EBXML - QUICK GUIDE

EBXML - INTRODUCTION

Businesses inevitably interact with each other in various ways. Until recent years, many large companies used to communicate automatically through Electronic Data Interchange *EDI*, which allows two companies to communicate using predetermined signals.

The trouble with EDI is that it is very expensive and originally it was created for the mainframe world. Now ebXML is replacing EDI.

Definition

ebXML stands for **E**lectronic **B**usiness E**x**tensible **M**arkup **L**anguage. It is a global standard for electronic business that enables anyone, anywhere to do business transactions with anyone over the Internet.

Features

The features of ebXML are as follows:

-
- ebXML is an end-to-end B2B XML framework.
- ebXML is a set of specifications that enable a modular framework.
- ebXML relies on the Internet's existing standards such as HTTP, TCP/IP, MIME, SMTP, FTP, UML, and XML.
- ebXML can be implemented and deployed on virtually any computing platform.
- ebXML provides concrete specifications to enable dynamic B2B collaborations.

ebXML Vision

ebXML is designed to create a global electronic market place where enterprises of any size, anywhere can:

- find each other electronically.
- conduct business -
    - using exchange of XML messages.
    - according to standard business process sequences.
    - with clear business semantics.
    - using off-the-shelf purchased business applications.
    - according to mutually agreed trading partner protocol agreements.

Why ebXML?

- Existing B2B Frameworks are not adequate:
    - EDI and RosettaNet are too heavy-weight and too rigid.
    - BizTalk is proprietary, single-vendor, and single-platform.
- Simple Object Access Protocol *SOAP*; Web Service Definition Language *WSDL*; and Universal Description, Discovery, and Integration *UDDI* alone are not adequate:
    - WSDL does not address business collaboration.
    - SOAP in its basic form does not provide secure and reliable message delivery.

- UDDI does not provide repository capability for business objects.

- There is a growing requirement to standardize business collaborations to address the following:
  - Business processes
  - The parties involved in business collaboration and their roles
  - Exchanging XML documents in the business collaborations
  - Security, reliability, quality of service requirements of business collaboration

  All these needs are addressed by ebXML.

ebXML Founding Organizations

ebXML is a joint initiative by UN/CEFACT and OASIS.

UN/CEFACT:

- It stands for United Nations Centre for Trade Facilitation and Electronic Business.
- It maintains the UN/EDIFACT standards for Electronic Data Interchange *EDI*.

OASIS:

- It stands for Organization for Advancement of Structured Information Standards.
- It creates and maintains XML interoperability specifications, broad industry support.

EBXML - ARCHITECTURE

By definition, the iterative life cycle of **B2B collaboration** includes the following steps:

- Process Definition
- Partner Discovery
- Partner Sign-up
- Electronic Plug-in
- Process Execution
- Process Management
- Process Evolution

The overall ebXML specifications are intended to cover almost the entire process of B2B collaboration and are designed to meet the needs described above.

The ebXML architecture as defined by the ebXML team provides:

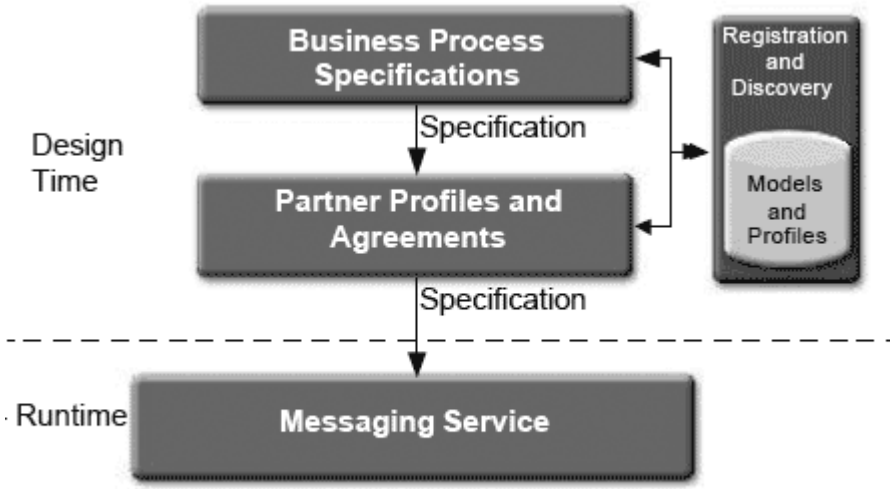
- A way to define business processes and their associated messages and content.
- A way to register and discover business process sequences with related message exchanges.
- A way to define company profiles.
- A way to define trading partner agreements.
- A uniform message transport layer.

