XBRL as an Information System

The captain calls the engine room.

- *How much?*
- *Thirty!*
- *Thirty what?*
- *How much what?*

Reliable and transparent knowledge sharing is essential to operate complex systems, especially when the participants' expertise is different and sometimes even their motivations are conflicting. Accounting aims to solve this situation for thousands of years, now can also utilize the global computer network. The affordable storage, process and communication performance grew exponentially but this does not automatically translate to better transparency. The ability to access millions of pages of plain or beautifully presented reports does not mean their content is valid, coherent and comparable.

The key is how we transfer our knowledge into computer systems, considering the benefits and limitations of this process. In the following chapter, we will overview the fundamental concepts required for this understanding. The definitions will not be scientifically precise or follow very complicated research areas, but clear enough to give a coherent path from basic elements like "data" to all terms in any XBRL report. These terms will be defined "as we use here".

Transferrable Knowledge

A good analogy is photography. The traditional analogue photography uses a film that is sensitive to the light that makes molecular changes in the material during the exposure, then the film is developed and fixed. The resolution of the image is very high you can magnify very small parts of it. On the other hand, the physical object is the only one instance of that picture, you must repeat the procedure to create a copy, you can only physically manipulate the image like trimming, storing or transferring means physical operations. Although knowledge is stored in an analogue photo but is not transferrable.

Digital images are different. They are just a long array of *numbers*, one of more of these numbers describe properties of a *dot*, a huge matrix of these dots are the *image* that can appear on your screen. The resolution is fixed when you create the image, when a digital camera converts the lights to this long array of numbers, you can't "magnify" it. On the other hand, you can store, copy, transfer this number array without any further physical operation in computers. You can use algorithms to change the values to edit (sharpen, lighten, …) the picture, other algorithms can combine hundreds or even millions of pictures.

A less apparent property is that you always need extra knowledge of how to organize and use these numbers to create the image: how many dots are in one row, how many numbers describe a dot (one for a black and white image, 3 for an RGB, you may have

another for transparency, etc.). It is even possible that those numbers have no connection with properties of light, like the artificially colored images of space probes.

An extreme example is a 3D MRI scan. The images are not created by recording the light, but reverse engineered from the measurements of a rotating sensor. It creates several images from the target area while advancing with a slow known speed, the slices then combined. 3D algorithms identify surfaces of the organs or find blood vessels to gain situational awareness or even plan the surgery process and allocate resources.

Summary:

- When you use computers for knowledge management, that it is a fundamentally different, digital platform.
- The result of the transfer process is limited, less than your original knowledge: you lose all attributes that you don't transfer; you initially decide the granularity and lose all finer details; the digitized values have a finite precision and lose the rest.
- The exact transfer rules are necessary to regain the transferred knowledge together with the content, and it will still not be the same as your knowledge was before the transfer.
- However, the transferred form allows digital storing, replication and algorithmic processing of knowledge – not available in its original form.
- You can create a "familiar form of knowledge" from a very different source, like an image of your organs or a mathematical function.
- You can combine huge amount of transferred knowledge using its digital form and reliably extract knowledge that was not present in any individual item but "among" them, like an image on a vertical plane from a hundred horizontal slice.

Information System

Information systems manage transferrable knowledge. They contain

- their own terminology,
- rules to transfer a real-life situation into a model that represents it using the local terminology with a given precision,
- algorithms that allow to validate, analyze, predict or manipulate this model.

In this way, information systems help their users to gain situational awareness and interact with complex systems. Creating the proper terminology is research (highest cost); then you need education to teach people to use them; they must use their knowledge to create and use these models. But when you operate a factory or a hospital, there is no question: proper information systems are necessary to optimize the processes and avoid disasters.

Information systems in general have multiple layers, with different knowledge segments. Following the previous analogy, an information system transfers video streams of the plant security cameras to an office – this system consists of hardware and software components. The security officers know the layout of the plant, assumes how people move from one place to another. They also know the "normal activities" at

each location, the uniforms and sometimes, faces of workers; know how that is related to the hours and days. They can also look up schedules to verify if a known person should be in or not.

