

# ebXML Message Service

An In-depth Overview

Christopher **Ferris** <chris.ferris@sun.com>

## Abstract

Today's business applications were designed primarily for use within the enterprise. However, with the growth of the Internet as a medium for inter-enterprise business communications, the technologies that have been employed to integrate business applications within the enterprise have proven largely unsuccessful when extended beyond the boundaries of the enterprise's corporate network infrastructure.

Businesses demand a quality of service that can ensure that correspondence with their partners, suppliers and customers is securely and reliably delivered in a timely manner. A business should expect no less when it comes to the exchange of business information via the Internet. Business quality messaging is a term that describes the set of requirements that a business should expect of a messaging service that is used to exchange business information with partners, suppliers and customers.

A number of emerging, de facto, or de jure standards currently exist that are designed to deliver business information over the Internet in a secure and reliable manner. However, for the most part, these are typically designed around a specific, industry vertical, transaction vocabulary. None to date has been designed to address the broader context of providing an open, non-proprietary messaging infrastructure that can be used for the secure, reliable exchange of any manner of business information, regardless of transaction vocabulary or encoding and regardless of the choice of vendor solution.

The ebXML Message Service specification [[EBXMLMS](#)] the first, truly open, standards track specification that has been designed to be suitable for use by businesses for the purpose of the secure, reliable exchange of any manner of electronic business information with their

partners, suppliers and customers.

## 1. Executive Overview

Today's business applications were designed primarily for use within the enterprise. However, with the growth of the Internet as a medium for inter-enterprise business communications, the technologies that have been employed to integrate business applications within the enterprise have proven largely unsuccessful when extended beyond the boundaries of the enterprise's corporate network infrastructure for a variety of reasons including:

- Lack of standard interfaces - business applications deployed by one enterprise are rarely compatible with those deployed by its partners, suppliers and customers either due to incompatible, vendor proprietary, interfaces or to differences in version and configuration
- Security issues - enterprise firewalls to the Internet will not allow direct access to the enterprise's computing resources
- For those protocols that are perceived as "firewall-friendly" such as HyperText Transfer Protocol ([HTTP](#)) and Simple Mail Transfer Protocol ([SMTP](#)), reliability is not an inherent feature

In short, today's business applications were neither designed for the Internet, nor were they designed for inter-enterprise messaging. In many cases, they were not designed for use with any network.

Businesses demand a quality of service that can ensure that correspondence with their partners, suppliers and customers is securely and reliably delivered in a timely manner. A business should expect no less when it comes to the exchange of business information via the Internet. Business quality messaging is a term that describes the set of requirements that a business should expect of a messaging service that is used to exchange business information with partners, suppliers and customers.

A number of emerging, de facto, or de jure standards currently exist that are designed to deliver business information over the Internet in a secure and reliable

manner. However, for the most part, these are typically designed around a specific, industry vertical, transaction vocabulary. None to date has been designed to address the broader context of providing an open, non-proprietary messaging infrastructure that can be used for the secure, reliable exchange of any manner of business information, regardless of transaction vocabulary or encoding and regardless of the choice of vendor solution.

Many of the existing standards address the requirements for business quality messaging. However, each addresses these requirements differently, resulting in interoperability issues between different standards. Thus, businesses that must use more than one standard vocabulary, in the course of doing electronic business with their business partners, are faced with the requirement to deploy multiple solutions to meet their business needs.

The ebXML Message Service specification [[EBXMLMS](#)] is the first, truly open, standards track specification that has been designed to be suitable for use by businesses for the purpose of the secure, reliable exchange of any manner of electronic business information with their partners, suppliers and customers.