Consequently, the technical architecture of ebXML is composed of five modules:

- Business Process Specifications
- Partner Profile and Agreements
- Registry and Repository
- Core Components

- Messaging Service

These modules will be covered in the next five subsequent chapters. The diagram diagram shows the simplified architecture of ebXML:



ebXML Architecture

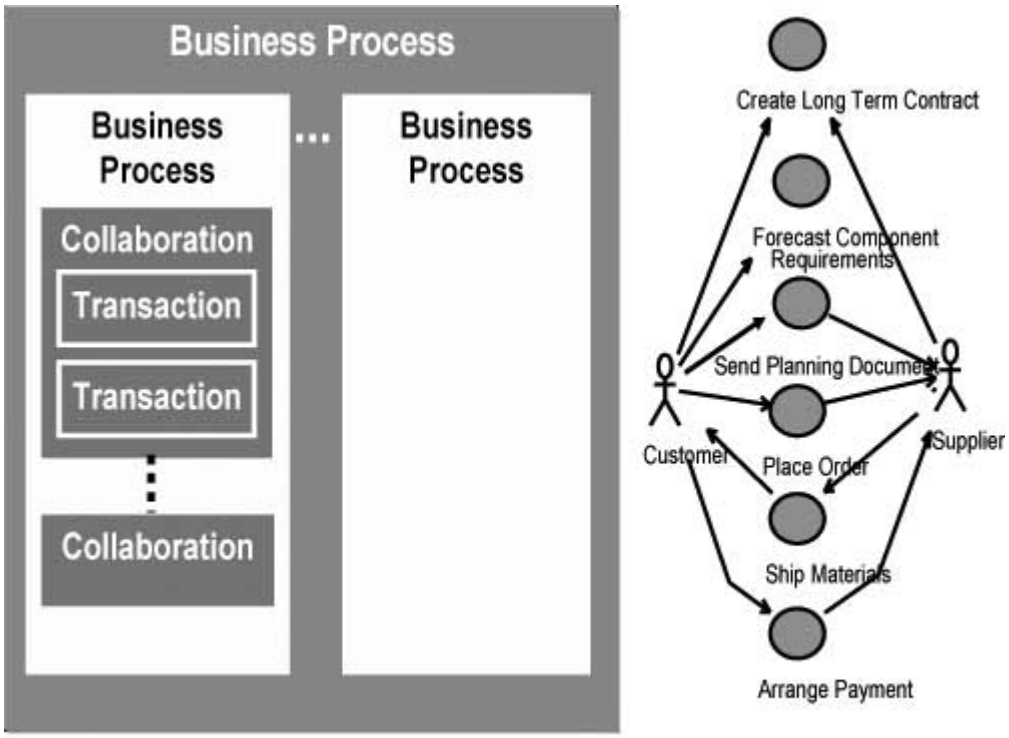
EBXML - BUSINESS PROCESS

A Business Process is something that a business does, such as buying computer parts or selling a professional service. It involves the exchange of information between two or more trading partners in some predictable way.

The specifications for business process definition enable an organization to express its business processes so that they are understandable by other organizations. It enables the integration of business processes within a company or among multiple companies.

The **ebXML Business Process Specification Schema** ***BPSS*** provides the definition of an XML document that describes how an organization conducts its business. An ebXML BPSS is a declaration of the partners, roles, collaborations, choreography, and business document exchanges that make up a business process.