The abstract nature of an information system extends its capabilities beyond of the represented real-life system. You can store the lowest layer of primary knowledge, sometimes with a lower resolution, like saving the snapshots of each camera in 5 seconds or keep the voice conversations for a month. This allows “rolling back” or “speeding up” the time within the information system to understand the events that lead to a disaster or realize opportunities for optimization, like change vehicle distribution or relocate depos to decrease waiting time or empty movements. The analysis can be centralized, the results can be reused, like aviation accident analysis and recommendations – breaking the limits of physical distances.

Information systems are not just software and not even have to be digital to achieve this level. If you know the language and the notation system of accounting, you can get an equivalent picture of the business of a medieval Venetian bank, the Osman Empire or Egypt. Of course, you will not know the color, the weight or the temper of the listed animals, the smell of the crops or anything about the people and vehicles that transported them, because the model does not store that. You can’t “magnify” the image beyond the original resolution. However, you may detect someone cheating after transferring many years of books into digital form and run algorithms that can detect trends and find outliers. On the other hand, any mistake in the language or the notation system will lead to misunderstanding and misinterpretation of the stored knowledge.

The Big Picture of XBRL

XBRL – eXtensible Business Reporting Language is an information system to create a transferrable knowledge “snapshot” of “any” business activity. The first version of XBRL was created in 2000, the 2.1 version in 2003 and stable since then, minor modifications to correct typos were published 20th February 2013. XBRL is the de facto global standard of financial reporting where the required content and structure of such reports is updated annually. It is the official standard to publish ESG (Environmental, Social and Governance) reports and used in many other areas. How can such an “old” standard be used both in a continuously changing and totally different environments?

The answer is that the XBRL standard consists of two information systems. The first contains the terms and rules of creating “any” report, which is a long list of “facts”, each is a numeric, date, text, etc. value of a “concept” for a given “moment or period”. The “concept” can be the income, share price, etc., but their definitions can be found in an external “taxonomy” to which the report only refers to. The second information system allows defining a taxonomy. Each taxonomy is a complete information system: it contains terms, presentation and validation rules, calculations.

So, in the case of US financial reports, the attributes of your transferrable business knowledge are defined in the US-GAAP taxonomy, which is updated annually according to the current requirements. However, your XBRL reports are always the same: they refer to the actual US-GAAP taxonomy version, and the list of facts refer to the concepts defined in that version. A software that handles all US reports just loads the referred

taxonomy and handles the facts according to that description. If you happen to report to the EU, you should use the current version of the ESEF taxonomy, but the software works the same way but now loads the referred ESEF taxonomy.

The infrastructure and services in the XBRL ecosystem are similar to the standard shipping container. The report layer ensures the external properties: file formats and general structure that allows basic validation, homogeneous storage and general content-related services – regardless of the actual content. The taxonomy standard acts like a standard documentation about the content: receivers can describe what they want to see in the container, from simple labels to mandatory content or even describe the requested relation among the properly labeled items in the container. This description is transparent to the XBRL handler infrastructure, thus general services (like inspectors at a container station or port) can use the taxonomy to peek into and validate the content of the container.

This means, a proper XBRL handler software can validate the report according to the required taxonomies while edited by the publisher, ensure that it only accepts valid reports, thus the receiver can assume that all facts in all incoming reports contain only properly labeled facts and the report passed all validations in the taxonomies. A taxonomy is updated via its description, the XBRL infrastructure will load and apply the new description, therefore it will adapt to the changes automatically for the new reports. At the same time, a decade old report refers to, therefore automatically validated against the decade old taxonomy.

Regardless of all its flexibility, XBRL cannot avoid the fundamental weakness of dealing with transferrable knowledge: all attributes, all possible values, etc. that you want to store in the report must be defined in a referred taxonomy. In practice, an American company creates a yearly report for the SEC. Most of the finance-related facts refer to concepts defined in the US-GAAP taxonomy, but that does not cover company identification, that comes from the DEI (Document and Entity Information) taxonomy. Similarly, the company itself, an alliance, a country, etc. may also create taxonomies. The publisher may decide to create one report for several authorities; the XBRL standard supports this attitude, and each receiver may use the segments (defined by the taxonomies) that they are interested in.

