

Understanding and Leveraging the “Semantic Glue” of XBRL-based Financial Reports

Information useful to software engineers building software used by accountants to create and work with XBRL-based report models and reports

By Charles Hoffman, CPA (February 14, 2024) (Work in Progress)

Many software engineers tend to approach working with XBRL seems to be to pick up the XBRL technical specification, read that specification, and then start coding “an XBRL application”. Well, here is a news flash: accountants could not care a less about XBRL. What accountants care about are things like creating financial reports better, faster, and cheaper and the features that XBRL or something like XBRL can enable.

By changing one’s perspective and leveraging the characteristics offered by XBRL in terms of enabling a new way of thinking about financial reports¹; I contend that it will reveal a new, and better, approach to creating and otherwise working with financial reports whether those reports are ever serialized as XBRL or not.

The best way to understand the big picture of what I am saying is to understand the history of the manually created blueprint, CAD/CAM, and BIM². Financial reports used to be like the manually drawn blueprints created by a draftsman on paper. But systems evolve³. In the future, financial reports will be more like how Building Information Modeling (BIM) works.

This document helps accountants and software engineers think about financial reports differently. To understand the bigger picture, I would strongly suggest reading *Logical Digital Twin of Financial Reports*⁴ before you tackle this document. That understanding will be assumed by the readers of this document. If you want to get a good grounding in how to think about financial reports, I would also recommend *Essence of Accounting*⁵. Finally, if you already have experience with XBRL, I would also encourage you to read *Essentials of XBRL-based Digital Financial Reporting*⁶ which helps you understand important subtleties and nuances related to XBRL-based digital financial reporting.

¹ Charles Hoffman, CPA, *Financial Report Knowledge Graphs*,
<https://xbrlsite.azurewebsites.net/2021/Library/FinancialReportKnowledgeGraphs.pdf>

² *Using Difference Between CAD/CAM and BIM to Understand How to Create Financial Reporting Expert Systems*,
<https://digitalfinancialreporting.blogspot.com/2023/03/using-difference-between-cadcam-and-bim.html>

³ *Evolution of a System*, <http://xbrlsite.com/2023/Library/EvolutionOfSystem.pdf>

⁴ *Logical Digital Twin of Financial Reports*,
http://www.xbrlsite.com/mastering/Part02_Chapter05.A0_LogicalDigitalTwin.pdf

⁵ Charles Hoffman, CPA, *Essence of Accounting*,
<https://xbrlsite.azurewebsites.net/2020/Library/EssenceOfAccounting.pdf>

⁶ Charles Hoffman, CPA, *Essentials of XBRL-based Digital Financial Reporting*,
http://www.xbrlsite.com/seattlemethod/platinum/EssentialsOfXBRL_PLATINUM.pdf

Financial Report Logical System

A financial report tends to be viewed as being published in the form of a document, information is “presented” and as you know, different people have different preferences for exactly how they want to see that information presented. Effectively, presentation of information is arbitrary and dependent on the preferences of the presenter of that information.

However, for our purposes we are going to need you to look at how the logic of the information within such a report is represented so that you can see the logic, understand that logic, and then also understand how to leverage that representation logic as somewhat of a “semantic glue” that holds the information representation together.

Further, think of the presentation of that information as a two-step process as contrast to a one step process. Step one is the logical representation. Then step two is taking that logical representation from step one and converting that logical representation into a presentation. Given the fact that presentations are arbitrary and that no matter how we present information; someone is going to want to be able to present something differently; we are not going to address presentation at all in this document other than the one “neutral” presentation that simply conveys the logic of the information logically to a reader such that they have a clear understanding of the logic of the information that is being conveyed.

We will use common practices and good practices in one specific way. For example, accountants use “underscores” and “double underscores” to present subtotals and totals. We will create exactly ONE WAY to present those subtotals and totals (i.e. we are not trying to provide the feature necessary to make every presentation possible).

Focusing on the presentation, other than the extent to see that the information can be clearly understood, is to completely miss the point of this document. We are simply explicitly suggesting that the reader not make that mistake. Focus on the information logic. Focus on how the logic can serve as “glue” and serve other purposes.

Thousands of Logical Pieces

What XBRL fundamentally enables is that it (a) breaks a financial report into thousands or even tens of thousands of individual “atoms” of logical information and (b) the ability to reliably work with those atoms of logical information in order to perform work.

This is as contrast to contemporary financial reports prepared using, say Microsoft Word, which is presentation-oriented information put into a physical document layout model that allows for the presentation of the information in the form of a document, pages, tables, rows, columns, cells, paragraphs and such.

Something to note is that it is very possible to reliably create a presentation from the logical artifacts. It is also possible to convert one standard presentation into logical artifacts. However, it becomes increasingly challenging to convert multiple different arbitrary presentation formats into the logic or meaning of what is being presented.

Atomic Design Methodology

The atomic design methodology⁷ is leveraged to convert the thousands of logical pieces into a workable mechanism to interact with those thousands of logical pieces. Using the terminology of the atomic design methodology; smaller “atoms” which are the very basic building blocks are combined to form “molecules”. Molecules tend to be more tangible and

⁷ Atomic Design Methodology, <https://digitalfinancialreporting.blogspot.com/2023/12/atomic-design-methodology.html>

operational than the smaller atoms. Organisms are assemblies of molecules that function as a logical unit. These organisms tend to be even more tangible and operational; but also, more complex and sophisticated than molecules.

But we will not be using the terminology of the atomic design methodology; we will instead use the terminology of financial reporting and accounting. Some of the terminology is familiar but we also need to create some new terminology in order to refer to this new approach to looking at a financial report.



Model Based Approach

Rater than type rows and columns into a table to present information; we are going to build a model and use that model to generate a table to present information. Software engineers understand this as the “model-view-controller” software design pattern⁸ for constructing user interfaces.

Modeling of Information Meaning

A model-based approach is used to creating financial report models and reports⁹. The models created are models of information logic. That information has certain specific signatures that allow specific logical patterns to be identified and worked with¹⁰. Even though the semantics of these information model patterns are not explicitly specified within an XBRL-based report; prototype theory can be used to identify a semantic object by observing, using software, the pieces of the object(s) in order to derive the semantic objects with certainty.

#	Information Model Pattern (Concept Arrangement Pattern) ¹	XBRL Calculation Relations Exist?	Specific XBRL Formula Pattern Exists?	Member Arrangement Pattern ² Exists?	Specific Report Date Dimension Exists?	Specific Reporting Scenario Dimensions Exist?	Originally Stated Label Role ³ Exists in XBRL Presentation Relations?	Restated Label Role ⁴ Exists in XBRL Presentation Relations?	Period Start Label Role Exists in XBRL Presentation Relations?	Period End Label Role Exists in XBRL Presentation Relations?
1	Set	Never	Never	Optional	Never	Optional	Never	Never	Never	Never
2	Roll Up	Always	Never	Optional	Never	Optional	Never	Never	Never	Never
3	Roll Forward	Never	Always ⁵	Optional	Never	Never	Never	Never	Always	Always
4	Roll Forward Info	Never	Never	Optional	Never	Never	Never	Never	Always	Always
5	Adjustment	Never	Never	Never	Always	Never	Always	Always	Never	Never
6	Variance	Optional	Always ⁶	Always	Never	Always	Never	Never	Never	Never
7	Text Block	Never	Never	Optional	Never	Never	Never	Never	Never	Never
8	Member Aggregation	Optional	Always	Always ⁷	Never	Never	Never	Never	Never	Never
9	Arithmetic	Never	Always ⁸	Optional	Never	Optional	Never	Never	Never	Never

Organism (a.k.a. Block, Infon) Discovery and Identification

Just as the “stuff” in a human body is not just a collection of random things, but rather are specific types of things that can be categorized; the “stuff” in a financial report are likewise sets of types of things that can be organized into categories. Atomic design methodology refers to these things as “organisms”, Situation Theory¹¹ refers to them as “infons”, and the

⁸ Wikipedia, *Model-View-Controller*, <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>

⁹ *Foundational Information Patterns*, <http://xbrlsite.com/seattlemethod/platinum-testcases/FoundationalInformationPatterns.pdf>

¹⁰ *Information Model Identification*, <http://www.xbrlsite.com/mastering/InformationModelIdentification.pdf>

¹¹ *Describing Situation Semantics using Situation Theory*, <https://digitalfinancialreporting.blogspot.com/2023/10/describing-situation-semantics-using.html>

Seattle Method refers to these types of things that can be categorized as “blocks” or “information blocks”.

Organisms, Infons, and Blocks are useful *general* units of logic-oriented information. These logic-oriented units of information within an XBRL based financial report have logical patterns. Using the global standard XBRL technical syntax “signatures” of the physical technical format of these thousands of logical pieces, the logical pieces of the report can be identified and then worked with. For example, a set of reported facts can be identified to be a roll up pattern because of the existence of XBRL calculation relations.

Further, the report logical pattern related information Blocks can be further identified as *specific* financial reporting related artifacts such as the “Disclosures” that have been provided within an XBRL-based digital financial report. For example, a Block of information that has been generally identified to be a Roll Up information pattern can further be identified to be the specific financial disclosure related to the subclassifications of inventory financial disclosure because of the roll up signature and the existence of the concept “mini:Inventories” as the total position within the Roll Up information pattern which is part of the specific financial reporting semantics of that more general logical artifact.

Logical Theory Describing Financial Report

There are many different ways the logic (a.k.a. semantics, meaning) of an XBRL-based report is documented. Two useful resources are the *Logical Theory Describing Financial Report (Terse)*¹² and the *Standard Business Report Model (SBRM)* financial report pieces¹³. Other approaches also exist. For example, XBRL International publishes the *Open Information Model 1.0*¹⁴.

Keep in mind that the documentation cited in this resource was created by a professional accountant, not an expert in UML modeling. Further, XBRL International does not do a particularly good job of explaining financial report logic and SBRM is not yet complete and a publicly available standard.

One of the very best approaches to understanding the logic of a financial report is to observe the XBRL-based reports submitted to the SEC and ESMA. While none of these reports is 100% correct; there are observable patterns in the reports and “right” and “wrong” can be determined by a knowledgeable observer simply observing the reports. Comparing and contrasting how different reporting economic entities represent information are the best clue as to what constitutes a correct or incorrect financial report. When published standards exist then it will be significantly easier to articulate this information. But for now; the definition of what constitutes a good XBRL-based financial report is the accounting standards documentation and reverse engineering existing reports¹⁵.

This is not to say that you must do all this reverse engineering for yourself. If you simply look at the evidence, which tends to be indisputable; you will reach the same conclusions that others have reached.

