Understanding ebXML
Unraveling the Business Web of the Future

David Mertz
June 01, 2001

When you read about ebXML, it's difficult to get a handle on exactly what it is -- and on what it isn't. The 'eb' in ebXML stands for "electronic business," and you can pronounce the phrase as "electronic business XML," "e-biz XML," "e-business XML," or simply "ee-bee-ex-em-el."

What is ebXML?
On one hand, ebXML seems to promise a grand unification of everything businesses do to communicate with each other. On the other hand, one could be forgiven for thinking that ebXML amounts to little more than a pious, but vacuous, declaration that existing standards are worth following. As with every "next big thing," the truth lies somewhere in the middle.

**ebXML Terminology**

*Registry:* A central server that stores a variety of data necessary to make ebXML work. Amongst the information a Registry makes available in XML form are: Business Process & Information Meta Models, Core Library, Collaboration Protocol Profiles, and Business Library. Basically, when a business wants to start an ebXML relationship with another business, it queries a Registry in order to locate a suitable partner and to find information about requirements for dealing with that partner.

*Business Processes:* Activities that a business can engage in (and for which it would generally want one or more partners). A Business Process is formally described by the Business Process Specification Schema (a W3C XML Schema and also a DTD), but may also be modeled in UML.

*Collaboration Protocol Profile (CPP):* A profile filed with a Registry by a business wishing to engage in ebXML transactions. The CPP will specify some Business Processes of the business, as well as some Business Service Interfaces it supports.

*Business Service Interface:* The ways that a business is able to carry out the transactions necessary in its Business Processes. The Business Service Interface also includes the kinds
of Business Messages the business supports and the protocols over which these messages might travel.

**Business Messages:** The actual information communicated as part of a business transaction. A message will contain multiple layers. At the outside layer, an actual communication protocol must be used (such as HTTP or SMTP). SOAP is an ebXML recommendation as an envelope for a message "payload." Other layers may deal with encryption or authentication.

**Core Library:** A set of standard "parts" that may be used in larger ebXML elements. For example, Core Processes may be referenced by Business Processes. The Core Library is contributed by the ebXML initiative itself, while larger elements may be contributed by specific industries or businesses.

**Collaboration Protocol Agreement (CPA):** In essence, a contract between two or more businesses that can be derived automatically from the CPPs of the respective companies. If a CPP says "I can do X," a CPA says "We will do X together."

**Simple Object Access Protocol (SOAP):** A W3C protocol for exchange of information in a distributed environment endorsed by the ebXML initiative. Of interest for ebXML is SOAP’s function as an envelope that defines a framework for describing what is in a message and how to process it.

The ebXML.org homepage offers this brief characterization:

> ebXML is a set of specifications that together enable a modular electronic business framework. The vision of ebXML is to enable a global electronic marketplace where enterprises of any size and in any geographical location can meet and conduct business with each other through the exchange of XML-based messages.

Or in other words, ebXML hopes to succeed Electronic Data Interchange, more often known by its abbreviation, EDI. (Official descriptions tend to emphasize learning from EDI rather than throwing it out.)

**ebXML terminology**

Sorting out ebXML involves a few steps. Perhaps the first thing necessary for understanding the details of ebXML is to digest an alphabet soup of new acronyms and other special terms. There are a number of these terms in the sidebar to the right (ebXML terminology) to consider before looking at the whole "vision" of ebXML interactions. Additional terms fit into the entire system, but these particular terms make a good starting point. With this new vocabulary in mind, and a bit of the following background on where ebXML comes from, you can begin to make sense of how all of the differing processes in ebXML hold together.

After describing what ebXML does (at least in outline) at the beginning of this article, a final section looks in more detail at the Business Process Specification Schema, which makes up one of the most important elements of ebXML’s underlying infrastructure.

**Background**

ebXML is an initiative whose participants and endorsers consist of just about every big company and association of government standards worldwide that you can think of. Well, maybe not every one you can think of, but certainly hundreds of large companies and bodies.
Computer/technology companies are not the only entities that endorse ebXML; backers include a large number of industrial, shipping, banking, and other general-interest companies. The direct sponsors of ebXML are OASIS (Organization for the Advancement of Structured Information Standards) and UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business). Lots of standards bodies also have a finger in the pie, including NIST (National Institute of Standards and Technology) and W3C (World Wide Web Consortium).

With such a collection of supporters, it would seem that ebXML is destined to take over the world. I tend to have a cynical attitude toward industry buzzwords and hype. In the case of ebXML, however, I mostly expect it to live up to its billing as a global protocol for most business transactions within the next five years.

In my opinion, ebXML will succeed in becoming universal by incorporating into the specifications more and more of what businesses do anyway as much as it will by actually getting businesses to do business differently. I'm not sure if my estimation is cynical or if it is encouragement at the openness of ebXML specifications, but the ebXML initiative clearly holds an embrace-existing-standards-and-methods attitude.

**Putting it together**

An illustration (Figure 1) based on the ebXML Technical Architecture Specification (see Related topics) will probably go a long way toward sorting out what ebXML means for business.

Company A in Figure 1 below will first review the contents of an ebXML Registry, especially the Core Library which may be downloaded or viewed there. The Core Library (and maybe other registered Business Processes) will allow Company A to determine the requirements for their own implementation of ebXML (and whether ebXML is appropriate for their business needs).