The role of a taxonomy creator organization is also hard. For example, the FASB is responsible to maintain the US-GAAP taxonomy with the aim to make the US economy sufficiently transparent. This requires defining all concepts, possible values, add all mandatory flags and validation rules to all “interesting” facts, so that companies will report, and analysts can use them. Although anything that is not defined in the US-GAAP, may still appear in the report under additional (like, company or state level) taxonomies, but that will remain invisible for analysts using the US-GAAP. The result of this scenario is that all such “primary” taxonomies tend to grow significantly over time, their creators require full control over them. Also, there are multiple similar organizations, like IASB creates the IFRS taxonomy for the same goal, the EU extends IFRS and creates the ESEF taxonomy, etc. The same applies to the ESG (environmental, social, and governance) taxonomies.

The situation may seem grim, but XBRL works with transferrable knowledge: through the taxonomies every value in every fact has an objective definition. We can provide

translation rules from one taxonomy to another, and the XBRL infrastructure will convert a report from its primary taxonomy to another one (also indicating the conversion errors or missing rules). This operation is called “concordance”, FAC (Fundamental Accounting Concepts) is a generic financial taxonomy over US-GAAP and IFRS with existing translation rules between them and FAC. The benefit of this approach is that accountants can start the general audit process using the FAC “view” of the report, but “drill down” to the underlying primary layer (US-GAAP or IFRS) when needed.

Knowledge Layers in XBRL

Any XBRL report is a complex information system. As vaguely explained above, it has different knowledge layers, each must be properly managed by the XBRL infrastructure and various participants (software developers, taxonomy managers, report creators, collectors, analysts). To understand its operation and use XBRL properly, one must be aware of all layers in sufficient details for their own tasks. The following sections will describe the layers enough for a reliable, general picture. You will be able to look at any XBRL report through various tools, identify its segments, the connections of the elements, what such a report can and what it cannot accomplish. You would need much more details to manage a taxonomy or write XBRL software, but you will have a general understanding of their roles and, if necessary, explanations.

The next sections will visit the following layers as they appear above the previous ones.

1. The terms used in the XBRL “report layer”. The definitions will be precise, focusing on understanding what they mean and where you find them in an XBRL report.
2. The XBRL “taxonomy layer”. What a taxonomy can contain, how it affects the structure and content of the report, to help you navigate in a “primary taxonomy” like US-GAAP.
3. The general content and aim of a “secondary taxonomy”, explained by examples from the FAC.

Layer 0: Fundamental Terms of Transferrable Knowledge

This is an area of ontology, semantic networks, knowledge graphs, etc., each have their own, sophisticated terminologies. Instead of going into any of them, we will only define a few “keywords” as they will be used in the subsequent layers.

Data

Data is the “atom” of transferrable knowledge, an unambiguous piece of information. It can be of various types, now listing the most common ones and the issues when using them.

- **Numeric** data is obviously, a number – but not only that.
  When you transfer any numeric value, it is stored with a given precision using a given number of digits. You may think that a fraction like 1/7 or a constant like PI is exact, but that only hides the limitations of the actual computer that you use. In XBRL, you always explicitly provide the precision of any numeric value, validation rules will consider it.

In most cases, the value represents a numeric attribute, and you must provide the measurement unit. Without the unit, the data is ambiguous. The unit also appears in calculations: in XBRL you often work with the “shares” unit and currencies like “USD” or “EUR”, or a combined unit like USD/share.