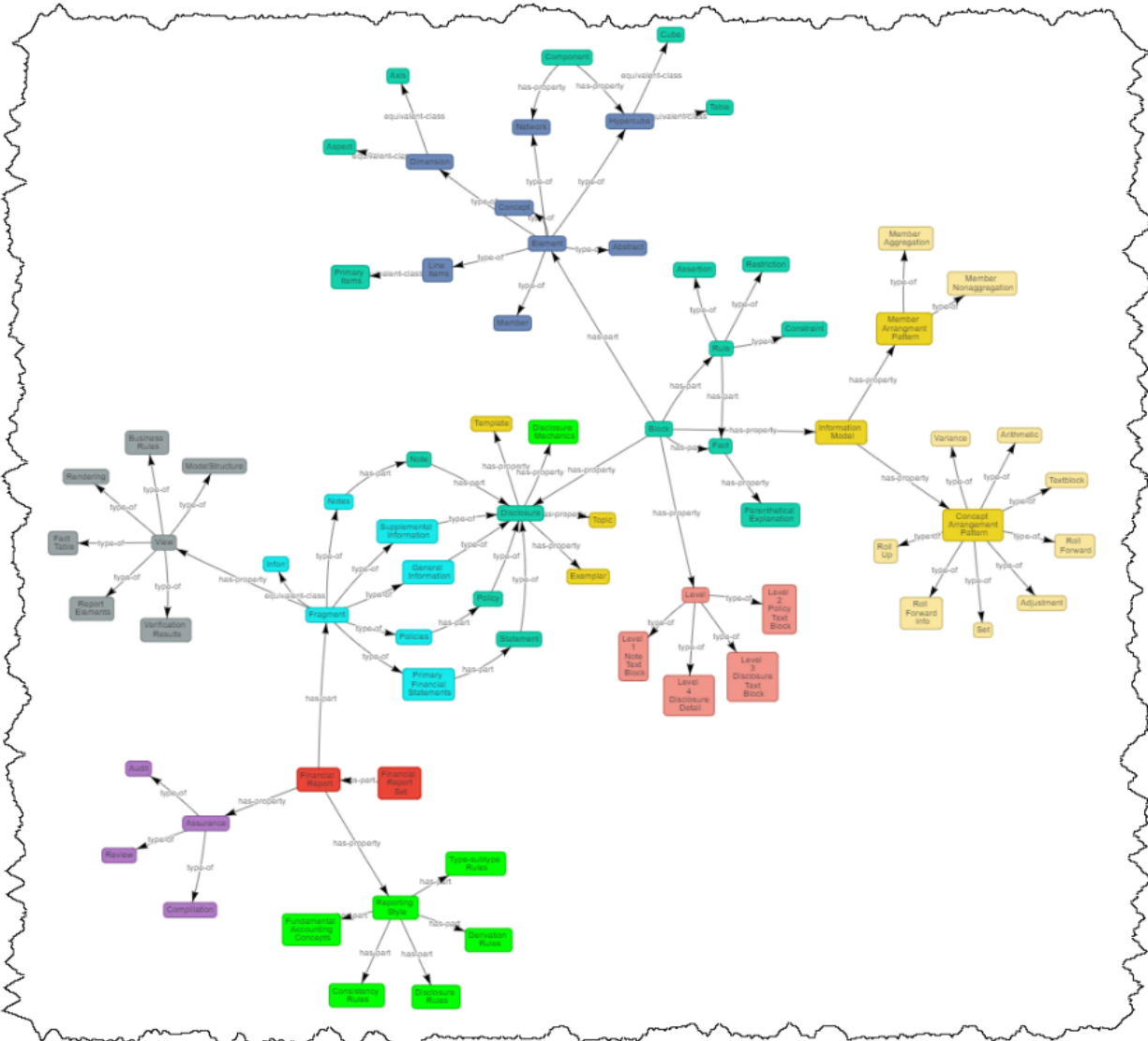
¹² Charles Hoffman, CPA, *Logical Theory Describing Financial Report*,

http://xbrlsite.com/seattlemethod/LogicalTheoryDescribingFinancialReport_Terse.pdf

¹³ Financial Report Pieces, <http://xbrlsite.com/seattlemethod/FinancialReportPieces.html>

¹⁴ XBRL International, *Open Information Model 1.0*, <https://specifications.xbrl.org/work-product-index-open-information-model-open-information-model.html>

¹⁵ *Analysis of 6,751 XBRL-based Public Company 10-Ks Submitted to SEC*, http://www.xbrlsite.com/mastering/Part05_Chapter08.F_AnalysisOf675110Ks.pdf



Seattle Method

The *Seattle Method*¹⁶ is a proven, industrial strength, good practices, standards-based pragmatic approach to creating provably high quality XBRL-based general purpose financial reports when the report creator is permitted to modify the report model explained in simple terms. This approach leverages the “semantic glue”, the logical patterns, of XBRL-based reports.

Seeing the Semantic Glue

And so, we will get to it. Given that all of the semantic glue is represented in the XBRL technical format we are going to leverage that global standard technical format to help explain the semantic glue. XBRL carries the semantics that is being represented. Another way to think about this is to consider the following idea: Suppose the logic represented using the XBRL

¹⁶ *Seattle Method*, <http://xbrlsite.com/seattlemethod/SeattleMethod.pdf>

technical format were represented, instead, using some other different technical format; does that change the meaning (the logic) of what has been represented?

Of course it does not. The logic or meaning or semantics being conveyed in one technical format, say XBRL, does not change if that same information (i.e. the logic) is represented using some other different technical format. The logic is consistent across all technical formats used to represent that logic.

By “semantic glue” we mean the unbreakable logical rules that tie what the atomic design methodology calls “molecules” and “organisms” together logically. How those artifacts are tied together technically is another matter that is not relevant to professional accountants; those details should be, must be, hidden from professional accountants in order for software to be approachable by accounting professionals. Accounting professionals or subject matter experts in any area of knowledge must be able to work at the logic level for that area of knowledge without concerning themselves with the technical implementation details of the technical format a software application might input or output.

But that said, we are going to start the journey of understanding the semantic glue with the technical syntax in order to explain why there is no need for accounting professionals to ever be exposed to that technical format, and in our case the XBRL technical format. A good way of understanding the semantic glue is to understand what can go wrong and use the semantic glue to mitigate those impediments to a properly functioning financial report¹⁷.

Physical Format of Logical System: Global Standard XBRL

We are starting with the physical format of the logical system, in this case the global standard XBRL technical syntax, to ground the explanation of the system logic in something that software engineers can get their heads around easily.

The XBRL standard¹⁸ is made up of several different technical specifications¹⁹. We will not provide a primer here for the XBRL standard, there are other sources for that. What we will point out is the important detail that every XBRL specification has a published conformance suite that software vendors building software can, and should, use to make sure that software’s use of XBRL is (a) consistent with what was intended and (b) interoperable with other software implementation of that same standard.

The physical technical format of each specification used can be 99.99966% correct, which is sigma level 6²⁰, because of the high quality of the published conformance suites provided by XBRL International. The conformance suites are 100% automatable, meaning that 100% of the testing of the XBRL technical syntax conformance can be handled by software using completely automated processes.

There are two critically important details that can be understood by understanding the paragraph above:

1. The XBRL technical syntax can be very interoperable between software applications if the creators of software are aware of and make use of the published XBRL International conformance suites that were published precisely for this purpose.
2. The financial report logic contained within and expressed using such XBRL-based formatted financial reports can, likewise, be completely automated to the extent that

¹⁷ *Understanding What Can Go Wrong*, <http://xbrlsite.com/seattlemethod/platinum-testcases/UnderstandingWhatCanGoWrong.pdf>

¹⁸ XBRL International, *The XBRL Standard*, <https://specifications.xbrl.org/>

¹⁹ XBRL International, *XBRL Specifications*, <https://specifications.xbrl.org/specifications.html>

²⁰ Wikipedia, *Six Sigma, Sigma Levels*, https://en.wikipedia.org/wiki/Six_Sigma#Sigma_levels

such report logic is (a) identified and (b) published including the conformance suites which test software implementing that specific financial report logic.

To reiterate, this is precisely what the *Seattle Method*²¹ and *Standard Business Report Model* (SBRM)²² do, or should do. The objective is to have the same level of reliability from software at the report metamodel level that XBRL International provides for the XBRL technical format, which is 99.99966% reliable, sigma level six.

This is not to say that every aspect or every detail of an XBRL-based digital financial report is 100% correct. It is to say that for each specific area where problems are known to occur, specific undisputable rules are provided that make sure software follows those specific report model logic rules. This logic tends to be impossible to dispute because the logic is either obvious or driven by the logic of agreed upon domains such as mathematics, financial reporting logic, set theory, and such.

To understand what is being explained above will be easier by looking at the specific logical categories we are describing and using which we will do now.

Math Rules

If a report has an obvious mathematical computation, that mathematical computation should be provided to the user of the report model for two reasons. First, to document the fact that the relations exist. Second, to document that the math rules have been checked to make sure the math adds up.

Here is an example of this flaw where the XBRL calculation relations are not provided with this report model²³:

Concept [Aspect]	Period [Aspect]	
	2019-12-31	2018-12-31
Land	1,147,000	5,347,000
Buildings, Net	366,375,000	244,508,000
Furnitures and Fixtures, Net	34,457,000	34,457,000
Computer Equipment, Net	5,313,000	4,169,000
Other Property, Plant and Equipment, Net	6,149,000	6,702,000
Property, Plant and Equipment, Net	413,441,000	295,183,000

For contrast, here is a correctly represented report²⁴. Notice that above you see no single and double underscore indicating a roll up whereas below you do see the single underscore and double underscore:

²¹ Charles Hoffman, CPA, *Seattle Method*, <http://xbrlsite.com/seattlemethod/SeattleMethod.pdf>

²² OMG, *Standard Business Report Model* (SBRM), <https://www.omg.org/intro/SBRM.pdf>

²³ Hello World No Math rules, <http://xbrlsite.com/seattlemethod/platinum-testcases/helloworld-nomath/>

²⁴ Hello World with math rules, <http://xbrlsite.com/seattlemethod/platinum-testcases/helloworld/>

Concept [Aspect]	Period [Aspect]	
	2019-12-31	2018-12-31
Land	\$ 1,147,000	\$ 5,347,000
Buildings, Net	366,375,000	244,508,000
Furnitures and Fixtures, Net	34,457,000	34,457,000
Computer Equipment, Net	5,313,000	4,169,000
Other Property, Plant and Equipment, Net	6,149,000	6,702,000
Property, Plant and Equipment, Net	\$ 413,441,000	\$ 295,183,000

Finally, while the example above is for the math relations of a roll up type of mathematical computation; the same thing is true for every core mathematical computation: roll ups, roll forwards; other forms of arithmetic, restatement, variance, dimensional roll up, and any other pattern of mathematics logic. Simply leaving out this important part of a report model representation is hard to justify.

Model Structure Rules

The structure of a report model is defined by the XBRL presentation relations, calculation relations, definition relations, and XBRL formulas that describe the report. All of this structure is enforced by the XBRL technical specifications and therefore the XBRL International published conformance suites described in the prior section, except for one missing detail.

XBRL presentation relations do not specify rules for what type of report elements can be related to what other type(s) of report elements within the XBRL presentation relations.

In XBRL, there are exactly the following types of report elements: Network, Hypercube (a.k.a. Table), Dimension (a.k.a. Axis), Member, PrimaryItems (a.k.a. LineItems), Abstract, and Concept.