The ebXML Message Service specification [[EBXMLMS](#)] has been developed under the auspices of the ebXML initiative, a joint effort of UN/CEFACT and Organization for the Advancement of Structured Information Standards ([OASIS](#)) standards organizations. This initiative was founded with the following mission statement: *To provide an open XML-based infrastructure enabling the global use of electronic business information in an interoperable, secure and consistent manner by all parties.*

## **2. Design Criteria**

The ebXML Message Service was developed with several design goals in mind:

- Leverage existing standards wherever possible
- Simple implementation
- Support for enterprises of all sizes

- Support a wide variety of communication protocols ([HTTP](#), [SMTP](#), [FTP](#), etc.)
- Support for payloads of any type (XML, EDI transactions, binary data, etc.)
- Support reliable messaging
- Must be secure

Each of these design criteria is addressed in the ensuing sections.

## **2.1. Leverage Existing Standards**

The ebXML Message Service specification [[EBXMLMS](#)] leverages existing public standards wherever possible. For a time, there was debate in the community regarding the merits of Simple Object Access Protocol ([SOAP](#)) versus ebXML's Message Service. The real issues of the debate were largely non-technical. There was broad consensus within the community that convergence between the two standards was desirable.

Effective with the version 0.98 draft of the ebXML Message Service specification [[EBXMLMS](#)], published in March of 2001, the ebXML Message Service is defined as a set of layered extensions on the [SOAP](#)1.1 and [SOAP](#)Messages with Attachments specifications. Thus, the ebXML Message Service provides the necessary extensions for security and reliability that are not addressed directly by the [SOAP](#) 1.1 specification. Use of these de facto standards leverages existing code bases, simplifies development and integration, and reinforces ebXML's commitment to convergence with the work of the W3C XML Protocol Core ([XMLP](#)) WG.

## **2.2. Simple Implementation**

The ebXML Message Service specification [[EBXMLMS](#)] has been designed to be as simple as possible. As a pragmatic issue, complex specifications are difficult to correctly implement and integrate. This can lead to interoperability issues between different vendor's solutions. Earlier attempts at electronic commerce have demonstrated this problem. As a result, the ebXML specifications have been put to the test. A special project team was established within ebXML to provide for "proof-of-concept" implementations of the specifications as a means of ensuring their

viability. The ebXML Message Service has received the most attention of the "POC" team because it was one of the earliest specifications made available for public review.

### **2.3. Transport Protocol Independence**

The ebXML Message Service specification [[EBXMLMS](#)] a message structure and protocol that is independent of the underlying transport protocol, such as [SMTP](#), File Transfer Protocol ([FTP](#)), [HTTP](#) or any other protocol capable of exchanging MIME data.

Thus, businesses are free to choose the most suitable vehicle(s) for the actual transfer of messages with their partners, suppliers and customers while maintaining a standard message structure. This feature enables the messaging service to be integrated once with the enterprise applications rather than once for each transport protocol. Transport protocol adapters can be treated as "plug-ins" for the ebXML Message Service implementation.

### **2.4. Payload Independence**

The ebXML Message Service is payload neutral, meaning that any kind of information can be reliably routed. This information can include XML documents, binary data, or EDI messages. This permits businesses to use the latest technology while also allowing them to leverage their existing infrastructure.

The ebXML Message Service meets these goals by defining a MIME packaging scheme and an XML message header structure that can envelope XML documents or other forms of business information.

Although many business transactions involve only two parties exchanging messages directly, it is not uncommon for a marketplace to act as an intermediary that connects parties. For example, a sealed bid could be sent to a broker that exchanges documents on behalf of an anonymous party. Unless the routing information is separated from the business information of a message, an intermediary must be able to parse the entire message before rerouting it to the final recipient. If the bid is truly "sealed", then it may be required to be encrypted such that even the intermediary is unable to process the information.

The ebXML Message Service MIME packaging scheme separates routing information (header information) from the business information. This enables a simplified, yet highly efficient, processing by an intermediary. The intermediary need not understand or even process the contents of the message payload. Additionally, because the message payload may need to be encrypted, such that only the designated recipient is able to decrypt the message and process the contents, this separation of header and payload becomes even more important when involving intermediaries.