- **Identifier** data is a string that identifies a knowledge item.
  We use identifiers to refer to a knowledge item, therefore, this string must be unambiguously assigned to exactly one item (not more), and whenever an identifier string appears, the referred item must be available. In a modular system, identified knowledge items come from different sources and although they assign unique strings, they may collide when using them (multiple sources defined the “green” identifier).
  We use “namespaces” to avoid identifier collision: an identifier can be local: “localIdentifier”, or external: “namespace:identifier”. In an XBRL report, you will both see local identifiers like “UsdPerShare” or “Fact_1234” and external ones like “us-gaap:Assets”, “ifrs:Assets” (note the avoided name collision) or “ISO4217:USD” (dollars).
- **Category** – when the value can be one of a list, like green / blue / … or circle / square / …
  To avoid ambiguity, the possible values must be properly defined knowledge items and the value is a reference to an identifier. This is how you refer to US dollars by the “ISO4217:USD” identifier.
- **Date** – or anything you could display in a calendar.
  Storing dates can also be complicated. Date can be represented by a single number (in Unix, the time elapsed since January 1, 1970.) numeric or alphanumeric strings according to a date format template. As mentioned before; to retrieve transferred knowledge, you need the rules you used to store it.
  In XBRL, you will use the ISO 8601: “2015-10-30” for dates or “2015-10-30T08:00:00.000Z” for time values.
- **Text** – plain or formatted character streams, from names and addresses to long descriptions.
  With free texts we can store and share knowledge, but this is not “transferrable” as it requires a human to understand it. The “unambiguous” nature of text data only means that the information system stores it “as is”. The system can use algorithms from simple search and comparison operations to training Large Language Models (LLMs) – but this is still not “transferred knowledge”.
  Texts have an additional parameter, the language, in XBRL it can appear as xml:lang ="en-US”. This allows separation of texts by the language or automatic translation to a target language.
- **Stream** – any binary stream of various types like images, audio, video, document files, etc.
  It works like texts: the system must know its type and type-dependent attributes, can apply algorithms on it, but does reliably “look into” them for knowledge. For example, you can upload an image of a person, but it is ambiguous, not “the” image, you can upload hundred more images of the same person. However, the height of the same person is unambiguous, e.g. 175cm or 68.9 inches, the same measurement in different units.

Attribute

The attribute is an identified placeholder of a Data with specified type, it provides a context to a Data. In other words, in transferred knowledge, Data appears as the value of an Attribute, and as such, its type must match the type required by the Attribute. You can't provide a Text data to a numeric attribute, or in the case of categories, the identifier of the English language as the value of a Currency.

In XBRL, us-gaap:Assets and us-gaap:Liabilities are both attributes of "Monetary" type, thus a company can report the Data 1000 with unit ISO4217:USD for us-gaap:Assets and 500 with unit ISO4217:EUR as us-gaap:Liabilities. The dei:DocumentPeriodEndDate attribute is of "date" type, and can have the value of 2031-12-31. This example also demonstrates that Attributes have external identifiers and in almost all reports, you will find attributes from multiple sources (taxonomies).

Most attributes are singular, can have one Data as value, like the examples above. Other cases the value of the attribute is a collection of Data. In XBRL, a report contains a collection unit and context definitions, and a relatively large collection of facts.

Knowledge Item

In a general information system, we use knowledge items to represent an entity in the real world: a person, a classroom, a book, etc., in this case, we often use the term "digital twin". However, the knowledge item is not necessary to be a twin of something real, like the buildings and airplanes in a simulator; or even less tangible (you will see the XBRL Context term as an example of that).

So, we can't avoid the really abstract, bottom-up approach. When you transfer your knowledge, you collect data and assign it to attributes and create "virtual property sheets". These tables are the knowledge items, each "row" contains and attribute and its value for this item. Sometimes you give an identifier to this property sheet, and you can refer to it – in XBRL, you define the measurement units by filling up their property sheets, assign unique identifier to each and refer to them later in the report. Sometimes you don't and can search for the knowledge items by their content (values of specific attributes).

Aspect

This is a radical difference from the traditional top-down knowledge representation approach: you define a hierarchy of "types" or "classes", the attributes are defined as members of them, and you must enroll your knowledge items into them. This means, you must know this classification before starting the transfer process and it is hard to change later.

The bottom-up approach is that we define Aspects as a collection of mandatory and optional Attributes. If a Knowledge Item has all the mandatory Attributes of an Aspect, then it "belongs to it" and can be handled accordingly. For example, the Location Aspect requires Latitude and Longitude Attributes; any Knowledge Item with a Location Aspect can be displayed on a map. An entity in an XBRL report does not have such Data but may have a state or even address. An algorithm can generate Latitude and Longitude, the entity now "gained" the Location Aspect and appears on the map.

In a similar way, all XBRL reports are submitted using a “primary taxonomy”, like US-GAAP or IFRS, they know nothing about the FAC taxonomy. However, we have concordance rules that translate the primary values and add FAC Attributes to the Knowledge Item “property sheets”. In this way, these items “gain” Aspects defined in the FAC taxonomy, the related rules can validate, the user interfaces display them.

Knowledge Unit

A Knowledge Unit is a network of closely related Knowledge Items. In IXBRL, Knowledge Unit is an XBRL report, or all versions of the US-GAAP taxonomy. The latter is an important note: an XBRL report refers to the then-current version of the US-GAAP taxonomy, the infrastructure must find and connect the right version.