There are very obvious pathological relations between a type of report element that is a parent in a relationship and a type of report element that is a child in that same relationship. There are other sorts of such relations which may be disputable. But what is crystal clear is that XBRL International does not publish the following table which describes one example of a clear documentation of what is and what is not allowed:

		Parent						
		Network	Hypercube	Dimension	Member	LineItems	Abstract	Concept
Child	Network	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL	Illegal XBRL
	Hypercube	Permitted	Disallowed	Disallowed	Disallowed	Disallowed	Permitted	Disallowed
	Dimension	Disallowed	Permitted	Disallowed	Disallowed	Disallowed	Disallowed	Disallowed
	Member	Disallowed	Disallowed	Permitted	Permitted	Disallowed	Disallowed	Disallowed
	LineItems	Disallowed	Permitted	Disallowed	Disallowed	Disallowed	Disallowed	Disallowed
	Abstract	Permitted	Disallowed	Disallowed	Disallowed	Permitted	Permitted	Disallowed
	Concept	Disallowed	Disallowed	Disallowed	Disallowed	Permitted	Permitted	Disallowed

That table above is only readable by humans. But that same information can, and has been, articulated using XBRL definition relations by the *Seattle Method*²⁵. The benefit of expressing these rules using XBRL definition relations is that if someone disputes the above table; they

²⁵ Seattle Method, Model Structure Rules expressed using XBRL definition relations, <http://xbrlsite.com/seattlemethod/cm/model-structure-rules-strict-def.xml>

can simply tweak the XBRL definition relations to represent what they believe to be correct (given that XBRL International nor any other authority publish this information).

An example will help the reader understand why these rules are important. The following is a pathological example that makes the point. In the model structure below, you see a model structure flaw²⁶:

Label	Category	Data Type	Period	Balance	PreferredLabelRole	Name
Property, Plant and Equipment, Net [Roll Up]	Abstract					report:PropertyPlantAndEquipmentNetRollUp
Land	Concept	Monetary	Instant	Debit		report:Land
Buildings, Net	Concept	Monetary	Instant	Debit		report:BuildingsNet
Furnitures and Fixtures, Net	Concept	Monetary	Instant	Debit		report:FurnitureAndFixturesNet
Computer Equipment, Net	Concept	Monetary	Instant	Debit		report:ComputerEquipmentNet
Other Property, Plant and Equipment, Net	Concept	Monetary	Instant	Debit		report:OtherPropertyPlantAndEquipmentNet
Property, Plant and Equipment, Net	Concept	Monetary	Instant	Debit		report:PropertyPlantAndEquipmentNet

This can be a little hard to see in the above representation, but the relationships between the line items of property, plant, and equipment are represented pathologically as children of each other. Software can overcome this modeling flaw as can be seen below:

Reporting Entity [Aspect]		SAMP http://www.
Unit [Aspect]		iso4217:USD
Concept [Aspect]	Period [Aspect]	
	2020-12-31	
Land	\$	5,347,000
Buildings, Net	\$	244,508,000
Furnitures and Fixtures, Net	\$	34,457,000
Computer Equipment, Net	\$	4,169,000
Other Property, Plant and Equipment, Net	\$	6,702,000
Property, Plant and Equipment, Net	\$	295,183,000

To be crystal clear; there is nothing in the XBRL technical specification that prevents the pathological representation that is shown above.

By way of contrast, this is a correct modeling of the same information²⁷:

Label	Category	Data Type	Period	Balance	PreferredLabelRole	Name
Property, Plant and Equipment, Net [Roll Up]	Abstract					helloWorld:PropertyPlantAndEquipmentNetRollUp
Land	Concept	Monetary	Instant	Debit		helloWorld:Land
Buildings, Net	Concept	Monetary	Instant	Debit		helloWorld:BuildingsNet
Furnitures and Fixtures, Net	Concept	Monetary	Instant	Debit		helloWorld:FurnitureAndFixturesNet
Computer Equipment, Net	Concept	Monetary	Instant	Debit		helloWorld:ComputerEquipmentNet
Other Property, Plant and Equipment, Net	Concept	Monetary	Instant	Debit		helloWorld:OtherPropertyPlantAndEquipmentNet
Property, Plant and Equipment, Net	Concept	Monetary	Instant	Debit		helloWorld:PropertyPlantAndEquipmentNet

Reporting Entity [Aspect]		SAMP http://www.reportingscheme.c
Unit [Aspect]		iso4217:USD
Concept [Aspect]	Period [Aspect]	
	2019-12-31	2018-12-31
	Land	\$ 1,147,000
Buildings, Net	366,375,000	244,508,000
Furnitures and Fixtures, Net	34,457,000	34,457,000
Computer Equipment, Net	5,313,000	4,169,000
Other Property, Plant and Equipment, Net	6,149,000	6,702,000
Property, Plant and Equipment, Net	\$ 413,441,000	\$ 295,183,000

²⁶ Model Structure Flaw, <http://xbrlsite.com/seattlemethod/platinum-testcases/model-structure-flaw/>

²⁷ Hello World with proper model structure, <http://xbrlsite.com/seattlemethod/platinum-testcases/helloworld/>

A very, very high percentage of entities creating XBRL-based reports represent roll ups in those reports as can be seen above, a flat list of. But not 100%. So, what exactly does it mean within XBRL presentation relations when a concept represented as a child of another parent concept? And how exactly is that parent-child relations different in terms of semantics from two siblings of the same parent concept?

Sure, each individual person could have their own interpretation of these relations, and they do. But that is the point; there should not be a need for an interpretation, this should be clearly specified so that all these cases are clear for all reports.

Fundamental High-level Continuity Crosscheck Rules

The financial accounting and reporting logic of a financial report is defined by the report model of that report. A report model must be consistent with the fundamental concepts and relationships of the financial reporting scheme used to create that financial report. Further, each logical fragment of a report must not contradict some other logical report fragment.

A very high level example will help the reader understand the notion of fundamental high level concepts and continuity crosschecks. Consider the fundamental high-level rule of financial accounting conveyed by the accounting equation²⁸: Assets = Liabilities + Equity.

Here you see a very basic financial report where the facts reported to not follow the accounting equation²⁹:

Concept [Aspect]	Period [Aspect]	
	2020-12-31	
Assets	\$	8,000
Liabilities		1,000
Equity		4,000

If you take a close look at the report, you see that Assets does not equal the sum of Liabilities and Equity. If only XBRL verification were used to check this report, the report would seem valid because XBRL does not know about the accounting equation. However, if a universal fundamental relationship were published, that "Assets = Liabilities + Equity"; then the report could be checked to make sure the fundamental relationship between these concepts where consistent with what would be expected.

And here you see software doing exactly that. Notice that, because of the machine-readable rule, software can detect and therefore report this problem to the user of the software:

Concept [Aspect]	Period [Aspect]	
	2020-12-31	
Assets	ⓘ	8000
Liabilities		1000
Equity		4000

Here you see a report that is consistent with what would be expected³⁰:

²⁸ Wikipedia, *Accounting Equation*, https://en.wikipedia.org/wiki/Accounting_equation

²⁹ Accounting Equation, State 3, <http://xbrlsite.com/seattlemethod/platinum-testcases/ae-state3/>

³⁰ Accounting Equation, State 1, <http://xbrlsite.com/seattlemethod/platinum-testcases/ae-state1/>

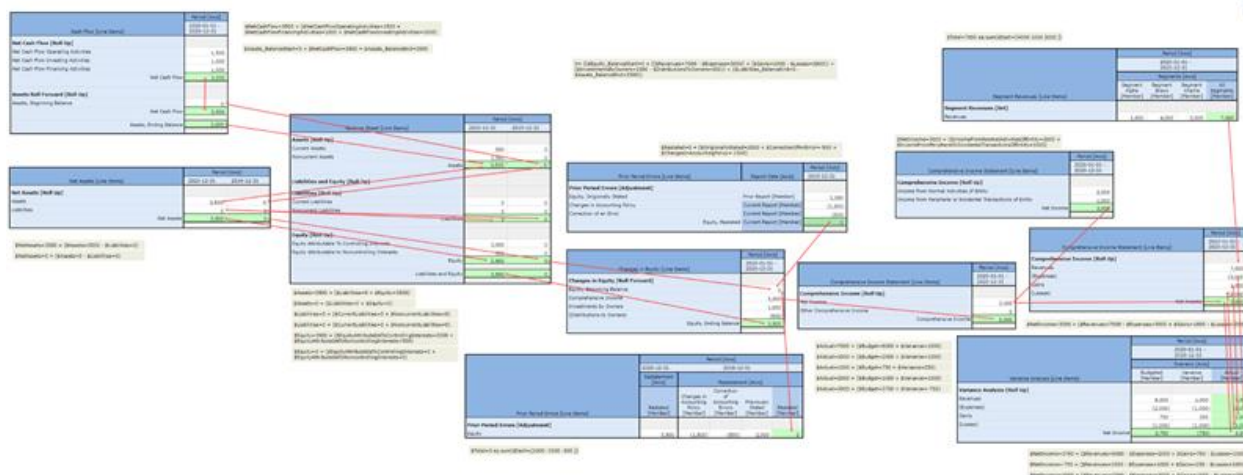
Concept [Aspect]	Period [Aspect]	
	2020-12-31	
Assets	\$	5,000
Liabilities		1,000
Equity		4,000

The accounting equation is only tip of a much bigger iceberg. There are many other fundamental high-level relationships which must be followed within financial reports³¹:

- Assets = Current Assets + Noncurrent Assets
- Liabilities = Current Liabilities + Noncurrent Liabilities
- Equity = Equity Attributable to Parent + Equity Attributable to Noncontrolling Interests
- Net Cash Flow = Net Cash Flow Operating + Net Cash Flow Investing + Net Cash Flow Financing

These are but a very few fundamental concepts and their relations, there are many, many more. These concepts and their relations differ between different financial reporting schemes. But every financial reporting scheme has the notion of these fundamental high-level accounting concepts and relations between the concepts.

Further, the idea of articulation³² helps you understand that these concepts are interrelated that that the relationships exist no matter where a concept is used within the financial report³³:



It is these high level financial concepts that tend to be the intersections between the different disclosures or blocks of information contained within a financial report.

The fundamental high level concepts also serve another purpose which will become clear to you in the next section of rules.

