Beginning To Define ebXML

Initial Draft

File Name	Version
BeginningToDefineebXML	1

Abstract

This document provides a visual representation of how the ebXML Architecture could work. As ebXML evolves, this document will also. This document is the combination of derived works from ALL the ebXML groups. Many specific architectural items still need to be clarified.

This document is work in progress of the ebXML Architecture team and will eventually be publicly available to all participating companies in ebXML.

Verify Version and Completeness Prior to Use

The responsibility for using the latest level of this document lies with the reader. Contact the owner to ensure correct version.

Document History			
Name	**Change**	**Reason for Change**	**Date**
Duane Nickull, XMLGlobal, duane@xmlglobal, 604-717-1100 David Webber, XMLGlobal	1.0	Initial Draft	12/1999 - 1/2000

1.0 Introduction

This document outlines the ebXML technical architecture.

2.0 Mandate

ebXML must exist as a vendor, platform, o/s and data source/destination neutral method for exchanging electronic business data on a global
scale.

3.0 ebXML Architecture

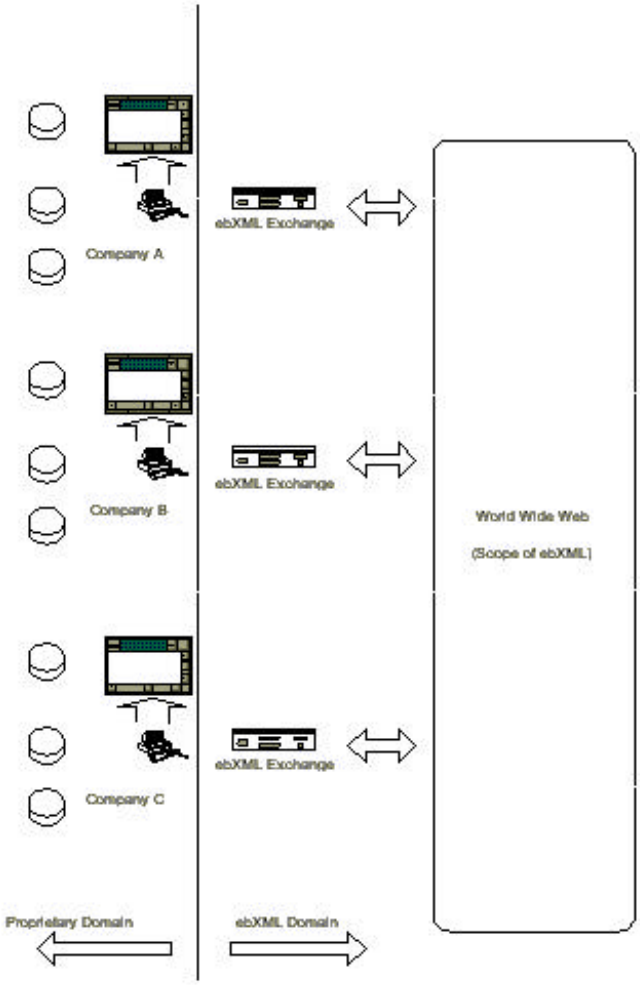
This paper begins to define the architecture and components that comprise ebXML. It presents the functional and technical requirements for each component. It also defines the functional interfaces between these components expressed in both a series of protocols and transactions.

Before defining the Architectural Componentry, a clear scope definition for ebXML is required.

3.1 ebXML Scope Definition

ebXML employs an Internet based Architecture. ebXML shall encompass all web bound messages at the point they leave a participant's private network or computer.

Image 1. examining the scope definitions for ebXML and defining entry/exit points.



This side is out of Scope

Data and messages on the private side are not within the scope of ebXML. The data does not need to conform to the ebXML specification unless the data is claiming to be ebXML compliant data. This can be done via a Root tag declaration. This rule will give companies the ability to preserve their existing system (not a requirement) and transform outgoing messages only. Alternatively, data may remain ebXML compliant within an organization's private network. Data not within the scope of ebXML can be defined as:

- data transversing a private intranet AND not claiming to be an ebXML compliant message
- data not claiming to beebXML compliant

This side is In Scope:

Data leaving the output of an ebXML Gateway Server AND claiming to be ebXML compliant data (via the root tag) MUST be conformant to the Technical Design Rules of ebXML. (to be defined). There should be strict conformance rule-sets. ebXML will be watered down if it is merely an "Envelope" for the actual data.
Syntax, semantic and security issues need to be firmly in place. Rule (Recommendation): All data claiming to be within the scope of ebXML MUST be compliant with the ebXML technical Architecture.