Summary

So, what do you do when you create and information system and transfer your knowledge into it? (You may select an area of your own and try these steps as a practice.)

1. Create an attribute set that you want to collect. Some of these attributes are your own, some come from an external source – some of those may be mandatory. You ensure that the attribute naming scheme ensures safe, unique identification.
2. Check the data types that you will collect. If you have numeric values, you look for the measurement units; with texts you identify the languages you want to use; for categories you define or import the identifiable items you want to refer to.
3. Identify the Knowledge Units that you want to create, select a proper format for them. (Hint: in simple cases, you may find a spreadsheet quite sufficient.)
4. Formalize your knowledge by creating “virtual property sheets” of Knowledge Items, by storing the value of their local attributes (numbers, dates, texts, ...).
5. Assign identifiers to those that you want to refer to and use them in the attribute Data cells to register such references.
6. Store the registered Knowledge Items into identified Knowledge Units.

Congratulations, you have created a “snapshot” of your knowledge in a transferrable form!

If you used a digital platform to do this, you can easily create new snapshots by repeating step 4-6. With proper segmentation of the knowledge layers, you may even change the attribute definitions among snapshots without needing to modify the already created ones.

The XBRL standard provides you with all preliminary steps: you get the attributes (and many other services) from the taxonomies, you know what Knowledge Units you will create and what formats you can use. You also benefit from decades of professional application and countless existing XBRL reports. You only need to understand the fundamental concepts of the standard, explained in the next sections.

Layer 1: General XBRL Terms

Although it may be tempting, please do not open an XBRL report at this time. The standard allows multiple syntaxes (XML, JSON, iXBRL and CSV), all can store any XBRL report but support different use cases with slightly different internal structure. The next

sections will explain the abstract terms, then build up the structure of an abstract XBRL report and finally point out the most important differences of the syntaxes.

The goal here is to prepare you to understand, look for information in an XBRL report and to confidently use an XBRL tool – you will need an XBRL reference to manually edit a report.

Contrary to the previous section, now we start with the top level and move down to the smallest details.

Entity

The Entity is the organization that uses XBRL to report about its operation.

As learned above, an entity is a Knowledge Item with a unique identifier that we can use to refer to it. In the US, the U.S. Securities and Equities Commission (SEC) assigns a unique Central Index Key (CIK) to the registered entities, you can use this identifier when sending a report to the SEC. It is important to know that CIK is not the only possible identifier, in the EU the Legal Entity Identifier (LEI) is generally used, S&P provides a global identifier, etc.

The same company can be registered at multiple organizations and use their identifiers. The conclusion here is that this is an external identifier, “cik:0000021344” or “lei: 57VG5X0E00X0QJU7CQ58” are apparently different identifiers but both refer to Coca Cola Co. However, each report collector organization states the preferred entity identification scheme.

It is easy to see that in the XBRL information system, the same Entity can publish multiple Reports (various report types over multiple years). By the XBRL syntax, the opposite multiplication is also possible: one Report could contain knowledge about multiple Entities. However, this is generally reported as a validation error, you should refer to only one Entity in a Report.

Report

In abstract terms, an XBRL Report is a Knowledge Unit, a collection of coherent and valid Knowledge Items that together forms a “snapshot” of the operation of one Entity, according to the requirements of an authority, like “a properly filled 10-K report of Coca Cola Co. for 2023”.

Technically, the report itself can be one XML, JSON or XHTML file, or one JSON index and a set of CSV files (in the CSV syntax). However, as the XBRL allows you to create and use your own taxonomy – if you used this option, your taxonomy is part of the report.

The abstract content of a report:

- Reference to external Knowledge Units (definitions of the XBRL standard and the primary taxonomies).
- Identifiable Unit definitions (used by the numeric Data).
- Identifiable Context definitions (from analysis perspective, these are the primary Knowledge Items of the report)
- Facts (you can think of them as the rows of the property sheet of a referred Context: a valid Attribute – Data pair in that Context).

Unit

Measurement units are typical examples of “well-known things” – and just like Entity identification, that can be a problem, to ensure that all reports refer the same way to the same thing.