³¹ *Fundamental Accounting Concepts and Relations*, US GAAP, http://accounting.auditchain.finance/fac/fac_ModelStructure.html

³² *Understanding Articulation*, <https://digitalfinancialreporting.blogspot.com/2023/08/understanding-articulation.html>

³³ PROOF Articulation, https://www.xbrlsite.com/seattlemethod/platinum/proof/PROOF_Articulation.jpg

To solidify your understanding, here is a full report of the fundamental accounting concepts continuity cross checks for the Microsoft 10-K³⁴:

Balance Sheet [Line Items]	Period [Axis]	
	2017-06-30	
	Value	Fact
Assets [Roll Up]		
Current Assets	159,851,000,000	fac:CurrentAssets[us-gaap:AssetsCurrent[159,851,000,000]]
Noncurrent Assets	81,235,000,000	fac:NoncurrentAssets[81,235,000,000] = fac:Assets[us-gaap:Assets[241,086,000,000]] - fac:CurrentAssets[us-gaap:AssetsCurrent[159,851,000,000]]
Assets	241,086,000,000	fac:Assets[us-gaap:Assets[241,086,000,000]]
Liabilities and Equity [Roll Up]		
Liabilities [Roll Up]		
Current Liabilities	64,527,000,000	fac:CurrentLiabilities[us-gaap:LiabilitiesCurrent[64,527,000,000]]
Noncurrent Liabilities	104,165,000,000	fac:NoncurrentLiabilities[104,165,000,000] = fac:Liabilities[us-gaap:Liabilities[168,692,000,000]] - fac:CurrentLiabilities[us-gaap:LiabilitiesCurrent[64,527,000,000]]
Liabilities	168,692,000,000	fac:Liabilities[us-gaap:Liabilities[168,692,000,000]]
Commitments and Contingencies	0	fac:CommitmentsAndContingencies[us-gaap:CommitmentsAndContingencies[0]]
Temporary Equity	0	fac:TemporaryEquity[0] = fac:LiabilitiesAndEquity[us-gaap:LiabilitiesAndStockholdersEquity[241,086,000,000]] - ((fac:Liabilities[us-gaap:Liabilities[168,692,000,000]] + fac:Equity[72,394,000,000]) - fac:CommitmentsAndContingencies[us-gaap:CommitmentsAndContingencies[0]])
Equity [Roll Up]		
Equity Attributable to Parent	72,394,000,000	fac:EquityAttributableToParent[us-gaap:StockholdersEquity[72,394,000,000]]
Equity Attributable to Noncontrolling Interest	0	fac:EquityAttributableToNoncontrollingInterest[0] = fac:Equity[72,394,000,000] - fac:EquityAttributableToParent[us-gaap:StockholdersEquity[72,394,000,000]]
Equity	72,394,000,000	fac:Equity[72,394,000,000] = fac:EquityAttributableToParent[us-gaap:StockholdersEquity[72,394,000,000]]
Liabilities and Equity	241,086,000,000	fac:LiabilitiesAndEquity[us-gaap:LiabilitiesAndStockholdersEquity[241,086,000,000]]

Type and Subtype Rules

The next layer of financial reporting concepts can be checked using what we refer to as “type-subtype” relations but can also be referred to as “wider-narrower” or “general-special” or “class-subclass” relations.

Consider the type “Current Assets”. That type might have subtypes “Cash and Cash Equivalents”, “Trade Receivables”, “Inventories”, “Prepaid Expenses”, and other such types of Current Assets.

By way of contrast, the concept “Net Income (Loss)” is not a type of Current Asset. That is an obvious example. There are other situations where this information is not as clear. This is where “type-subtype” relations are helpful.

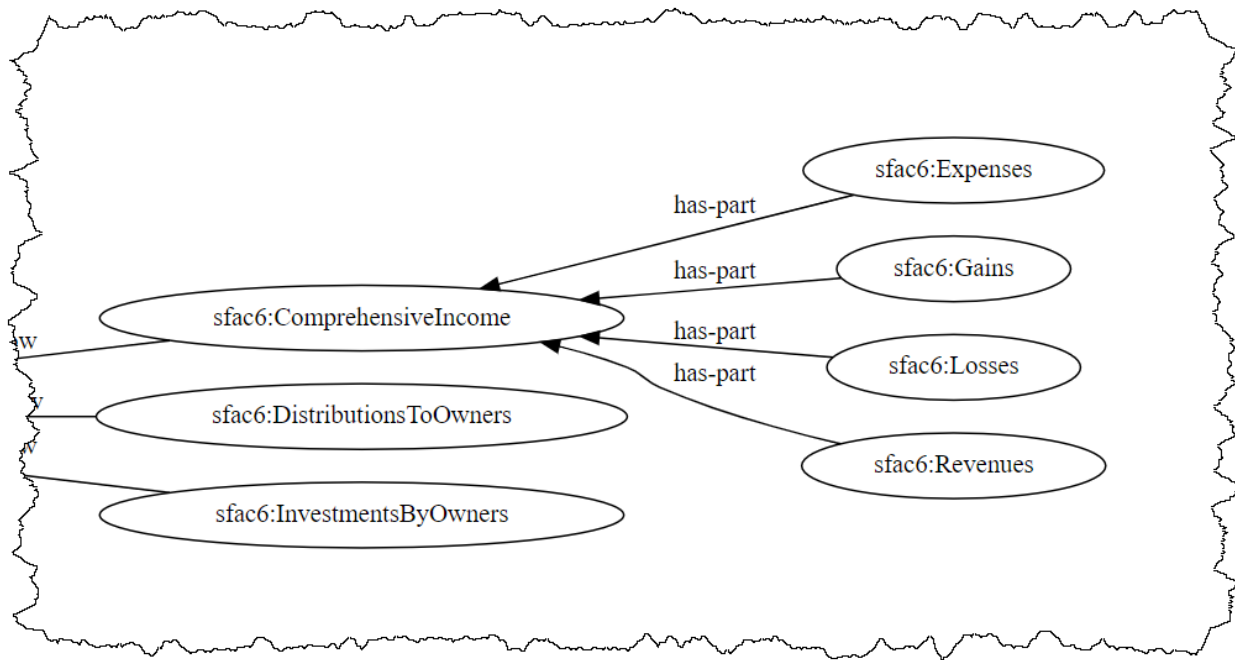
Consider this report example below where the line item “Investments by Owners” was inadvertently added to the income statement which is an obvious mistake³⁵:

³⁴ Microsoft, *Fundamental Accounting Concepts Continuity Cross Checks*, <https://xbrlsite.azurewebsites.net/2017/Prototypes/Microsoft2017/evidence-package/USFACRenderingSummary.html>

³⁵ Type-subtype Error, <https://www.xbrlsite.com/site1/seattlemethod/platinum-testcases/type-subtype-error/index.html>

Concept [Aspect]	Period [Aspect]	
	2023-01-01 2023-12-31	
Comprehensive Income [Roll Up]		
Revenues	\$	5,000
Investments by Owners		1,000
(Expenses)		(4,000)
Gains		1,000
(Losses)		(1,000)
Comprehensive Income	\$	2,000

The type-subtype rules indicate that Revenues, Expenses, Gains, and Losses are permitted to be part of Comprehensive Income; Investments by Owners is NOT part of Comprehensive Income³⁶:



The type-subtype rules detect these sorts of reporting violations. This is a rather obvious example; but you may be able to imagine that with a taxonomy like the US GAAP XBRL Taxonomy with about 20,000 concepts that it can be challenging to not make a mistake when you are modeling thousands of report line items. This is why type-subtype relations come in handy.

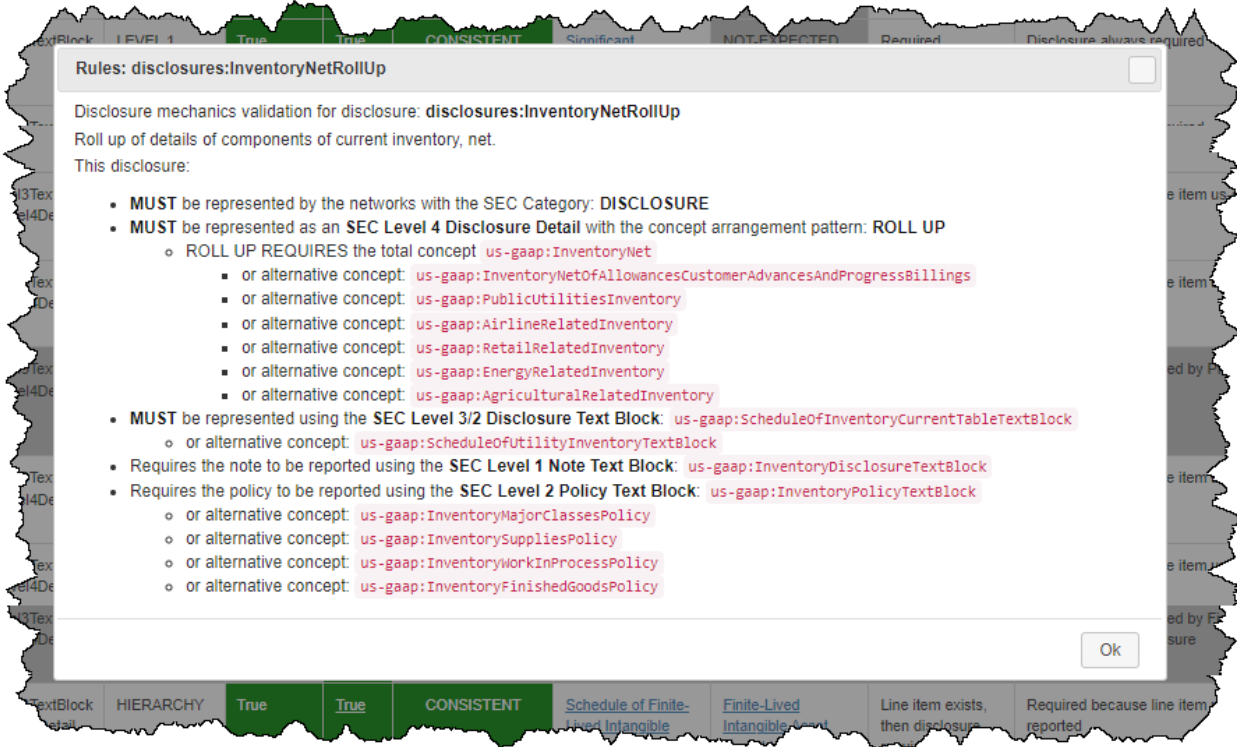
Disclosure Mechanics Rules

We mentioned that you can discover the general information Blocks, the organisms, within a financial report. But how do you know what that information Block is representing? Is it a

³⁶ Type-subtype rules, <https://auditchain.infura-ipfs.io/ipfs/QmP8U8EJRv4aocFP6xXGMRBniP7ghTyTMdQ89hc3hE1fYn/typeSubTypeGraph.html>

balance sheet? An income statements? The long-term debt maturities disclosure? The effective tax rate disclosure.

Financial reporting schemes don't tend to give specific names to disclosures. But what if they did? Well, I did exactly that. For my prototype financial reporting schemes and for a prototype of US GAAP disclosures, I gave each disclosure a unique name³⁷. Then I described the essence of the disclosure³⁸ in machine-readable form using XBRL. Here is an example of the rules for one US GAAP disclosure, a disaggregation of the components of inventories, which were represented using XBRL definition relations³⁹ and then rendered into human readable form:



This same thing can be done for ever disclosure. This allows each information Block in a report, each organism, to be identified as a specific financial disclosure. Then, these descriptions of the mechanics of a disclosure can be used to determine if a disclosure exists within a financial report.