3.2 Architectural Components

ebXML implementation is a distributed system that consists of ebXML Gateway Servers (EGS) and one or more integration agents organized into a heirarchy. ebXML is a scalable solution to exchange XML data based on pre-set business methodology and supports Registries, Repositories and an extensible communication messaging system.
ebXML Components:

The identifiable components which make up the ebXML Technical Architecture are:

1. The language - A set of elements and attributes. There is much to be decided with respect to these language componenets.
2. Registries and Repositories - a component to house Identifiable, reuseable business information objects (IRBIOs) and allow access to them based on XML queries
3. A protocol for transportation - using http will suffice within the scope fo ebXML.
4. A protocol for security - is being discussed
5. A syntax for expressing the language - ebXML will adhere to the XML 1.0 syntax. No other syntax will be considered conformant.
6. Ebxml Gateway Server - a defined entry point to the ebXML scope.

3.2.1 Language - ebXML Elements & Attributes

There have been several ideas surrounding the creation of an ebXML Language Component. There appear to be two main viable options. The first is the creation of an entire vocabulary, expressible as a set of Elements and Attributes, that can accommodate all business methodology. The second, is a system of X-Links embedded within DTD to reference those business methods via Registries and Repositories. Each have a series of considerations.

3.2.1.1 Overall design goals

The deliverable for an ebXML language is to have a formal specification for conducting all business transactions via the ebXML architecture. It must be delivered in a timely manner in order to avoid fragmentation of the existing proposal into factional specifications. The final specification must include a way to encapsulate all current semantic information from all existing standards into the ebXML messaging. The ebXML Committees which are responsible for delivery of language initial recommendations are largely the Business Methodology Committee www.ebxml.org/project_teams/business_process.htm and the Messaging and the Transport, Routing and Packaging Committee www.ebxml.org/project_teams/transport.htm. The Core Components Group would also further refine the language Recommendation.

3.2.1.2 Considerations

Watering down a set of Elements and Attributes so it can be extensible into everything to everybody is probably a bad choice. By adopting this kind of a standard, you are not really creating a standard at all but rather, an enveloping system. Several of these enveloping architecture scurrently exist. The finalization of a formal specification for the Language will have to be adopted by all involved. The development of a "One size fits all" language set of Elements and Attributes would probably be difficult due to the overwhelming diversity from contributors. These contributors to the final specification would often have conflictory semantic standards they wish to see preserved within the final specification. An addressable system, employing Registries and repositories is probably a better solution. One potential drawback is the relative infancy of X-link and lack of extensibility of current XML 1.0 DTD's. Both of these drawbacks are being addressed. The maturing of XLink is well under way with efforts by most software vendors to support the latest recommendations from the w3c.org. A new, enhanced version of a DTD, called an eDTD, is now being discussed and solves many of the shortcomings of the existing XML v1.0 DTD.