In XBRL, there are some internal pre-defined units, like xbrli:shares to indicate the number of shares or xbrli:pure that indicates no measurement unit. In general, XBRL recommends common standard sources for externally managed units, like ISO4217 for currencies.

You can find simple units like currency for Attributes like us-gapp:Assets, or combined ones like currency / shares to store the price.

So in the XBRL report you collect all possible measurement units, assign a local identifier to them that you will use with all numeric Data; you refer to the actual units via the recommended external identifier like ISO4217:USD. The benefit of this approach is obvious already in one report: you can simply and reliably collect all Data given in dollars, check if and where you used other currencies, etc. When dealing with multiple reports, you can rely on the standard external identifiers, you will find all Data reported in USD in all reports.

Context – primary

The Context is the key term in the XBRL report and a bit more abstract than the others. As you recall the definition of the Knowledge Item, the natural scenario is that it is a “digital twin” of an external entity (building, vehicle, person), or the primary form of a virtual entity (like vehicles in a simulator) – but sometimes it is not. In these cases, the Knowledge Item is just a virtual property sheet that we create following our knowledge transfer needs. Context is such a Knowledge Item, a locally identified collection of attributes and their values, in its primary form, related to an entity and a given time, either instant or period. “Context A” is a property sheet containing data of Coca Cola Co. at 2023-01-01; “Context B” collects data representing the continuous operation of Coca Cola Co., from 2023-01-01 to 2023-12-31; “Context C” is the snapshot of Coca Cola Co. at 2024-01-01.

As you already know, one Report contains 1-n, locally identified Context definitions. The Context contains one Entity reference, so technically one Report could contain Context definitions referring to different Entities, but this is prohibited. All Context references must point to the same Entity, using the same external identifier. In practice, this is not a limitation, the report is created by the Entity who is responsible for its content, there is no reason to report anything from another Entity. On the other hand, the syntax of the XBRL standard allows collecting Contexts from different Entities into a single Report – an XBRL summary of year 2023 from all beverages companies is technically possible.

The flexibility of the Context time allows store knowledge in finer granularity: you can create monthly snapshots or week-based periods, anything that you want to put into the report. You can create long, detailed “property sheets” – Knowledge Items with lots of Attribute / value pairs for big milestones (quarters, years), and smaller or more technical summaries of your weekly operation.

The benefit of this approach is again apparent: you find all time information in the Context definitions. It is easy to check if a report contains anything about an arbitrary moment or period and collect those Data – even if you must deal with thousands of reports of different types.

Context – dimensions

With primary contexts you can change the "resolution" of transferred knowledge in time. But what if you want to increase the granularity by a different factor? You want to segment your knowledge related to transport activities by states or regions? Your marketing performance metrics by age groups? Your travel indicators to road, rail, air, ...? You may create Attribute "clones" for those segments, but that practically multiply these sets while all these clones represent the same information, just for a different segment. XBRL provides an elegant solution to this problem: add these factors to the Context definition as optional, custom Dimensions.

Imagine that you want to store the values travelMiles Attribute of your company for 10 years in an XBRL report. You create the related 10 primary contexts; all refer to your company but the subsequent years. You can draw a bar chart displaying these values. Now, you want to separate Rail, Road, Maritime, Air. You can create Attributes like travelMilesRail, ..., but later you may need travelCost, etc. separated the same way. Later you may want to split by "Private" and "Business" – yet another multiplier... not convenient. What you really want is just to split those bars. You want to keep the bars of the total chart, create a split chart by coloring parts of the big bar by the means, or create a chart displaying Rail only. All while collecting the same, single Attribute, travelMiles, for different... sub-contexts.

This is what you get with adding Dimensions to a Context. You already have the primary Context that you used to collect data for your company by year, perhaps many others apart from travelMiles. Now, you create new contexts, the company and the year is the same, but you add the "TravelMode" custom dimension and the different "Rail", "Road", ... values for each. You can use these contexts to collect travelMiles independently, and use them as you like, e.g. drawing those charts. The "virtual property sheets" of these "finer" Contexts will likely be much smaller, but you can still collect any Attribute that you want to store in this more granular form. Similarly, if you want to further split these Contexts to Private and Business segments, you can add that too, so you create a Context where TravelMode = Rail and TravelExpenes = Business, etc. Of course, a Context can contain only one of the possible values for each Dimension. Adding new dimensions changes a 1-dimensional list to a 2D grid, a 3D cube, etc., the formal name of this structure is "hypercube" (that you will find in the XBRL reference).