Below you see a representation of the disclosure mechanics rules⁴⁰ for the SFAC 8 Financial Reporting Scheme⁴¹:

³⁷ US GAAP Financial Reporting Scheme (Prototype), Disclosures, <http://accounting.auditchain.finance/reporting-scheme/us-gaap/documentation/Disclosures.html>

³⁸ Inventory roll up disclosure, human readable HTML, <https://xbrlsite.azurewebsites.net/2020/reporting-scheme/us-gaap/disclosures-topics/disclosures-detail/Disclosure-517.html>

³⁹ Inventory roll up disclosure, XBRL definition relations, <https://xbrlsite.azurewebsites.net/2020/reporting-scheme/us-gaap/disclosure-mechanics/517-rules-def.xml>

⁴⁰ Disclosure Mechanics Rules, SFAC 8, <https://auditchain.infura-ipfs.io/ipfs/QmeNs1iW3bvrp5YPkyEzEWJqaBjDnJQiAQcbXfyrc4rjPR/disclosures.html>

⁴¹ SFAC 8 Financial Reporting Scheme, http://www.xbrlsite.com/seattlemethod/golden/sfac8/sfac8_ModelStructure.html

#	Type	Name	Rule Expression
1	disclosure	disclosures:BalanceSheet Added • detections:1	<i>Balance Sheet</i> • disclosures:BalanceSheet requires: • Concept Arrangement Pattern cm:Arithmetic ◦ with sfac8:Assets • Concept sfac8:Liabilities • Concept sfac8:Equity <i>1 instance:</i> <i>In network 11-Statement of Financial Position:</i> sfac8:Liabilities is presented sfac8:Equity is presented Detected block Assets [Arithmetic] with sfac8:Assets
2	disclosure	disclosures:ChangesInEquity Added • detections:1	<i>Changes in Equity</i> • disclosures:ChangesInEquity requires: • Concept Arrangement Pattern cm:RollForward ◦ with sfac8:Equity <i>1 instance:</i> <i>In network 31-Statement of Changes in Equity:</i> Detected block Equity, Beginning Balance [RollForward] with sfac8:Equity
3	disclosure	disclosures:ChangesInFundBalance Added • detections:0	<i>Changes in Fund Balance</i> • disclosures:ChangesInFundBalance requires: • Concept Arrangement Pattern cm:RollForward ◦ with sfac8:FundBalance
4	disclosure	disclosures:ChangesInNetAssets Added • detections:0	<i>Changes in Net Assets</i> • disclosures:ChangesInNetAssets requires:

These disclosure mechanics rules are not directly used to verify a financial report; rather they are used in conjunction with the next verification category, the reporting checklist rules.

Reporting Checklist Rules

Every financial reporting scheme has a set of rules regarding when a disclosure must appear in a financial report. For example, some disclosures such as a balance sheet, income statement, statement of changes in equity, and cash flow statement are always required.

Other disclosures are required when certain line items appear on a financial statement. For example, if the "Inventories" line item exists, then the "Inventories subclassifications" and the "Inventories Policy" are required to be disclosed.

The reporting checklist rules enables many, but not all, of what might be found in a manual reporting checklist memory jogger to be performed using automated.

The reporting checklist rules are represented within an XBRL definition linkbase⁴². That information can be viewed in either human readable or machine-readable form. Then, the reporting checklist along with the disclosure mechanics rules which are used to find the disclosures within a report can be used to determine if what is found within the report is consistent with what the financial reporting rules in the reporting checklist specify⁴³:

⁴² Reporting checklist rules represented within XBRL definition linkbase, <http://xbrlsite.com/seattlemethod/golden/sfac8/dr-rules-def.xml>

⁴³ Reporting checklist human readable results, <https://auditchain.infura-ipfs.io/ipfs/QmTH2iYTSWStdh5kXbadA1T9rPjAE7Z5qLF8L4wxgdhxig/disclosureChecks.html>

#	Type	Name	Rule Expression
1	disclosureCheck	require [disclosures:BalanceSheet] Added <ul style="list-style-type: none"> ok:1 failed:0 	Require disclosure: <ul style="list-style-type: none"> disclosures:BalanceSheet 1 instance: See link above
2	disclosureCheck	require [disclosures:ChangesInEquity] Added <ul style="list-style-type: none"> ok:1 failed:0 	Require disclosure: <ul style="list-style-type: none"> disclosures:ChangesInEquity 1 instance: See link above
3	disclosureCheck	require [disclosures:ChangesInFundBalance] Added <ul style="list-style-type: none"> ok:1 failed:0 	Require disclosure: <ul style="list-style-type: none"> disclosures:ChangesInFundBalance 1 instance: See link above
4	disclosureCheck	require [disclosures:ChangesInNetAssets] Added <ul style="list-style-type: none"> ok:1 failed:0 	Require disclosure: <ul style="list-style-type: none"> disclosures:ChangesInNetAssets 1 instance: See link above
5	disclosureCheck	require [disclosures:ComprehensiveIncome] Added <ul style="list-style-type: none"> ok:1 failed:0 	Require disclosure: <ul style="list-style-type: none"> disclosures:ComprehensiveIncome 1 instance: See link above

Other Reporting System Rules

Reporting systems may specify other additional rules which may need to be satisfied when creating financial reports. For example, financial reports submitted to the U.S. Securities and Exchange Commission (SEC) must also comply with the Edgar Filer Manual⁴⁴ rules. Also, XBRL US Data Quality Committee rules⁴⁵ may be used to verify reports.

Other financial reporting schemes of filing systems might specify completely different rules. For example, financial reports submitted to the European Single Market Authority (ESMA) using the European Single Electronic Format (ESEF) format must comply with the ESEF Reporting Manual rules⁴⁶.

These rules may only apply when financial reports are serialized to the XBRL global standard technical syntax format.

⁴⁴ SEC, Edgar Filer Manual, <https://www.sec.gov/edgar/filermanual>










⁴⁵ XBRL US Data Quality Committee Rules, <https://xbrl.us/data-quality/rules-guidance/>

⁴⁶ ESMA, ESEF Reporting Manual, <https://www.esma.europa.eu/document/esef-reporting-manual>

Other rules may apply when Inline XBRL reports are created. Going into further detail with respect to these various rules categories are beyond the scope of this document.

Example of Report and Semantic Glue

An example report that a software engineer can reverse engineer can help you understand the semantic glue that has been discussed within this section. For this purpose, we will use the PROOF reference implementation files⁴⁷ including the report model and report⁴⁸ along with the report model and report verification results⁴⁹.

#	Verification Category	Result
1	XBRL Technical Syntax Verification	
2	Report Mathematical Computations Verification (XBRL Calculations)	
3	Report Mathematical Computations Verification (XBRL Formulas)	
4	Report Model Structure Verification	
5	Fundamental Accounting Concept Consistency Crosschecks Verification	
6	Type-subtype (wider-narrower) Associations Verification	
7	Disclosure Mechanics Verification	
8	Report Disclosure Checklist Verification	
9	Other	

The following are helpful reverse engineering tips to help the reader understand what they are looking at:

1. Start with the report, the XBRL instance. From the XBRL instance, XBRL rules require a physical connection to every XBRL file used by the XBRL instance.
2. Additional XBRL files can be appended to a report for verification or physically connected to the report. In our case, the AuditChain Pacioli verification report provides a summary of additional semantics that were added to the XBRL based report and report model.
3. These were the additional verification files specifically added to this report: ['http://xbrlsite.com/seattlemethod/cm/model-structure-rules-strict-def.xml', 'http://www.xbrlsite.com/seattlemethod/platinum/proof/disclosure-mechanics/disclosure-mechanics.xsd', 'http://www.xbrlsite.com/seattlemethod/platinum/proof/fac/reporting-styles/BSC-IS01-CF1_schema.xsd', 'http://www.xbrlsite.com/seattlemethod/platinum/proof/reporting-checklist/reporting-checklist-rules-def.xml', 'http://www.xbrlsite.com/seattlemethod/platinum/proof/type-subtype/type-subtype.xsd']

⁴⁷ PROOF reference implementation files, <http://www.xbrlsite.com/seattlemethod/platinum/proof/ref/index.html>

⁴⁸ PROOF report model and report, <https://suite.auditchain.finance/storage/395dfa84-e4e4-11ec-8fea-0242ac120002/uwDPU5Bvy/instance.xml>

⁴⁹ PROOF report model and report verification results, <https://auditchain.infura-ipfs.io/ipfs/QmVdn6akCxSxB7yKb94qTFkG46UY4sNQPVRYQ9eyVC5eLK/>

4. Note that while the technical files are really of no use to accountants reviewing reports, the information provided by the files is very important. Organizing this information within a GUI/UX is key to making XBRL-based reports approachable to accounting professionals.

And so, this ends our explanation of the semantic glue available. The information in this section can help software developers understand this semantic glue. All this should just “happen” for accounting professionals using these reports.

The next section will provide additional information that may be helpful in understanding the semantic glue by explaining how that semantic glue can be leveraged by software applications.

Leveraging the Semantic Glue

This section explains how the semantic glue that was pointed out in the previous section can be leveraged within software applications to assist accounting professionals using that software to construct high quality financial report models.

Providing this functionally within software assures that the core report logic of a financial report is always correct. That lets the accounting professionals making use of that software focus on representing information within that report model as contrast to having to work to get that financial report model logic fundamentally correct. Basically, software will not need to verify the core report logic of a report model to see if it is right because the software will never let them get that report model logic wrong.

Specific examples will explain which will be provided next. But fundamentally, three different dynamics are at play:

1. Leveraging an undisputed or even undisputable universal logical conceptualization of a business report metamodel. When over 90%+ of XBRL-based reports follow a certain specific logical pattern, it is very hard to consider that specific logical pattern wrong. (This is either hard coded or some flexibility can be provided using user preferences where deemed necessary.)
2. Well understood and useful common conventions or best practices used within an area of knowledge. (These can be either hard coded or some flexibility can be provided using user preferences where deemed necessary.)
3. Precise flexibility to work with your specific report model or report. (These are completely flexibility within the permitted “boundaries” or “guardrails” or “bumpers” provided in #1 and #2.)

While it can be hard for a software engineer that does not have knowledge of the area of knowledge they are working in to understand the three points above and while even having conversations with business professionals that are not looking at their own area of knowledge from the appropriate perspective to guide software developers to build the right software; subject matter experts within an area of knowledge that also understand software development principles can help guide software developers effectively.

Leveraging Model Structure Rules for Report Model Creation

The model structure rules are helpful in the process of report model creation and configuration because those rules can be used to filter and structure the actual report model. Permitted associations can be allowed and associations that are not permitted can be enforced by the

software. Consider the example below of an information Block which is a roll up pattern of mathematical logic:

Reporting Entity [Aspect]	SAMP http://www.reportingscheme.c		
Unit [Aspect]	iso4217:USD		
	Period [Aspect]		
	2019-12-31	2018-12-31	
Land	\$ 1,147,000	\$ 5,347,000	
Buildings, Net	366,375,000	244,508,000	
Furnitures and Fixtures, Net	34,457,000	34,457,000	
Computer Equipment, Net	5,313,000	4,169,000	
Other Property, Plant and Equipment, Net	6,149,000	6,702,000	
Property, Plant and Equipment, Net	\$ 413,441,000	\$ 295,183,000	

Consider the following:

- The abstract container “Property, Plant and Equipment, Net [Roll Up]” is placed in the report model to serve as an explicit container for the roll up logic.
- It is a known fact that every roll up has exactly one total concept. That concept has a specific period type, “instant” which is as of a specific date or “duration” which is for a specific period. It is never the case that instants and durations are intermingled within a roll up; a roll up is either one or the other. That information can be used to filter concepts.
- What would the logic be if a concept were a child as opposed to a sibling of the other concepts within a roll up? A concept can never be a child of another concept within a roll up; that relationship makes no sense.
- This is not to say that nested roll up logical patterns cannot be supported; they can. But that is a different logical pattern than we are discussing here.
- A roll up logical pattern is different from a roll forward logical pattern and other logical patterns. Software treats roll ups different than roll forwards. Same for other logical patterns as well.

This may seem “simplistic” to an uninformed observer. But that is not what is going on. This is “simple”, not simplistic. It looks simple because it is elegant. The technical complexities of managing these details are buried within the software. Simple means easy to use. Simplistic means dumbing down a problem to make the problem easier to solve. Nothing is being dumbed down.

Next, we will leverage the mathematics of a logical pattern to help us even more.

