPublicCompanyReports.pdf

[25] More issues in XBRL-based Digital Financial Reports, http://xbrlsite.azurewebsites.net/2020/master/About50MoreIssuesOfPublicCompanyReports.pdf

[26] Accounting Errors, http://xbrlsite.azurewebsites.net/2018/Library/AccountingErrorsFoundDuringValidation.pdf

[27] High Quality Examples of Errors in XBRL-based Financial Reports, http://xbrl.squarespace.com/journal/2017/4/29/high-quality-examples-of-errors-in-xbrl-based-financial-repo.html

[28] European Guide to good Practice in Knowledge Management - Part 5: KM Terminology, page 3 (PDF page 9), http://arielsheen.com/wp-content/uploads/2019/10/CEN-CWA14924-05-2004-Mar.pdf#page=9

When one understands these dynamics of an XBRL-based report when a reporting economic entity can "reshape" or "alter" or otherwise modify their report then one will understand why some method[29] for controlling such modifications is necessary. This control mechanism contributes to making sure that processes yield high-quality financial information that is provided by such reports.

The method I use is the ***Seattle Method***[30].

5. Rules of Thumb

The following is a summary of the rules of thumb for creating XBRL-based digital financial reports.

1. Financial reports are logical systems that are true and fair representations of the financial position and financial condition of an economic entity. That logical system, be it represented on paper, e-paper, or digitally must be proven to be properly functioning.
2. Use a proven method. Don't create a report by simply doing "stuff". Do things right and do the right things.
3. Don't unconsciously mix/switch dimensional and nondimensional modeling approaches within one information block.
4. Use good software that performs the task you need performed.
5. Always use dimension defaults for every dimension.
6. When creating a base taxonomy, each specific dimension should have a clearly specified dimension default and that dimension default must be used by every company creating a report.
7. Check your math using automated processes.
8. Always be consistent.
9. Copy good ideas, steer clear of bad ideas, and understand how to tell the difference.
10. Focus on logic, not the technology details. Good software hides the technical details in the background. If your software does not hide the technical details, bet better software.

6. Next Steps

To understand XBRL-based financial reporting in more detail please have a look at *Mastering XBRL-based Digital Financial Reports*[31]. That document is a summary and synthesis of all the information from my blog[32].

[29] *Understanding Method (Abridged)*, http://xbrlsite.azurewebsites.net/2020/Library/UnderstandingMethod_Abridged.pdf

[30] Charles Hoffman, CPA, Seattle Method, http://xbrlsite.com/seattlemethod/SeattleMethod.pdf

[31] *Mastering XBRL-based Digital Financial Reports*, http://xbrl.squarespace.com/mastering-xbrl/

[32] XBRL Blog Archive, http://xbrl.squarespace.com/blog-archive/