With custom Context dimensions, XBRL gives you a fully flexible environment to store your knowledge in theoretically any granularity (within practical technical and size boundaries). This structure is that it allows you to use the same Attribute set that you use in the primary contexts. Adding new dimensions results a huge theoretical "cell count" but you only create the Contexts for those that have any value in it, like you would do in real life: create a property sheet in your catalogue only when you want to write something on it.

Fact

Obviously, the main goal of any XBRL report is to contain a lot of Facts. The previous terms finally allow us to define the structure of a Fact and explain how they should be handled. In simple terms, a Fact is a "row" of the "virtual property sheet" of a Context, a properly given Attribute – Data pair. It is also apparent that this is a bottom-up structure, a simple list of Facts builds up the complex structure of the report by referring to other Knowledge Items using their identifiers.

A Fact consists of the following elements.

- Context reference (mandatory local identifier): connects the Fact to exactly one Context; the Attribute – Data pair belongs to that Knowledge Item. As you know, all the complex structural details (entity, time, dimensions) exist in the Context definition, the Fact does not contain such information.
- Concept reference (mandatory external identifier): identifies an Attribute definition as "taxonomyName:conceptId"; the XBRL report assigns the local taxonomyName to an actual Taxonomy when imports it, and you will find the conceptID in that Taxonomy.
- Data: the value of that Attribute in the referred Context. As you recall from the definition of Data, in some cases, you must provide additional information depending on the Data type to avoid ambiguity.
  For Numeric Data, those are
  - Unit reference (local identifier of a Unit definition),
  - precision by giving the number of decimals to consider: value of 2 means 1.23xxx, -6 is 123,xxx,xxx.xxx (rounded to million); INF means "infinite precision", the exact value must be considered.

  For Text Data, you should provide the language.
  Identifiers (references and categories) must be resolvable.
  Date values most be in the ISO 8601: "2015-10-30" format.

Of course, there are less apparent factors that are also required to have a valid XBRL report; a compliant XBRL processor software is responsible for checking them.

- Conflicting facts. As an XBRL report is just a text file with a long list of Facts, it is possible that more Fact items "point to the same cell of the virtual property sheet": the Context and the Concept reference is the same. A careful reader may still point out the "ambiguity" of the actual Data, and that is correct. You are allowed to give multiple Text Data for the same "cell" for different languages (for obvious reasons), or multiple Numeric Data if their Units are different (like, monetary values in different currencies for convenience). The software may check if the Numeric Data versions are the same within the given precision; but would be pointless to expect similar service for Texts. A complete match among Facts is a likely copy-paste mistake and not relevant; but any Fact collision that leads to real Data conflict is an error and must be resolved.
- The Concept definition contains the expected Data type, and in Numeric case, the type of the measurement unit. Trying to give a Date to a Numeric value or providing "elapsed days" in USD is a conflict between the Data and the Attribute definition.

- The Concept definition also contains if it should be given for a period or for a date instant. If the Fact refers to an “instant” concept (share price at a given date) and a “period” Context (over the last year), that is a conflict between the Concept and the Context.

Knowledge in XBRL

XBRL is not new, for example, the US SEC collects almost 300 report types in XBRL format, close to 800,000 reports. It processes many of them and through the EDGAR API, you can access around 100 million data points about more than 17 thousand companies. (Rough numbers based on internal technical tests).

Technically this is a huge, live knowledge graph: an interconnected network Knowledge Items of Entities, Contexts and Facts. It is populated from the incoming XBRL reports that are also stored during processing, so you have both the read-only source documents (legal responsibility) and the continuously changing knowledge graph that is the best, transferrable knowledge representation of the US economy.

Companies regularly send XBRL reports to the SEC. You already know, this means lots of Contexts related to a company and a period or instant, filled with comparable values of standard Attributes defined in shared taxonomies. The reports are checked during processing, so they are valid by the processing rules. However, subsequent reports may contain Contexts pointing to the same company and time, in conflict with previous data – depending on the conflict resolution policy this can be a way to correct any error in the knowledge base while maintaining the integrity and read-only nature of the individual reports.