Leveraging Math Rules to Verify Report Math

The mathematics rules are helpful in the process of report model creation and configuration also. Let’s continue with our roll up example we used above. Notice the single underscore and double underscore which are used by convention to indicate and differentiate subtotals and totals by accountants in their area of knowledge.

Using the edit view of the same fragment of information from our report, software can help the user see that the roll up properly roles up within the report they are working with for the report fragment the user is currently focused on as is shown below:

Concept [Aspect]	Period [Aspect]	
	2019-12-31	2018-12-31
Land	1147000	5347000
Buildings, Net	366375000	244508000
Furnitures and Fixtures, Net	34457000	34457000
Computer Equipment, Net	5313000	4169000
Other Property, Plant and Equipment, Net	6149000	6702000
Property, Plant and Equipment, Net	✓ 413441000	✓ 295183000

So above you can see, and the user of a software application would be able to see, that there are no mathematics errors in the report fragment they are working with. The roll up “foots”. The same sort of mechanism can be used to see of a report fragment “cross casts” (a.k.a. cross foot):

Concept [Aspect]	Period [Aspect]					
	2022-01-01 2022-12-31					
	Segments [Axis]					
	Segment Alpha [Member]	Segment Bravo [Member]	Segment Charlie [Member]	Segment Delta [Member]	Segment Echo [Member]	All Segments [Member]
Segment Revenues [Set]						
Revenues	0	0	0	0	0	✓ 0
Expenses	0	0	0	0	0	✓ 0
Gains	0	0	0	0	0	✓ 0
Losses	0	0	0	0	0	✓ 0
Comprehensive Income	✓ 0	✓ 0	✓ 0	✓ 0	✓ 0	✓ 0

In addition, this same mechanism can be used to be sure that each and every mathematical computation within a report is correct. Below you see a report that has three report fragments or Blocks of information. The selected information Block was modified to make the Block not foot correctly. That fact is seen in the visualization of the specific information Block; but it is also seen in the summary of all the information Blocks within the report:

Reporting Entity [Aspect]	GH259400TOMPUOLS65II http://standards.i
Unit [Aspect]	iso4217:EUR

Concept [Aspect]	Period [Aspect]
	2022-01-01 2022-12-31
Comprehensive Income [Roll Up]	
Revenues	17000
(Expenses)	3000
Gains	1000
(Losses)	2000
Comprehensive Income	⊖ 3000

⏪ ⏴ ⏵ ⏩

Ne ▾

Networks (3)

- 11-Statement of Financial Position
- ▾ 21-Statement of Comprehensive Income
 - 📄 Comprehensive Income Statement [Hypercube]
- 31-Statement of Changes in Equity

To keep our example simple, we have only three information Blocks. A real report would have perhaps several hundred report fragments and many of those would be mathematical computations (roll ups, roll forwards, restatements perhaps, comparisons of different reporting scenarios, dimensional roll ups such as segment information). Each and every one of these mathematical computations is monitored by the software and using clever software interfaces the information is make know to the software user complements of the mathematics rules of the logic of each information Block.

Not “some” of the report mathematics; 100% of the report mathematics. And if a rule is inadvertently left out of the report, there are mechanisms to detect that fact also.

Leveraging Fundamental High-level Continuity Crosschecks

Continuing on with showing how helpful the report mathematical rules are; let us focus for a moment not on the mathematics of an individual information Block; but rather let us not consider the relations between information Blocks.

To be sure, a report should not contradict itself or be inconsistent. The fundamental accounting concept continuity crosscheck rules help the software user be sure that there are no contradictions or inconsistencies between fragments of a report represented by an information Block. Continuing with a very basic report but one that will allow use to make important points:

Concept [Aspect]	Period [Aspect]	
	2023-12-31	2022-12-31
Assets [Roll Up]		
Current Assets	1	0
Noncurrent Assets	0	0
Assets	1	0
Liabilities and Equity [Roll Up]		
Liabilities [Roll Up]		
Current Liabilities	0	0
Noncurrent Liabilities	0	0
Liabilities	0	0
Equity [Roll Up]		
Equity Attributable To Controlling Interests	0	0
Equity Attributable to Noncontrolling Interests	0	0
Equity	0	0
Liabilities and Equity	0	0

01-Balance Sheet

- Balance Sheet [Hypercube]
- 02-Net Assets
- 03-Income Statement
- 04-Income Statement Alternative
- 05-Comprehensive Income Statement
- 06-Cash Flow Statement
- 07-Changes in Equity

Above you see two information Blocks: an Assets [Roll Up] and a Liabilities and Equity [Roll Up]. Each of the roll ups foot correctly. But the report model is showing three mathematics errors. Why?

There is a fundamental accounting concept continuity cross check rule that specifies that the accounting equation must be complied with: $Assets = Liabilities + Equity$. We changed the Assets [Roll up] to make it still roll up; but that change made the accounting equation rule show that the report was inconsistent with that fundamental rule. Notice the bell in the bottom left-hand corner of the screen with a red bubble with a 3 in the bubble. Clicking on the bubble shows what is going on:

RuleType	Expression	Period	StructureIdentifier	Result
RollUpRule	Net Assets==Assets-Liabilities	2023-12-31	NetAssets	false
ConsistencyRule	\$Assets = (\$LiabilitiesAndEquity)	2023-12-31	BalanceSheet	false
RollForwardRule	\$Assets_BalanceStart + \$NetCashFlow = \$Assets_BalanceEnd	2023-01-01 2023-12-31	CashFlow	false
RollUpRule	Assets==Noncurrent Assets+Current Assets	2023-12-31	BalanceSheet	true
RollUpRule	Liabilities==Noncurrent Liabilities+Current Liabilities	2023-12-31	BalanceSheet	true
RollUpRule	Equity==Equity Attributable to Noncontrolling Interests+Equity Attributable To Controlling Interests	2023-12-31	BalanceSheet	true
RollUpRule	Liabilities and Equity==Liabilities+Equity	2023-12-31	BalanceSheet	true
RollUpRule	Assets==Noncurrent Assets+Current Assets	2022-12-31	BalanceSheet	true
RollUpRule	Liabilities==Noncurrent Liabilities+Current Liabilities	2022-12-31	BalanceSheet	true
RollUpRule	Equity==Equity Attributable to Noncontrolling Interests+Equity Attributable To Controlling Interests	2022-12-31	BalanceSheet	true
RollUpRule	Liabilities and Equity==Liabilities+Equity	2022-12-31	BalanceSheet	true
RollUpRule	Net Assets==Assets-Liabilities	2022-12-31	NetAssets	true
RollUpRule	Net Income==Revenues-Expenses+Gains-Losses	2023-01-01 2023-12-31	IncomeStatement	true
RollUpRule	Net Income==Income from Normal Activities of Entity+Income from Peripheral or Incidental Transactions of Entity	2023-01-01 2023-12-31	IncomeStatementAlternative	true
RollUpRule	Comprehensive Income==Net Income+Other Comprehensive Income	2023-01-01 2023-12-31	ComprehensiveIncomeStatement	true

The accounting equation rule, which is a fundamental high-level accounting concepts continuity cross check rule, detects the inconsistency and informs the software user. That one inconsistency causes several inconsistencies in the report actually. When the mistake is fixed, the software helps the software user understand that everything is now correct in the report and report model;

Concept [Aspect]	Period [Aspect]	
	2023-12-31	2022-12-31
Assets [Roll Up]		
Current Assets	0	0
Noncurrent Assets	0	0
Assets	✓ 0	✓ 0
Liabilities and Equity [Roll Up]		
Liabilities [Roll Up]		
Current Liabilities	0	0
Noncurrent Liabilities	0	0
Liabilities	✓ 0	✓ 0
Equity [Roll Up]		
Equity Attributable To Controlling Interests	0	0
Equity Attributable to Noncontrolling Interests	0	0
Equity	✓ 0	✓ 0
Liabilities and Equity	✓ 0	✓ 0

- 01-Balance Sheet
- Balance Sheet [Hypercube]
- 02-Net Assets
- 03-Income Statement
- 04-Income Statement Alternative
- 05-Comprehensive Income Statement
- 06-Cash Flow Statement
- 07-Changes in Equity

So again, don't be confused by the simple nature of the report shown to make this point. A small, simple report was used only to keep the explanation easy. These same ideas work within real world financial reports.

To really appreciate the full potential here, the notions of articulation⁵⁰ and "intermediate components" (i.e. reporting economic entities using different combinations of subtotals and totals in reporting) need to be clearly and well understood.

Leveraging Types and Subtypes

When using financial reporting schemes such as US GAAP which has about 20,000 different report elements and IFRS which has about 10,000 report elements (but will eventually grow to be more similar to US GAAP); strategies need to be employed to help software users get to the right report element for their purpose.

A big part of this is using fundamental high-level concepts and type-subtype (a.k.a. general-special, wider-narrower, class-subclass) associations to find report elements. Other approaches such as disclosure templates used in conjunction with type-subtype associations.

In addition, there are other accounting associations that have been standardized⁵¹ and very likely even more such semantics will be provided for.

Finally, another very important thing to understand is the importance of references to authoritative accounting literature such as the Accounting Standards Codification⁵² (ASC) for

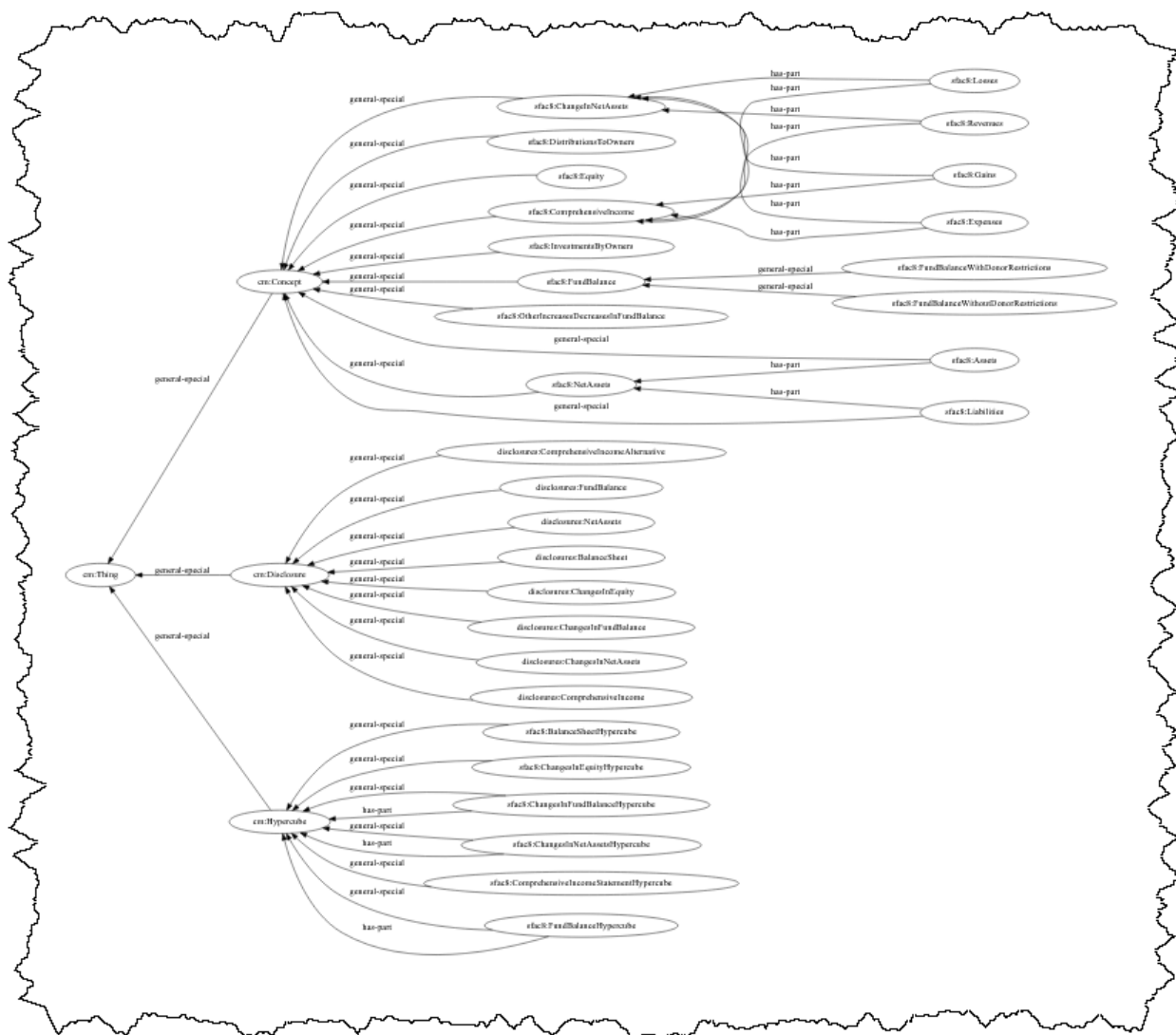
⁵⁰ Understanding Articulation, <https://digitalfinancialreporting.blogspot.com/2023/08/understanding-articulation.html>