3.2.1.3 Option 1: Entire ebXML Vocabulary

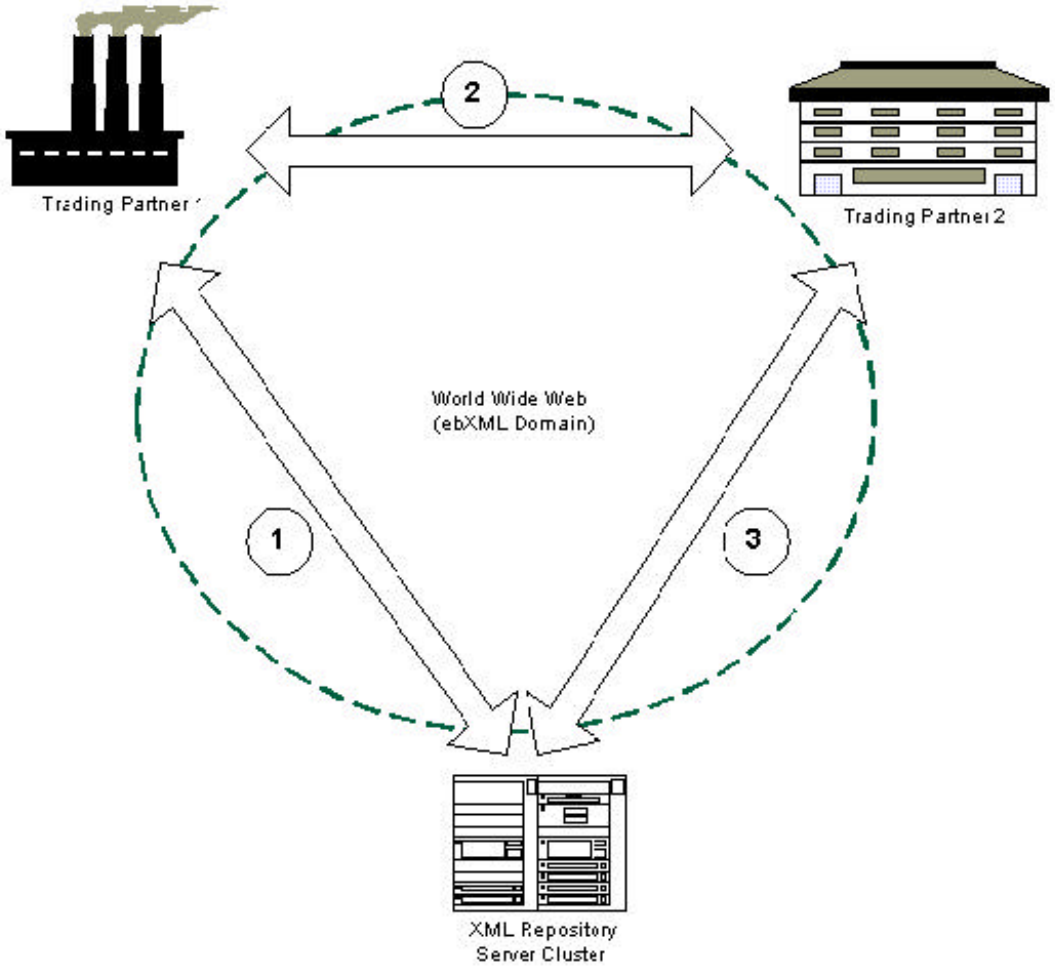
Synopsis:
The idea is to create an entire language which will be used to A set of predetermined XML elements and attributes organized into easy to understand guidelines. These Core components must adhere to the following design rules;
Proposal:
A ebXML Language must be extensible enough to support being mapped to by *ALL* other standards. A good test for this might be to map business methods like Purchase Orders from standards like x-12. The resultant ebXML should then be able to be mapped back to it's original form by reversing the mapping procedure and result in an exact duplicate of the original. This must be true for *ALL* participating standards (Editor's note - this may get real ugly!!!) There are many existing standards that must be considered when finalizing the Element sets.

Elements must be defined by the messaging and business methodology committees. There should be several draft periods to allow for all parties to analyze whether the elements and attributes will meet their needs. <IMHO> I think that this is an unrealistic solution based on the need for a "timely" solution. Also, it is not likely, given the existing fragmentation of current standards, that such an agreement would be forthcoming.</IMHO>

3.2.1.4 Option 2: Web Based Repository System

Synopsis:

An XML repository contains one or more XML glossaries, each of which is designed for use in a single business domain or industry. The XML repository provides a single point of reference that XML/EDI based applications can use to ensure consistency and maintainability of shared definitions and processes. XML repositories must allow both automated and manual Internet-based queries to the information stored in them with a standardized API. There are currently many advocates of repositories for business semantics. This makes it difficult to identify the best type of repositories for use with XML/EDI.



Demonstration of a Repository based Transaction:

1. A user or software process in Trading Partner 1 (the factory) queries the global repository for those common business objects to be passed to trading partners such as Trading Partner 2.
2. References to the identified business objects are exchanged as part of the set up process for the transaction by the trading partners. This would ideally be accomplished by using a system of X-Links embedded within DTD's. The X-Linking mechanism references the Identifiable, re-useable Business Information Object (IRBIO) stored in the XML Repository.