Following diagram gives a conceptual view of Business Process.



Business Collaborations

A Business Collaboration is a choreographed set of business transaction activities, in which two trading partners exchange documents.

The most common one is a Binary Collaboration, in which two partners exchange documents. A Multiparty Collaboration takes place when information is exchanged between more than two parties.

Multiparty collaborations are actually choreographed Binary Collaborations.

At its lowest level, a business collaboration can be broken down into business transactions.

Business Transactions

A Business Transaction is the atomic level of work in a business process. It either succeeds or fails completely.

Business transactions are transactions in which trading partners actually transfer business documents.

Business Document Flows:

A business transaction is realized as a Business Document flows between requesting and responding roles. There is always a requesting business document, and optionally a responding business document, depending on the desired transaction semantics, for example, one-way notification vs. two-way conversation.

Actual document definition is achieved using the ebXML core component specifications, or by some methodology external to ebXML but resulting in a DTD or Schema that an ebXML business process specification can point to.

Choreography:

Choreography is expressed in terms of states and the transitions between them. A business activity is known as an abstract state, with business collaborations and business transaction activities known as concrete states. The choreography is described in the ebXML business process specification schema using activity diagram concepts such as start state, completion state etc.

Business Documents

Business documents are composed of business information objects, or smaller chunks of information that have previously been identified.

These chunks, or components, do not carry any information, of course. They are merely structures, such as an XML schema or a DTD, that define information and presentation. The end result is a predictable structure into which information is placed, so that the receiver of the final document can interpret it to extract the information.

Business Process Specification Example

A partial example of business process specification is given below:

```
<BusinessTransaction name="Create Order">
    <RequestingBusinessActivity name=""
        isNonRepudiationRequired="true"
        timeToAcknowledgeReceipt="P2D"
        timeToAcknowledgeAcceptance="P3D">
    <DocumentEnvelope BusinessDocument="Purchase Order"/ >
    </RequestingBusinessActivity>
    <RespondingBusinessActivity name=""
        isNonRepudiationRequired="true"
        timeToAcknowledgeReceipt="P5D">
    <DocumentEnvelope isPositiveResponse="true"
```

```
        BusinessDocument="PO Acknowledgement"/>
    </DocumentEnvelope>
    </RespondingBusinessActivity>
</BusinessTransaction>
```

Conclusion

A business process specification:

- Describes collaboration between two partners
- Defines roles, relationships and responsibilities
- Defines choreography of business documents
- Expressed in platform and vendor neutral format
- Can be modeled with UMM *UN/CEFACTModelingMethodology*
- Formally described by Business Process Specification Schema *BPSS*
- Referenced by CPP and CPA.
- Refers to business document definitions.

EBXML - CPP AND CPA

Collaboration Protocol Profile

A Collaboration Protocol Profile *CPP* provides all the necessary information on how a particular trading partner intends to do electronic business. A CPP defines the following attributes of a trading partner:

- Business capabilities through business process.
- The role *buyerorinsurer* they play within a collaboration.
- Delivery channels and transport protocols. *HTTP*, *SMTP*, *etc.*
- Packaging way of business documents.
- Security constraints *SSL*, *DigitalCertificates*.
- Per-party configuration to business process specifications.

A CPP is stored in ebXML registry with a Globally Unique Identifier *GUID* and business partners can find each other's CPP through registry.

The information within the CPP is available to be searched on, so a potential trading partner can determine whether the organization has the capabilities to do business.