As a result, the SEC “knows as much as possible” from the US economy. They can improve their understanding by refining the taxonomies and asking for more details for some report types. Fun fact, they could even export any segment or their complete knowledge to a single XBRL report as the syntax allows everything that this would need. Why is it interesting? The SEC collects data for the US economy. The ESMA does the same for the EU; other countries and regions have their own local organizations, most likely using XBRL for its flexibility. There is no technical limitation to use XBRL as a bulk query and transport platform between these organizations, creating a global oversight on the economy. Of course, each organization my decide to keep parts of their knowledge local – in a transferrable knowledge environment, this is not just a dream but a simple technical task. You can identify the Attributes, Entities, Contexts, Dimension granularity, etc. that you want to leave out of the export.

However, the homogeneity of the technical layer is only a necessary foundation for a global knowledge share platform; these organizations use different terminologies in their custom, locally shared taxonomies. To overcome this limitation, you should understand the key features of the primary taxonomies (Layer 2), then see how to create homogeneous knowledge from reports using different terminologies (Layer 3).

This level shows the true potential of XBRL: this global platform connects to a specific environment through the Concept and Unit definitions. XBRL is already the official platform for collecting ESG knowledge globally, but practically any scenario that wants to reliably collect and share knowledge can use an XBRL Information System.

Technical Appendix: XBRL Format Differences

XBRL itself is an abstract standard and defines multiple formats to publish reports, optimized for a specific use case. This is a short summary of these use cases and the consequent structural differences, for the actual technical details please consult an XBRL reference.

- XML (XBRL): content-oriented and optimized structure. This xml file first defines the contexts and units as separate identified tags, followed by the list of facts that refer to the contexts and units, thus minimizing the file size and making report management easier. The taxonomy creators, the report publishers and those who analyze them work with the same, standard XML tools; with proper XML background knowledge the whole environment is self-explanatory. However, this is also a compromise: extracting the data from source systems and transforming them into the report xml is complicated; and regardless of this effort, the result is not suitable for human consumption like a properly formatted business summary. The latter must be created from the XML or even from yet another intermediary data collection.
- JSON: data and process-oriented structure. The facts are independent JSON objects that contain all context and unit information. As there is no reference resolution, this format allows very simple fact processing, filtering on context data, and any fact subset remains self-contained. However, this results in larger file size, does not support context data modification because context data is embedded in all facts, and it requires different (JSON) tooling. It is unlikely that reporters would use JSON as their primary format; and even though report processors would benefit from it, they can produce it automatically from other formats.
- XHTML: presentation-oriented format. In this case, the report is an XML-compatible HTML file that any HTML editor tool can create, the XBRL compatibility is provided by extra inline XBRL tags. The report publisher uses these tags to create the required XBRL report environment: include taxonomies, define contexts and units. Then, wrap all XBRL facts of the XHTML report (numeric values and non-numeric fragments) and connect to the context and units using the proper tags. In this format, all facts are user-friendly text, including numbers and dates which must be converted back to data values, this conversion is controlled by the ‘format’ fact attribute. The XHTML format is arguably the hardest to produce or process, however, it is perfect for human consumption both printed and online. From the publishers’ point of view, for a little extra effort (they execute the same data extraction and transformation, only to a slightly different format) they get one result that is suitable for both the authorities and the public. The report processors extract XBRL information from XML and XHTML. The latter is slightly more complicated, the files are significantly larger with the human-only content, but they can use the same tools and algorithms. The only critical difference is converting text values back to numeric or date data that appears only with this format and must be flawless.
- CSV: generation-optimized format. This consists of a JSON ‘catalog’ and multiple CSV files that contain data for the facts. A report publisher with a complex IT

infrastructure and many source systems can automate the data extraction and conversion into the CSV files and create a JSON catalog to create a valid XBRL report. For a bit more initial investment, this environment allows automatic updates to the report data content while the structure or the target report taxonomy is the same, and they can follow changes by modifying the CSV generation scripts and the catalog. The report processors need a tool to import the CSV format but with that, it's just another XBRL input.

Layer 2: Primary XBRL Taxonomies

Concepts

Dimension elements

Statement hierarchies

Validation rules

Labels and explanations

Pivot tables

The Role of Primary Taxonomies

Layer 3: Secondary Taxonomies – FAC

? fundamental terms

Concordance in action

Outlook to ESG