3. The references are used to map received data into the organization's application system. The intent of the global repository is to be a dynamic mechanism that responds through an open API (application programming interface).

Many papers exist to aide clarification of what is required from an efficient XML repository. Put simply, an XML repository for XML/EDI needs to contain several key components that set it clearly apart from other types of repositories. These components include:

1. Relevant EDI conventions that underpin business information interchanges with code and element attributes gleaned from twenty years of EDI knowledge.
2. XML glossaries that relate groups of facts relevant to a particular industry or business domain together.
3. Web forms-based interfaces allowing end users to locate specific information content in an intelligent
4. A read-only API that allows programs to access the information they need from the repository using a simple reference number indexing method and specifying the form in which the retrieved information must be returned to them.
5. A simple reference index schema that links together legacy and newer XML-based business entities.
6. An extensible architecture that allows registration of other types of information and process components in the repository. Examples of information that can usefully be made available through an XML repository include XML/EDI DTDs, X-Schema or RDF rules for processing EDI message, XSL or other stylesheets for displaying messages, intelligent agents to handle processing commands such as SQL calls, JavaScript, Java and other pre-programmed components.
7. Object facilitation layers allowing OMG MOF and UML 1 technologies to be used with XML repositories.
8. Deployment and maintenance through the Internet to ensure open access to the information.
9. Custodian and domain expert who manages and controls the information to ensure its quality and integrity.

3.2.1.5 ebXML Practicality

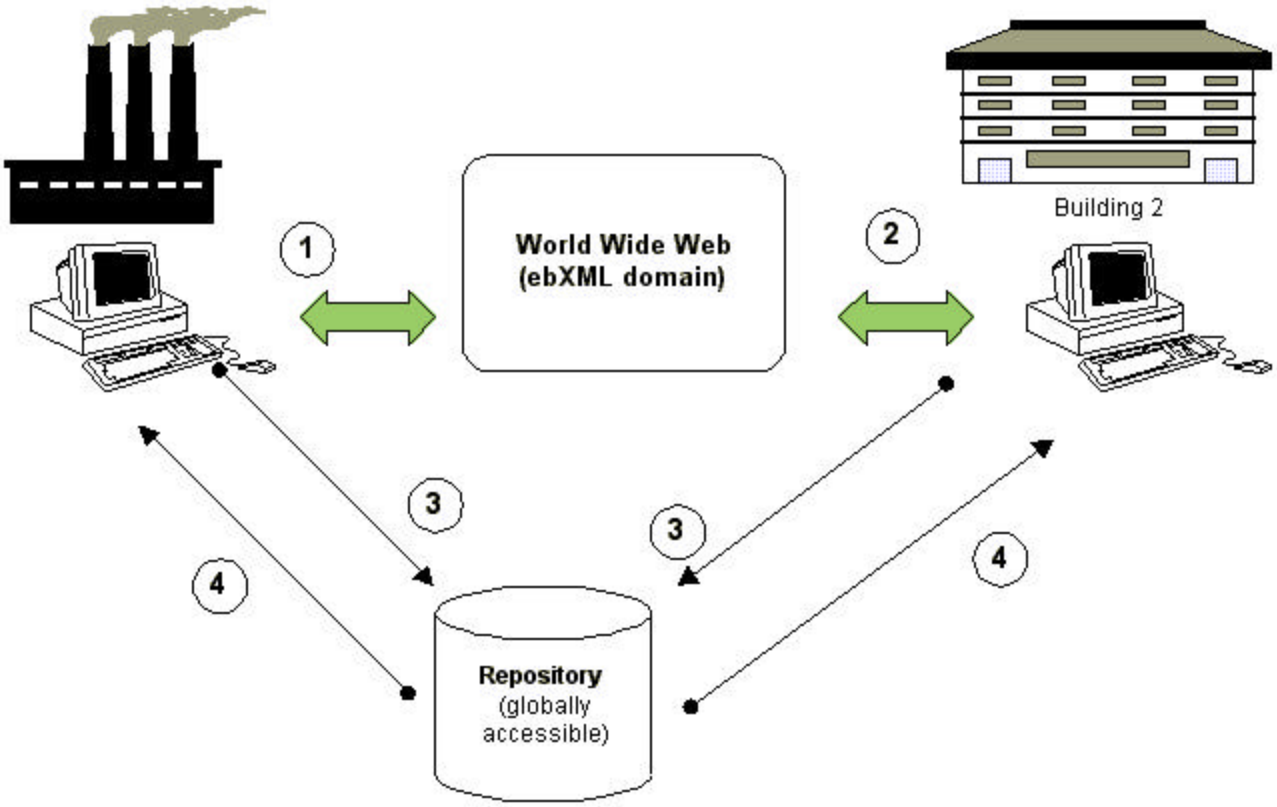
The subject of Repositories is addressed from the parent document.