Structure of a CPP

CPP defines namespaces on its root element and a version to distinguish any subsequent changes. The structure of a CPP consists of a root Collaboration Protocol Profile element with following elements:

- **PartyInfo:** The PartyInfo element provides information about the organization.
- **Packaging:** The Packaging element provides information about the way in which messages are actually constructed. Messages are processed as SOAP Messages.
- **Signature:** Optional part of the document
- **Comment elements:** can be included.

```
<CollaborationProtocolProfile
xmlns="http://www.ebxml.org/namespaces/tradePartner"
```

```
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:xlink="http://www.w3.org/1999/xlink"
version="1.1">
<PartyInfo>
    ...
    <!--REQUIRED, Repeatable-->
...
</PartyInfo>
<Packaging >
    ...
    <!--REQUIRED-->
    ...
<Packaging>
<ds:Signature>
    ...
    <!--OPTIONAL-->
    ...
</ds:Signature>
<Comment>
    ...
    <!-- OPTIONAL -->
    ...
</Comment>
</CollaborationProtocolProfile>
```

Trading Partner Agreement

A Trading Partner Agreement *TPA* is a contract defining both the legal terms and conditions and the technical specifications for both partners in the trading relationship. A CPA is derived from CPP's of trading partners.

The rules specified by the electronic TPA are independent of the business processes at either party. A technical description of the terms and conditions from the TPA is expressed in an XML document, which configures each IT systems to operate under the agreement rules.

TPA properties include its name, partner names, starting and ending dates, roles, and other parameters. Typically, one party generates a CPA and offers it to the other party for approval. Once both sides have reached agreement, they each take an electronic copy of the same CPA and use it to configure their systems.

The CPA may also be added to the registry for reference, but this is not a standard requirement.

Structure of a CPA

CPA defines namespaces on its root element and a version to distinguish any subsequent changes. The structure of a CPP consists of a root Collaboration Protocol Agreement element along with the following elements:

- **Start and End:** These elements represent, in coordinated universal time, the beginning and end of the period during which this CPA is active.
- **PartyInfo:** The PartyInfo element provides information about the organization. Here PartyInfo elements are included for both parties involved in the agreement.
- **Packaging:** The Packaging element provides information about the way in which messages are actually constructed. Messages are processed as SOAP messages.
- **Signature:** Optional part of the document.
- **Comment elements:** can be included.

```
<CollaborationProtocolAgreement
xmlns="http://www.ebxml.org/namespaces/tradePartner"
xmlns:ds = "http://www.w3.org/2000/09/xmldsig#"
xmlns:xlink = "http://www.w3.org/1999/xlink"
cpa
version="1.7">
```

```
<Status value = "proposed"/>
<Start>1998-04-07T18:50:00</Start>
<End>1999-04-07T18:50:00</End>
<ConversationConstraints invocationLimit = "150"
concurrentConversations = "10"/>
<PartyInfo>
    ...
    <!--REQUIRED, repeatable-->
    ...
</PartyInfo>
<PartyInfo>
    ...
    <!--REQUIRED, repeatable-->
    ...
    </PartyInfo>
<Packaging >
    ...
    <!--REQUIRED, repeatable-->
    ...
</Packaging>
<ds:Signature>
    <!--OPTIONAL-->
</ds:Signature>
<Comment xml:lang="en-gb">
    <!--OPTIONAL-->
</Comment>
</CollaborationProtocolAgreement>
```

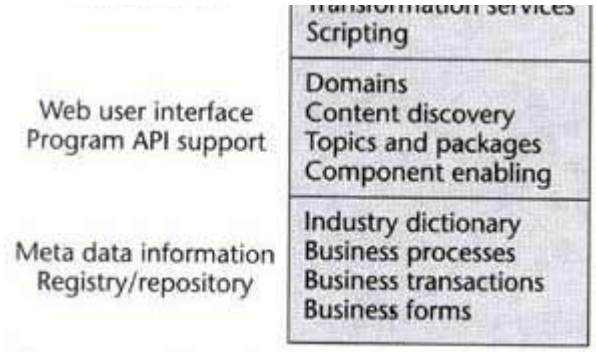
EBXML - REGISTRY AND REPOSITORY SERVICE

What is Registry and Repository:

An ebXML registry serves as the index and application gateway for a repository to the outside world, and it contains the API that governs how parties interact with the repository. An ebXML repository is the holder of the components.