⁵¹ XBRL International, Accounting semantics arcroles 1.0, <https://www.xbrl.org/REQ/accounting-semantics-req/REQ-2023-01-04/accounting-semantics-req-2023-01-04.html>

⁵² FASB, Accounting Standards Codification (ASC), <https://asc.fasb.org/>

US GAAP and IFRS Standards Navigator⁵³ and other financial reporting schemes. These resources can be and often are supplemented by useful interpretations and other non-authoritative but very useful tools for understanding and user this important information. For example; instructions, commentary, guidance, and so forth.

Combining these hierarchies of semantic information, facets, search, filter, and the proper use of authoritative references and non-authoritative but often more helpful and practical tools for working with the thousands, even tens of thousands of logical artifacts accountants must use effectively. Here is a very basic example of such information: (human readable⁵⁴; machine readable⁵⁵)



⁵³ IFRS Foundation, IFRS Navigator, <https://www.ifrs.org/issued-standards/list-of-standards/>

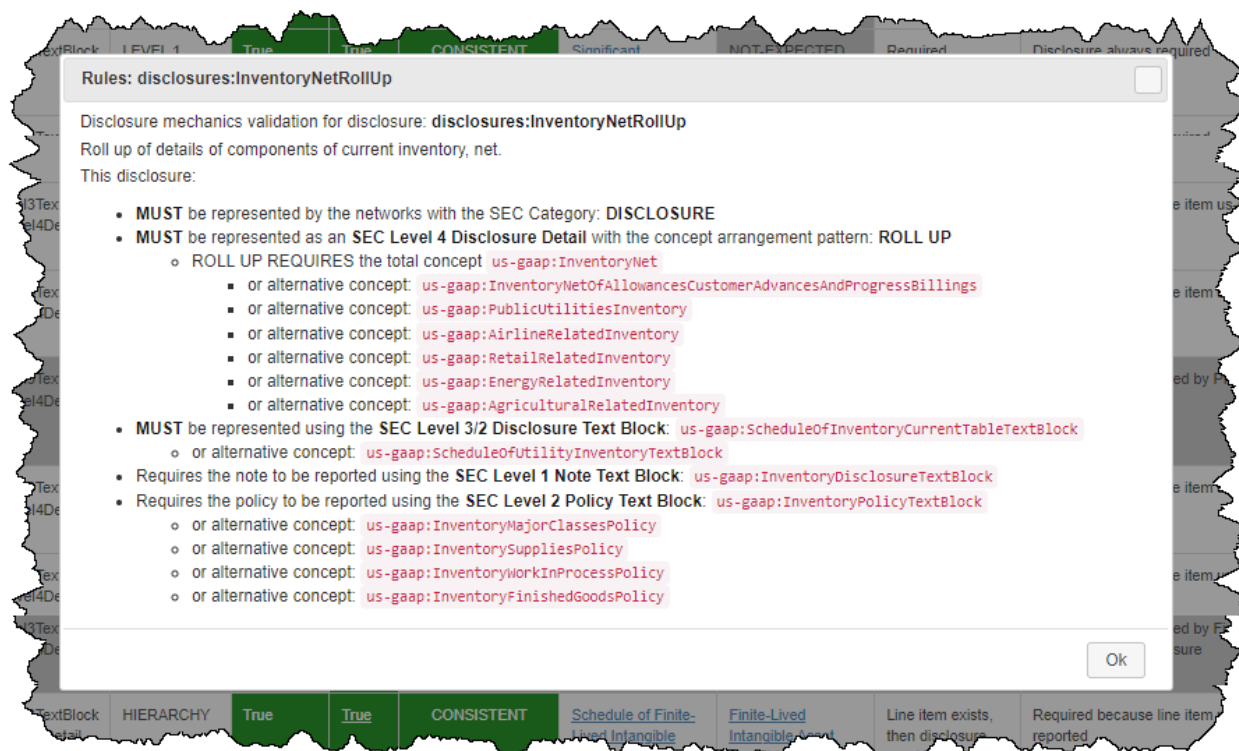
⁵⁴ Types and subtypes, human readable, <https://auditchain.infura-ipfs.io/ipfs/QmTH2iYTSWStdh5kXbadA1T9rPjAE7Z5qLF8L4wxgdhig/typeSubTypeGraph.html>

⁵⁵ Types and subtypes, machine-readable, <http://xbrlsite.com/seattlemethod/golden/sfac8/typeSubtype-rules-def.xml>

The opportunities here to provide innovative and clever tools to accountants is literally endless; but we will leave it at this for now.

Leveraging Disclosure Mechanics Rules

Similar in nature to type-subtype associations but specifically for information related to how to properly construct a disclosure is the disclosure mechanics rules. While the relation types are different, this information is effectively a hierarchy of specific information that describes each specific disclosure. Here is an example which we provided earlier:



That hierarchy of information can be used in clever ways to get software users to the report element information they need to construct a report and make sure the “organisms” or “assemblies” of report elements which they create (the disclosures) are consistent with expectation including report logic and financial accounting and reporting logic.




Leveraging Reporting Checklist

The reporting checklist leverages the disclosure mechanics rules to identify specific information Blocks as the Disclosures that make up a financial report. The reporting checklist is a set of rules that specifies what disclosures are required to be provided within a report. One use of the reporting checklist rules is to create what I refer to as an “agenda” which is a list of the remaining report fragments which need to be created in order to satisfy the rules specified in the reporting checklist.

The agenda is basically the opposite of the reporting checklist. The reporting checklist specifies what is required, the agenda specifies what is remaining to be completed in order to have what can be considered a complete report.

Again, using a very simple example, consider the prototype financial reporting scheme SFAC 8⁵⁶. From that financial reporting scheme, the reporting style “BS1-IS1” specifies that three Disclosures (Blocks of information, organisms) are required to be reported per the reporting checklist⁵⁷ of that financial reporting scheme: BalanceSheet, ComprehensiveIncome, and ChangesInEquity.

When the report is started, the Agenda looks like this:

Disclosure	Explanation	Add
	A report REQUIRES a BalanceSheet.	
	A report REQUIRES a Statement of ComprehensiveIncome.	
	A report REQUIRES a Statement of ChangesInEquity.	

The yellow indicates that the disclosure is yet to be created. The Agenda even provides the functionality to import the required disclosure from the financial reporting scheme templates. Create the disclosure and the agenda item turns green. Complete all of the disclosures and all the Agenda items will then be green as per the following:

Disclosure	Explanation	Add
disclosures:BalanceSheet	A report REQUIRES a BalanceSheet.	
disclosures:ComprehensiveIncome	A report REQUIRES a Statement of ComprehensiveIncome.	
disclosures:ChangesInEquity	A report REQUIRES a Statement of ChangesInEquity.	

This notion of an Agenda comes from a tool developed by NASA called CLIPS⁵⁸. CLIPS is an expert system that works similar to PROLOG. This video DEMO: Agenda⁵⁹ helps you understand what the Agenda is.

State Machine

What is behind a lot of the leveraging of semantic glue is what is known as a state machine. A state machine, or a finite-state machine⁶⁰ really, effectively is aware of the state of a report and provides information that is available to the software that is running behind the scenes to help the software user to create the report.

The following is the information the state machine of the software has available to anticipate the next move of the user of the software application:

⁵⁶ SFAC 8 Financial Reporting Scheme,

http://www.xbrlsite.com/seattlemethod/platinum/sfac8/sfac8_ModelStructure.html

⁵⁷ BS1-IS1 reporting checklist, <http://www.xbrlsite.com/seattlemethod/platinum/sfac8/fac/BS1-IS1/dr-rules-def.xml>

⁵⁸ *Using CLIPS to Understand Expert Systems and Logic Programming*,

<http://xbrl.squarespace.com/journal/2016/9/15/using-clips-to-understand-expert-systems-and-logic-programmi.html>

⁵⁹ DEMO: Agenda: <https://youtu.be/x7whbd4StzI>

⁶⁰ Wikipedia, *Finite-state Machine*, https://en.wikipedia.org/wiki/Finite-state_machine

State Properties		Report Elements		
Reporting scheme	2	All	Local (Added)	Percent
Fact count	122	Structure	24	100.00%
Structure count	24	Hypercube	24	0.00%
① Hypercube count	24	Dimensions	0	0.00%
Hypercube (explicit)	24	Members	0	0.00%
Hypercube (implied)	0	Lineltms	24	0.00%
Block count	32	Abstracts	37	0.00%
Disclosure count	35	Concepts	87	0.00%
① Pattern count	32			
Set				
Roll Up	12			
Roll Forward	11			
Roll Forward Info				
Adjustments				
Variances				
Text Blocks	9			
Member Aggregation				
Arithmetic				
① Validation inconsistencies depending on SBRM	0			
XBRL Calculation	0			
XBRL Formula	0			
Model structure	0			

Blocks
Disclosures
Reporting Style - (MINI-BSC-IS01-CF1) Statement of Financial Position CLASSIFIED, Income Statement Multistep, Cash flow statement NORMAL

Human Readable Renderings Generation

There are two users that need to interact with the report logic. The first is the software application. The second is a human that is using the software application.

To do this, the information that the software is processing needs to be made available to both the software and the human making use of the software. To do that, the machine-readable logic is transformed into something that a human can work with.

That transformation is done using code that is used by the software application to convert the information that exists within the report model and report that conforms to the report metamodel.

One of the very hardest transformations that takes place is to turn the information model Blocks into human readable form. Each different logical pattern of information has a different preferred rendering format that helps humans make use of the information generally by reading the information.