3.2.2 Registries and Repositories

The design rules for registries and repositories are to be given careful consideration. The Registry and Repositories Committtee is looking at the issue very carefully and will forward Design Rule Recommendations to the Steering Committee.

What is a Repository:

For the purposes of the ebXML Architecture, a repository is an Internet accessable server that acts as an Object Database. The repositories will likely house versioned data objects. The data objects housed in a particular repository are likely to be based around a similar theme. The Repository accepts incoming requests (in XML syntax) for an Object. The Repository then routes the correct response back to the resource specified in the Request.



In the above example, we see the architecture of a typical Repository Transaction.

The Factory on the top left sends an ebXML message (Item 1) to the entity in the building on the top right (Building 2). The message is sent via the Internet in XML syntax. The Messaging, transportation and Routing Committee will provide final specification for this. Both of the participating organizations can request objects via an ebXML Gateway Server. In another possible scenario, both organizations cans also use an XML syntax containing XLinks embedded within DTD's to access the objects. (See example syntax below).

<!ATTLIST ListPrice href CDATA #fixed
'handle://&root-url; #id(V1304)' %Xlink-syntax;>

3.2.3 Transport Protocols

The transportation of ebXML data can be easily accomplished by adopting the Hyper Text Transfer Protocol.

Overall design goals:

Choose the best protocol for web based delivery of text based messages and requests.

Considerations:

Use the existing HTTP as described at http://www.w3.org/Protocols/
EbXML

3.2.4 Security Concerns

There are numerous concerns to address within the scope of ebXML for Security.

Messages:

All ebXML messages must be secure.

Registries and Repositories:

Must also be secure.

There are several existing security protocols that may fill this void. Much work needs to be done in this area.

3.2.5 ebXML Syntax

ebXML needs top conform to a standardized syntax.

Overall design goals:

- A syntax needs to be readable by both machines and humans.
- The syntax used should have a wide range of software tools that currently support it via the Internet
- The syntax should be O/S and Platform independent
- The syntax must be conformant to an existing specification (i.e. - no new proposals)
- A validating procedure needs to be present in the syntax specification
- The syntax needs to be not overly complicated.
- The syntax needs to include a section to deal with forwards compatibility issues.
- The syntax must be able to deal with multibyte character sets

Considerations:

Considering the above Design Goals, the only current viable option is to adopt the XML 1.0 syntax. It has basically been implied from the start
that XML 1.0 would be the default syntax. Issues that need to be clarified are whether or not other syntax's should also be supported. The first
option is to limit data traveling within the scope of ebXML to valid XML. The second is to allow other syntax's to enter the scope as well.

This problem does not seem to be within any one ebXML Committees domain. It does however, traverse all Committees. By default, the
Technical Architecture Committee should forward a recommendation to the steering committee.

Option 1: XML 1.0 only.

Synopsis:

This option will declare that XML 1.0 syntax to be the only valid syntax for messaging and data within the scope of ebXML.

<IMHO> I think that this should be placed into words and voted on as a recommendation to the Steering committee.</IMHO>

Option 2: Allow other syntax as well

Synopsis:

The objectives would be to allow non XML syntax within the scope of ebXML. To further the cause for this argument are speed (efficiency) at

the parsing level and smaller bandwidth for the transport layer.

Against are the overwhelming amount of extra parsing applications that may need to be included within the ebXML Architecture.