- The ebXML registry is central to the ebXML architecture.
- The registry can also be viewed as an API to the database of items that supports e-business with ebXML.
- The ebXML registry serves as a database for sharing relevant company information for ebXML business transactions, such as corporate capabilities, business process, technical blueprints, order forms, invoices, and so on.
- Items in the repository are created, updated, or deleted through requests made to the registry.
- Repositories provide trading partners with the shared business semantics.
- The ebXML registry is an interface for accessing and discovering shared business semantics.
- The registry interface is designed to be independent of the underlying network protocol stack, such as HTTP or SMTP over TCP/IP.

The registry provides a stable, persistent store of submitted content, which includes XML schema and documents, process descriptions, core components, context descriptions, UML models, information about parties, and even software components. This can be represented as a software stack of services, as shown below:



Goals of ebXML Registry

The goal of ebXML registry is to enable information sharing between interested parties for the purpose of business process integration between them.

Benefits of ebXML registry

An ebXML registry provides the following benefits:

- Discovery and maintenance of registered content.
- Support for collaborative development, where users can create XML content and submit it to the registry for use and potential enhancement by the authorized parties.
- Persistence of Web Services Business Process Execution Language $WS-BPEL$, WSDL, and business documents during interactions between trading partners.
- Secure version control of registered content.
- Federation of cooperating registries to provide a single view of registered content by seamless querying, synchronization, and relocation of registered content.
- Event notification via email or Web services.

Compliance

According to the ebXML Registry Services Specification, a registry implementation complies with the ebXML specification if it meets the following conditions:

- It supports the ebXML Registry Information Model.
- It supports the syntax and semantics of the registry interfaces and security.
- It supports the ebXML registry DTD.
- Support of the syntax and semantics of SQL query in the registry is optional.

A registry client implementation complies with the ebXML specification if it meets the following conditions:

- It supports the ebXML CPA and bootstrapping process.
- The syntax and the semantics of the registry client interfaces.
- The ebXML error message DTD.
- The ebXML registry DTD.

Registry Objects and Metadata

Registry objects

Refers to an object that is submitted to registry for storage and safekeeping

- called 'Repository item'

- XML document or DTD, business process models, CPPs, etc.

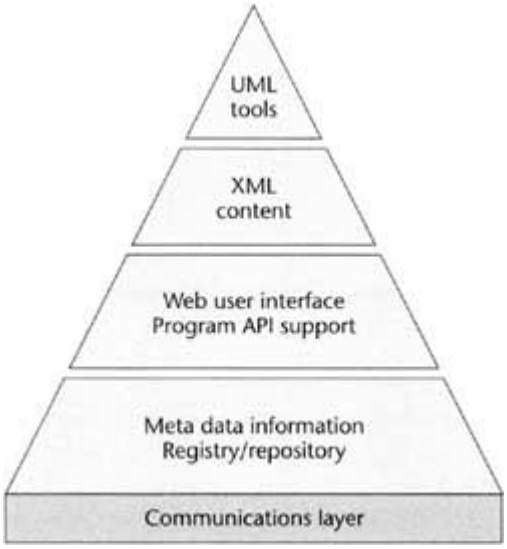
Metadata

- It is used by registry to classify and manage registry objects.
- It is represented by Registry Entry

Registry Information Model *RIM*

The Registry Information Model *RIM* provides a high-level blueprint for metadata in the ebXML registry. This can be represented as a software stack of services or as a service pyramid as shown in the figure below. The elements of the information model represent meta data about the content, not the content itself in the repository. The registry information model defines the types of objects stored and organized in the registry.

The information model is a roadmap to the type of meta data and the relationships between metadata. The registry information model may be mapped to a relational database schema, object database schema, or some other physical schema.



EBXML - CORE COMPONENTS

"A Core Component captures information about a real world business concept, and relationships between that concept and other business concepts. A Core Component can be either an individual piece of business information, or a family of business information pieces. It is core because it occurs in many different areas of industry/business information exchange"

> *...Definition form xbXML simplified by Eric Chiu*

A core component is a basic, reusable building block that contains information representing a business concept. Some examples of core components for parts of a purchase order are Date of Purchase Order, Sales Tax, and Total Amount.