## **2.5. Reliable Messaging**

Message delivery must be reliable. Any message service used for business purposes must tolerate faults in the network, the software, or the hosts on which the services are run. Error recovery, retry logic and duplicate detection must be implemented in order to guarantee message delivery. Without support for some level of reliability, businesses cannot effectively exchange information electronically with sufficient degree of certainty that the messages they send have been received, or vice versa.

Reliable messaging defines an interoperable protocol such that any ebXML Message Service implementations can reliably exchange messages. The ebXML Reliable Messaging protocol provides for the following:

- For any given message provided to the the ebXML Message Service by the sending application, one and only one copy of the message will be delivered to the receiving application
- A positive acknowledgement will be sent from the receiving ebXML Message Service to the sending ebXML Message Service indicating that the message has been received and stored persistently
- If an acknowledgment is not received the sending ebXML Message Service will either retry delivery or notify the sending application

The ebXML Message Service does not place any requirements on the reliability of the underlying transport. The service incorporates the necessary logic to achieve the level of reliability businesses require. The underlying transport need not guarantee

transmission or acknowledge transmission, so almost any available transport can be used. This permits the ebXML Message Service to be used with high-end application servers, web servers, or even email. This gives businesses a choice of solutions that are interoperable and fit their existing infrastructure and IT budgets.

## **2.6. Security**

The ebXML Message Service, by its very nature, presents certain security risks. An ebXML Message Service may be at risk by means of:

1. Unauthorized access
2. Data integrity and/or confidentiality attacks (e.g. through man-in-the-middle attacks)
3. Denial-of-Service, spoofing, bombing attacks

Each of these security risks may be addressed in whole, or in part, by the application of one, or a combination, of the countermeasures described in the ebXML Message Service specification [[EBXMLMS](#)]. The specification describes a set of profiles, or combinations of selected countermeasures, that have been selected to address key risks based upon commonly available technologies. Each of the specified profiles includes a description of the risks that are not addressed.

Application of countermeasures should always be balanced against an assessment of the inherent risks and the value of the asset(s) that might be placed at risk.

No technology, regardless of how advanced it might be, is an adequate substitute to the effective application of security management policies and practices. The ebXML Message Service is no exception. The ebXML Message Service specification [[EBXMLMS](#)] and the ebXML Technical Architecture Security specification provide some guidance as to how security management policies and practices can and should be used to minimize risks that are introduced when doing business electronically via the Internet.

In keeping with fundamental design criteria, security is addressed using open standards. The joint W3C/IETF XML Signature standard serves as the basis for

enabling an ebXML Message to be digitally signed enabling the recipient of a digitally signed ebXML Message to authenticate its source and verify that the integrity of the message has not been compromised in transit. The choice of the W3C/IETF XML Signature standard provides businesses with a wide variety of PKI environments that can be used with the ebXML Message Service. This choice can be very important to businesses.

The payload of an ebXML message can be encrypted and digitally signed such that only the intended recipient can access and verify the authenticity and integrity of contents of the message using popular MIME-based cryptographic standards such as S/MIME and PGP/MIME.

Because the ebXML Message Service has been designed to be transport and network protocol neutral, a variety of network protocol security standards such as SSL and IPSEC may be used to provide confidentiality, authentication and message integrity, thus enhancing the security countermeasures that are defined in the ebXML Message Service Specification.

Emerging security standards such as Security Assertion Markup Language ([SAML](#)), now being developed by the [OASIS](#) Security Services Technical Committee, may also be used in conjunction with the ebXML Message Service through extension of the [SOAP](#) Envelope. [SAML](#) is a standard that is being developed to enable the exchange of security assertions between enterprises in a secure manner.