**Figure 1: High-level overview of ebXML interaction between two companies**
Based on a review of the information available from an ebXML Registry, Company A can build or buy an ebXML implementation suitable for its anticipated ebXML transactions. The hope of the ebXML initiative is that vendors will support all of the elements of ebXML. At such time, an "ebXML system" might be little more than a prepackaged desktop application. Or maybe, more realistically, the ebXML system will at least be as manageable as a commercial database system (which still needs a DBA). Figure 1 suggests that the hypothetical Company B uses something like this prepackaged application.

Either way, the next step is for Company A to create and register a CPP with the Registry. Company A might wish to contribute new Business Processes to the Registry, or simply reference available ones. The CPP will contain the information necessary for a potential partner to determine the business roles in which Company A is interested, and the type of protocols it is willing to engage in for these roles.

Once Company A is registered, Company B can look at Company A’s CPP to determine that it is compatible with Company B’s CPP and requirements. At that point, Company B should be able to negotiate a CPA automatically with Company A, based on the conformance of the CPPs, plus agreement protocols, given as ebXML standards or recommendations.

Finally, the two companies begin actual transactions. These transactions are likely to involve Business Messages conforming to further ebXML standards and recommendations. At some point in all of this, however, "real-world" activities will probably occur (for example, the shipment of goods from one place to another, or the rendering of services). ebXML will have helped in agreeing to, monitoring, and verifying these real-world activities. Of course, in our "information economy," a lot of what goes on might stay within the realm of ebXML -- maybe everything within a particular business relationship.

The Business Process Schema

The UN/CEFACT Modeling Methodology (UMM), which utilizes UML, may be instrumental in modeling the ebXML Business Processes. However, such modeling is simply a recommendation, not a requirement. In any case, since this article targets XML developers and does not address OOD (object-oriented design), it is more interesting herein to look at the representation of the models in XML documents conformant to the Business Process Specification DTD and XML Schema. The DTD (named "ebXMLProcessSpecification- v1.00.dtd") appears, at this time, to be the primary rule representation. Both this DTD and a W3C XML Schema, which is (presumably) semantically and syntactically compatible, may be found in the EbXML_BPschema_1.0 recommendation (see Related topics).

ebXML process specifications have a root element ProcessSpecification. A particular process specification may contain subnode references to other process specifications, as well as to document specifications and other information. The DTD declaration for ProcessSpecification provides an overview of the structure of a Business Process document:
Listing 1: ProcessSpecification DTD declaration

```xml
<!ELEMENT ProcessSpecification
  (Documentation*,
    MultiPartyCollaboration)*)>
<!ATTLIST ProcessSpecification
  name    ID    #REQUIRED
  version CDATA #REQUIRED
  uuid    CDATA #REQUIRED >
```

The attribute `uuid` is a globally unique identifier for a process specification; the `name` and `version` are specific to the model represented (the `name` should not collide with nested process specifications).

Within a process specification, a `Package` defines a set of collaborations that may be either `MultiPartyCollaboration` elements or `BinaryCollaboration` elements. Collaborations, in turn, contain a variety of roles for the parties. An excerpt from the sample process specification contained in the EbXML_BPschema_1.0 recommendation (see Related topics) is helpful in sorting out this structure:

Listing 2: A package of collaborations

```
<Package name="Ordering">
   <!-- First the overall MultiParty Collaboration -->
   <MultiPartyCollaboration name="DropShip">
      <BusinessPartnerRole name="Customer">
         <Performs authorizedRole="requestor"/>
         <Performs authorizedRole="buyer"/>
         <Transition fromBusinessState="Catalog Request" toBusinessState="Create Order"/>
      </BusinessPartnerRole>
      <BusinessPartnerRole name="Retailer">
         <Performs authorizedRole="provider"/>
         <Performs authorizedRole="seller"/>
         <Performs authorizedRole="Creditor"/>
         <Performs authorizedRole="buyer"/>
         <Performs authorizedRole="Payee"/>
      </BusinessPartnerRole>
   </MultiPartyCollaboration>
   [...]  
   <BinaryCollaboration name="Request Catalog">
      <AuthorizedRole name="requestor"/>
      <AuthorizedRole name="provider"/>
      <BusinessTransactionActivity name="Catalog Request"
            businessTransaction="Catalog Request"
            fromAuthorizedRole="requestor"
            toAuthorizedRole="provider"/>
   </BinaryCollaboration>  
   [...]  
</Package>
```

Conclusion

The approval of ebXML specifications is moving along at a fairly rapid pace (certainly for a standards organization). The draft specifications were approved as version 1.0 recommendation in mid-May, 2001. I suspect that it will take another year or two to shake out all of the issues and details for such an ambitious vision. It appears, however, that ebXML is on the way to widespread
use a few years down the road. Now is the time, therefore, for businesses to begin a serious consideration of their own ebXML implementation plans.
Related topics

- Find the sponsors of ebXML at: UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) and OASIS (Organization for the Advancement of Structured Information Standards).
- The ebXML initiative has produced a number of specifications. Find all of these specifications at ebXML.org. As specifications proceed through their approval process, their exact URLs will change, so it is best simply to navigate via the ebXML homepage. If later versions of the documents mentioned here are produced, it will obviously make sense to refer to those superceding versions:
  - Glean a sense of the whole ebXML system (with a bit of work) from: ebXML Technical Architecture Specification 1.04 (ebXML_TA_v1.0.4).
- Find out more about SOAP messaging in an article by Bob DuCharme about building your own SOAP client and in the developerWorks XML messaging with SOAP tutorial.
- Uche Ogbuji covers ebXML and other matters relating to semantic transparency in his developerWorks column, Thinking XML. Check out the previous installments:

© Copyright IBM Corporation 2001
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)