Over years and years standard renderings of these different logic patterns have been created and perfected by software developers⁶¹. While the actual rendering of the information is the hardest to generate, there are other human readable artifacts that are provided to the software user. There are two forms of interfaces: static and dynamic (i.e. pivotable).

Here is a static interface:

The screenshot displays the Auditchain software interface. At the top, there is a navigation bar with the Auditchain logo, a user profile for Charles Hoffman, and a 'VALIDATION DASHBOARD' link. Below the navigation bar, the main content area shows a financial statement for the reporting entity GH259400TOMPUOLS65II. The statement is organized into a table with columns for 'Concept [Aspect]', '2022-12-31', and '2021-12-31'. The table is divided into sections for Assets, Liabilities, and Equity. A right-hand sidebar shows a tree view of the financial statement structure, with 'Balance Sheet [Hypercube]' selected. The table data is as follows:

Concept [Aspect]	Period [Aspect]	
	2022-12-31	2021-12-31
Assets [Roll Up]		
Current Assets [Roll Up]		
Cash and Cash Equivalents	-648551.94	398937.76
Receivables	2035468.27	1231338.47
Inventories	451842.19	467010.2
Current Assets	1838758.52	2097286.43
Noncurrent Assets [Roll Up]		
Property, Plant and Equipment	1245567.16	1266995.32
Noncurrent Assets	1245567.16	1266995.32
Assets	3084325.68	3364281.75
Liabilities and Equity [Roll Up]		
Liabilities [Roll Up]		
Current Liabilities [Roll Up]		
Accounts Payable	2689452.31	1595349.42
Accrued Expenses	0	0

Note that there are different views of the information that can be used. Explaining all this is beyond the scope of this document by documentation is available that helps software developers understand the sorts of things that are possible such as *Getting Started with Auditchain Luca*⁶².

⁶¹ Comparison of Renderings for Concept Arrangement Patterns, <http://xbrlsite.com/seattlemethod/platinum-testcases/ComparisonOfConceptArrangementPatternRenderings.pdf>

⁶² Getting Started with Auditchain Luca, <https://digitalfinancialreporting.blogspot.com/2024/01/getting-started-with-auditchain-luca.html>

Here is a dynamic (pivotable) interface:

The screenshot shows the Pesseract application window. The title bar reads "Instance (instance_dr.xml) - Pesseract". The menu bar includes File, Home, Options and Preferences, Tools, View, Knowledge Base, Debugging, Windows, and Help. The toolbar contains icons for Get Started, New, Open, Save, and various report-related functions like XBRL Syntax, Disclosure Mechanics, Reporting Checklist, To Do List, Report Properties, Referenced Taxonomies, and Viewer. The main window is divided into several panes:

- Components (24):** A list of report components with a search filter and view options (Network View, Component View, Block View).
- Component Properties:** A pane showing details for the selected component, including Network, Table, Disclosure, Confidence, and Status.
- Main Data Table:** A table displaying financial data for a Balance Sheet. The table is structured as follows:

Reporting Entity [Axis]	GH259400TOMPUOL565II http://standards.iso.org/iso/17442	
Unit [Axis]	USD	
Balance Sheet [Line Items]	Period [Axis]	
	2022-12-31	2021-12-31
Assets [Roll Up]		
Current Assets [Roll Up]		
Cash and Cash Equivalents	(648,552)	398,938
Receivables	2,035,468	1,231,338
Inventories	451,842	467,010
Current Assets	1,838,759	2,097,286
Noncurrent Assets [Roll Up]		
Property, Plant and Equipment	1,245,567	1,266,995
Noncurrent Assets	1,245,567	1,266,995
Assets	3,084,326	3,364,282
Liabilities and Equity [Roll Up]		
Liabilities [Roll Up]		
Current Liabilities [Roll Up]		
Accounts Payable	2,689,452	1,595,349
Accrued Expenses	0	0
Current Liabilities	2,689,452	1,595,349
Noncurrent Liabilities [Roll Up]		
Long-term Debt	338,349	361,286
Noncurrent Liabilities	338,349	361,286
Liabilities	3,027,801	1,956,635
Equity [Roll Up]		
Paid In Capital	0	0
Retained Earnings	56,524	1,407,647
Equity	56,524	1,407,647
Liabilities and Equity	3,084,326	3,364,282

The above is a desktop application called Pesseract which can be downloaded and installed locally⁶³. This working proof of concept is very good at viewing reports but cannot be used to actually create reports.

Logical Schema

The semantic glue provides what amounts to a logical schema of a financial report⁶⁴. That logical schema, which effectively describes a financial report logically, can then be used to drive what amounts to an expert system for creating financial reports⁶⁵. As far as I can tell, Auditchain Luca is the world's first working expert system for creating financial reports⁶⁶. That same logic that is used to drive the software creating financial reports and verifying that the report has been created correctly can also be used to extract information from such reports

⁶³ Pesseract, <https://pesseract.azurewebsites.net/>

⁶⁴ Logical Schema of a Financial Report, <http://xbrlsite.com/seattlemethod/LogicalSchemaOfFinancialReports.pdf>

⁶⁵ Expert System for Creating Financial Reports Explained in Simple Terms, <https://xbrlsite.azurewebsites.net/2022/Library/ExpertSystemForCreatingFinancialReports.pdf>

⁶⁶ World's First Expert System for Creating Financial Reports, <https://digitalfinancialreporting.blogspot.com/2023/01/worlds-first-standards-based-expert.html>

and effectively perform financial analysis on those reports. If you think about it, all this makes sense. The same logic **describes** the financial reporting standards that are used to **create** a report, **verify** that the report has in fact been created per that description, and therefore can be used to effectively **extract** information for financial analysis.

Further, it would be incredibly difficult to manage the thousands or tens of thousands of individual logical pieces of a report without (a) a logical schema that describes the pieces and how they relate to one another and (b) enables software to help a human interact with all those pieces and keep everything sorted out.

But, if you have both the logical schema and the software then all this can actually be made to work effectively, creating a virtuous cycle that feeds itself.

You can use logic to connect things together effectively and flexibly. This approach has advantages over trying to connect things together using the “workbook”, “sheet”, “column”, “row”, “cell” document position information or geospatial position information. Defining the logical artifacts and using software makes a completely new approach to financial reporting possible. This same approach can be used more generally for creating “semantic spreadsheets”⁶⁷. This modern approach to spreadsheets will not replace all uses of the traditional electronic spreadsheet, but it can replace some use cases because of the unique capabilities that it brings to the table.

Semantic Glue Comprehensive Example

The PROOF⁶⁸ provides a complete and comprehensive example of representing the semantics of an XBRL-based digital financial report that will help a software engineer understand the semantic glue they have to work with. All the XBRL files can be obtained from this one web page⁶⁹:

PROOF (Platinum, CM)						
Standards Terms Structures Entry Point Knowledge Graph Reporting Styles (FAC Consistency, Mappings, Derivation, Reporting Checklist, Type-subtypes) Model Structure Reference Implementation Download						
Line	Label	Report Element Category	Period Type	Balance	Report Element Name	
1	01-Balance Sheet	Network			http://www.xbrl.com/seattlemethod/proof/role/BalanceSheet	
2	Balance Sheet [Hypercube]	Hypercube			proof:BalanceSheetHypercube	
3	Balance Sheet [Line Items]	LineItems			proof:BalanceSheetLineItems	
4	Assets [Roll Up]	Abstract			proof:AssetsRollUp	
5	Current Assets	Concept (Monetary)	As Of	Debit	proof:CurrentAssets	
6	Noncurrent Assets	Concept (Monetary)	As Of	Debit	proof:NoncurrentAssets	
7	Assets	Concept (Monetary)	As Of	Debit	proof:Assets	
8	Liabilities and Equity [Roll Up]	Abstract			proof:LiabilitiesAndEquityRollUp	
9	Liabilities [Roll Up]	Abstract			proof:LiabilitiesRollUp	
10	Current Liabilities	Concept (Monetary)	As Of	Credit	proof:CurrentLiabilities	
11	Noncurrent Liabilities	Concept (Monetary)	As Of	Credit	proof:NoncurrentLiabilities	
12	Liabilities	Concept (Monetary)	As Of	Credit	proof:Liabilities	
13	Equity [Roll Up]	Abstract			proof:EquityRollUp	
14	Equity Attributable To Controlling Interests	Concept (Monetary)	As Of	Credit	proof:EquityAttributableToControllingInterests	
15	Equity Attributable to Noncontrolling Interests	Concept (Monetary)	As Of	Credit	proof:EquityAttributableToNoncontrollingInterests	
16	Equity	Concept (Monetary)	As Of	Credit	proof:Equity	
17	Liabilities and Equity	Concept (Monetary)	As Of	Credit	proof:LiabilitiesAndEquity	
18	02-Net Assets	Network			http://www.xbrl.com/seattlemethod/proof/role/NetAssets	
19	Net Assets [Hypercube]	Hypercube			proof:NetAssetsHypercube	
20	Net Assets [Line Items]	LineItems			proof:NetAssetsLineItems	
21	Net Assets [Roll Up]	Abstract			proof:NetAssetsRollUp	

⁶⁷ *Special Purpose Logical Spreadsheet for Accountants*, <http://www.xbrl.com/2023/Library/SpecialPurposeLogicalSpreadsheetsForAccountants.pdf>

⁶⁸ PROOF, <https://digitalfinancialreporting.blogspot.com/2023/12/proof.html>

⁶⁹ PROOF (Platinum, CM), Model Structure, http://www.xbrl.com/seattlemethod/platinum/proof/base-taxonomy/proof_ModelStructure.html

At the time of this writing, Auditchain Pacioli⁷⁰ and Auditchain Luca⁷¹ were freely available to create and verify XBRL-based reports. All the documentation and files you need to experiment with Auditchain Pacioli and Luca are provided on the web page for the PROOF.

Further Reading

The following is additional helpful information. The documents are arranged in no particular order.

Essentials of XBRL-based Digital Financial Reporting⁷²: This document helps the reader understand important issues related to using XBRL to create XBRL-based financial reports effectively.

Accounting Basics (Brainstorming)⁷³: This document contains a lot of information about business events, the notion of classic transactions, ACTUS, etc.

Essence of Accounting⁷⁴: Relooks at some fundamental and foundational idea about accounting and reporting from the perspective of "digital".

Rules of Thumb⁷⁵: Best practices and good practices relating to representing financial report information using XBRL.

⁷⁰ Auditchain Pacioli Power User Tool, <https://pacioli.auditchain.finance/tools/PowerUserTool.swinb>

⁷¹ Getting Started with Auditchain Luca, <https://digitalfinancialreporting.blogspot.com/2024/01/getting-started-with-auditchain-luca.html>

⁷² Charles Hoffman, CPA, *Essentials of XBRL-based Digital Financial Reporting (Platinum)*, http://www.xbrlsite.com/seattlemethod/platinum/EssentialsOfXBRL_PLATINUM.pdf

⁷³ Accounting Basics (Brainstorming), <https://xbrlsite.azurewebsites.net/2022/library/AccountingBasics.pdf>

⁷⁴ Charles Hoffman, *Essence of Accounting*, <https://xbrlsite.azurewebsites.net/2020/Library/EssenceOfAccounting.pdf>

⁷⁵ *Rules of Thumb*, http://www.xbrlsite.com/mastering/Part04_Chapter07.G4_RulesOfThumb.pdf