2.2. Building on Top of XML

Many technical people have questions about the relationship between XBRL and XML. They look at XBRL from the perspective of other traditional applications for XML and immediately are confused by XBRL, thinking it's overly complex.

**NOTE**

This section gets a little technical. It's intended for technical people who understand XML and want to know what XBRL brings to the table. If you don't know what XML is, you may want to skip this section.

2.2.1. XBRL isn't like other XML languages

Comparing and contrasting XBRL to other applications that use XML is a good way to understand it. Many people who are familiar with XML make two fundamental mistakes when they run across XBRL:

- **They think that XBRL is an XML language that's just like other XML languages.** That's not the case. XBRL is an XML language, but it goes way further than most XML languages in meeting the needs of its user domain.

- **They don't spend enough time digging into what XBRL is or why XBRL is what it is.** XBRL has a lot to it, and this information can be hard to find and rather voluminous. (Of course, one reason why we're writing this book is to make the information easier to find and more concise!)

XBRL is both an approach to making use of XML and a layer on top of XML that an XML Schema (which describes information structures, much like a data dictionary) alone doesn't provide. XML provides the syntax XBRL uses, so XBRL can use the entire family of W3C (World Wide Web Consortium) XML specifications, which you can see at see www.w3.org.

XBRL uses a specific approach to using XML. For example, XBRL consciously makes it easy for its users to avoid using the XML content model. XBRL also builds upon XML, providing things necessary to effectively automate the process of exchanging business information. For example, think of the times when even the smallest math error can create devastating problems when exchanging information. XBRL had to solve these types of data integrity problems in order to be useful within its environment. Every software application didn't need to solve this problem separately with its own proprietary solution, potentially causing inconsistent results.

2.2.2. XBRL versus XML

The following list compares XBRL and XML. (Thank you to UBmatrix who originally published a white paper on this topic and others who have contributed to this work, making the information available under a creative commons license that provides the basis for this useful comparison.)

- **XBRL is XML; XBRL uses the XML syntax.** XBRL also uses XML Schema, XLink, Namespaces in XML, and other global standards from the XML family of specifications.

- **XBRL expresses meaning; XML articulates only syntax.** XML Schema constrains syntax, but doesn't express semantics. XBRL's fundamental goal is to express business meaning, called semantics. To do so, XBRL had to use the XML syntax to and family of specifications to build additional features. To do what XBRL does with XML, you'd basically have to reinvent what XBRL has already created. Every software vendor reinventing what many business users of XBRL need makes little sense.

- **XBRL allows content validation against the expressed meaning.** Traditional XML languages, validated by XML Schemas alone, don't express enough meaning. If
that meaning doesn't exist (or isn't expressed), you can't validate information sets against that meaning. XBRL does express meaning and therefore does enable you to validate by using that expressed meaning. In addition, XBRL enables the exchange of that meaning to those consuming your information because it’s expressed in a standard way separate from business applications. With traditional XML approaches, validation isn’t as rich, so the necessary rules are embedded within the applications that read or write the XML. These rules are written application by application and in different, proprietary ways that are impossible to exchange across business systems because no one standard approach exists to enable such an exchange. With XBRL, you can exchange both the information itself and the business rules that support creating accurate information, allowing you to effectively communicate business information.

- **XBRL separates concept definitions from the content model.** Typically with XML, the concept definitions and the content model are mixed together. Further, XML provides you with only one implicit set of relations (because it has only one content model) and the definition of those relations is mixed with the definition of elements and attributes. XBRL, on the other hand, uses an *atomic approach* (flat XML content model) in defining concepts and moves the expression of relations away from the XML schema. This separation of concept and relation definition leads to the next benefit of XBRL.

- **XBRL can express multiple hierarchies of explicit relations.** Because XBRL separates concept and relation definitions, you can define more than one hierarchy of such relations. Further, the hierarchies of relations defined can be explicit, unlike XML’s implicit content model.

- **XBRL provides organized, prescriptive extensibility (the ability for users to make adjustments).** XML’s greatest strength is also its greatest weakness: XML is extensible everywhere, in every direction. XBRL is extensible in a specific, prescriptive, and therefore predictable manner. As such, the extensibility is usable without modifying software for the extension. You can think of this difference as XBRL always having the same shape.

- **XBRL provides a multidimensional model.** Online analytical processing systems (OLAP)-type systems can use XBRL’s multidimensional model to provide flexible information presentation and the ability to "slice and dice" information. Business intelligence systems in particular are big users of the multidimensional model. Although you can make XML fit into a multidimensional model, it can be a struggle in many cases. XBRL fits quite nicely into the existing applications that make use of the multidimensional models. Alternatively, you can use an existing architecture and application profile that’s intended to fit into an application that uses the multidimensional model. Getting information into applications that use the multidimensional model is important because more and more applications, such as business intelligence applications, are leveraging the characteristics of the multidimensional model.

- **XBRL enables intelligent, metadata-driven connections to information.** With XBRL, business users can connect information by adjusting metadata rather than by requiring technical people to write code. As such, rather than building multiple point solutions, XBRL enables the creation of effective and efficient solutions that allow extensibility and that don’t require programming modifications to connect to new information or new information models. These metadata-driven connections are possible because of the prescriptive manner of XBRL’s extensibility; the "shape" of XBRL is always the same. With XML, a programmer has to enable pretty much every new connection when writing code because XML communicates only technical syntax and does so at the data level, not the meaning level, of information and because the shape of different XML implementations can be so varied.

XML replaces a multitude of different approaches to exchanging data with one standard Web-friendly approach. Can traditional XML approaches do all the things that XBRL can do? Absolutely. XBRL is a traditional XML language with an additional layer built on top of it. In order to have the functionality of XBRL, most traditional uses of XML require building all of XBRL’s functionality for each XML language created. But XBRL provides all this "out of the box" because things like expressing meaning and validation against that meaning are so core to XBRL’s reason for being. XBRL already includes these pieces because business users have said they needed these features.