### **3. Relationship to ebXML Architecture**

The ebXML Message Service provides the message exchange functionality within the ebXML Infrastructure. Within the context of ebXML as a whole, the ebXML Message Service provides the messaging interface for the ebXML Registry/Repository and its clients. In addition, the ebXML Collaboration Protocol Agreement ([CPA](#)) may be used to provide a runtime configuration mechanism for the ebXML Message Service. The [CPA](#) is the key to interoperability within ebXML. It allows two ebXML Message Service implementations to be configured with the same [CPA](#).

While the ebXML Message Service was designed to work within the overall context

of the ebXML initiative, due to the modular nature of the ebXML Technical Architecture, the ebXML Message Service can be used independently of ebXML as a whole. Software vendors can easily integrate ebXML Message Service functionality into their existing enterprise solutions. The specification is simple and straightforward making it easy to implement.

In addition, the Java API for XML Messaging (JAXM) provides an ebXML Message Service implementation that can be used to quickly integrate ebXML Message Service functionality into an application or product. The ebXML Message Service yields an interoperable mechanism that can be easily implemented and used by any business, large or small, for a variety of applications.

## **4. Overview of the ebXML Message Service**

A complete message, referred to as the Message Package, is a MIME multipart/related object. MIME types are used throughout to describe all of the contents of the Message Package. The Message Package contains two principal parts: a SOAP Message container and zero or more payload containers. The SOAP Message contains the ebXML SOAP extension elements routing information, trading partner information, message identification, and delivery semantics information. The payload is optional, and can contain any type of information that is to be exchanged between parties. The following diagram describes the structure of an ebXML message:

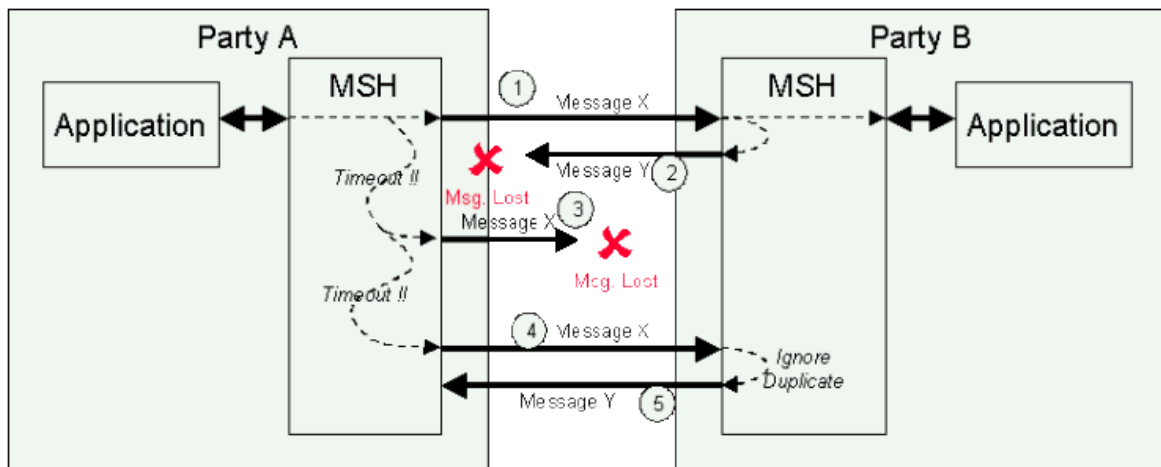


**Figure 1. Structure of an ebXML Message**

The ebXML Message Service introduces a manifest along with each message. The manifest contains references to each of the payload objects along with schema location and version information about the payload. This versioning of inner and outer layers permits the ebXML Message Service to be truly payload neutral. Because the versions are separated, an older version of the ebXML Message Service software can still route messages with newer version numbers without having to upgrade the ebXML Message Service software. Conversely, when the ebXML Message Service is versioned, the payload objects that it carries are not effected. Other standards that have not separated the payload from the message envelope and headers suffer from potential version problems. This limits the flexibility of these standards in a global context where uniform versioning is highly unrealistic.

To guarantee reliable message delivery, positive acknowledgement and persistent storage is required. Prior to sending a message, the sending ebXML Message Service will save the message in persistent storage. Once the message has been correctly received, the receiving ebXML Message Service will save the message in persistent storage and send an acknowledgement message to the sending ebXML Message Service. After the sending ebXML Message Service receives the acknowledgement, it may delete the message from persistent storage if no longer needed. If the sending ebXML Message Service does not receive acknowledgement, it can resend the message or notify the sending application that the message was unable to be delivered. The entire messaging operation is asynchronous, meaning that transmission of one message need not be completed before additional messages are sent. The following diagram describes the messaging process in detail:

If the sending ebXML Message Service does not receive acknowledgement, it will attempt a recovery sequence. The sending service will resend the message and again wait for acknowledgement. This cycle is repeated a number of times. The number of retries and retry interval can be defined as needed. The following diagram describes the recovery sequence:



**Figure 2. Reliable Messaging Recovery Sequence**

## 5. Integrating with the ebXML Message Service

It is expected that enterprise applications will leverage the ebXML Message Service through either direct integration or some type of EAI interface software. Direct integration permits seamless operation, but requires vendors to add ebXML Message Service functionality to their existing software. Until this is done, the latter option is a viable alternative.

End user software can range from large-scale EAI applications to small business accounting packages. In any implementation, the end user software must supply basic information header information and the optional payload. The ebXML Message Service will accept this information and prepare a well-formed ebXML message and route it appropriately. If an error occurs, recovery will be attempted. If the error is unrecoverable, the end user application must then handle the error in a business specific way.

## 6. Summary

The ebXML Message Service meets the needs of businesses of all sizes through a simple, interoperable, and reliable service that is easy to integrate. The use of open standards allows implementations to leverage existing code bases. This simplifies implementation. JAXM is available as a reference implementation for businesses interested in quickly adopting ebXML Message Service functionality.

The service delivers a level of reliability that guarantees at most one copy of a message is sent to the recipient and confirms receipt. This level of reliability is not possible with most standards available today. The service can operate over any transport, such as [SMTP](#) or [HTTP](#). This permits enterprises of all sizes to participate in global electronic commerce at low cost.

The ability to exchange any type of information, including XML, binary data, or EDI messages makes the ebXML Message Service highly flexible. This flexibility permits new businesses to use XML messages for documents while businesses with existing EDI infrastructure can continue to leverage their legacy systems.

Key design strategies offer significant advantage other standards. The separation of

header and payload permits easy and efficient rerouting of messages. Confidentiality and authenticity of the payload is not lost in the in the routing process. These advantages allow complex business models, that might include a number of intermediaries, to offer unique services in a global marketplace.

The modular nature of the ebXML Technical Architecture permits use of the ebXML Message Service either within the context of ebXML as a whole, or independently. Thus, ebXML meets the needs of today's businesses interested in effectively exploiting electronic commerce.

## **Bibliography**

[EBXMLMS]

[http://www.ebxml.org/specdrafts/ebXML\\_Message\\_Service\\_Specification\\_v0.92b.pdf](http://www.ebxml.org/specdrafts/ebXML_Message_Service_Specification_v0.92b.pdf)

## **Glossary**

CPA	Collaboration Protocol Agreement
FTP	File Transfer Protocol
HTTP	HyperText Transfer Protocol
OASIS	Organization for the Advancement of Structured Information Standards
SAML	Security Assertion Markup Language
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
XMLP	XML Protocol Core

## **Biography**

Christopher **Ferris**

Senior Staff Engineer

Sun Microsystems

Technology Development

Burlington

USA

Email: [chris.ferris@sun.com](mailto:chris.ferris@sun.com)

*Christopher Ferris* Christopher is a senior staff engineer in Sun's Technology Development group. He is vice-chair of the ebXML Transport Routing and Packaging project team (TR&P). Christopher has been at Sun for over ten years and has over twenty years experience in distributed IT systems architecture, design and implementation.