In general, core components are used in many different domains, industries, and business processes. In the ebXML environment, core components are the building blocks for XML semantics and business vocabulary that are used in messages and documents.

From a specific business document in a business process, we can refer to a core component, which holds a minimal set of e-business information. If the business processes are the verbs in e-business terms, the core components represent the nouns and adjectives.

A core component can be used across several business sectors, but it also can become context-specific to a business domain, such as an individual industry area.

A core component works with a registry, since it is storable and retrievable using a standard ebXML registry. A central core component library serves as a reference document for common

business practices across industry business processes.

Tools and References

The list of essential references and tools for core components provided by ebXML for the business and technical analyst is as follows:

- **Context and the Re-usability of Core Components**: This document contains context definitions, the sources of classification value lists, and a pictorial model depicting the relationships of core component and context descriptor.
- **Catalog of Context Drivers:** This document provides a catalog of context drivers.
- **Document Assembly and Context Rules:** This describes the procedures and schemas for assembling documents using contextually driven core components.
- **Core Components Dictionary:** This document is divided into sections. Each section begins with the information on the applicable category and core component type.
- **Core Components Editor and Browser:** These tools help analysts browse existing core components and integrate them to define the format of the XML messages exchanged between trading partners and to properly define and apply the context rules.

Core Components Examples:

- Core component A:
  - Vendor *Industry*1
  - Manufacturer *Industry*2
  - Supplier *Industry*3
- Core component B:
  - Distributor *Industry*1
  - Wholesaler *Industry*2
  - Merchant *Industry*3
- Core component C:
  - Store *Industry*1
  - Outlet *Industry*2
  - Retailer *Industry*3

Conclusion

Core Components are -

- Uniquely identifiable.
- Reusable low-level data structures
  - -e.g., party, address, phone, date, currency
  - -Context-sensitive
- Used to define business process and information models.
- Facilitates interoperability between disparate systems.
- A core component in ebXML can contain another core component.

EBXML - MESSAGING SERVICE

A complete message is called the message package, which is a Multipurpose Internet Mail

Extensions *MIME* object. The message package contains two principal parts:

- **SOAP Message Container:** This is required part of the message and contains the SOAP extension elements for ebXML, such as routing information, trading partner information, message identification, and delivery semantics information.
- **Payload Containers:** This is optional part of the message and can contain any type of information that is to be exchanged between parties.

Messaging Design Criteria

According to the messaging service specification, the design goals for the ebXML message service are to:

- Leverage existing standards wherever possible.
- Be simple to implement.
- Support enterprises of all sizes.
- Support a wide variety of communication protocols *HTTP*, *SMTP*, *FTP*, *etc.*
- Support payloads of any type *XML*, *EDItransactions*, *binarydata*, *etc.*
- Support reliable messaging.
- Ensure security.

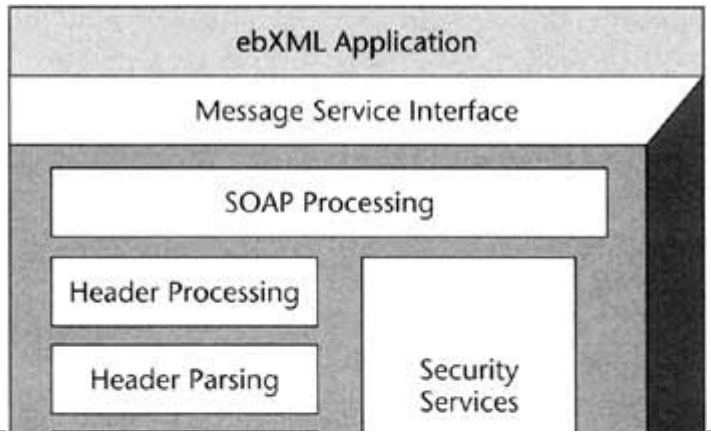
Messaging Architecture

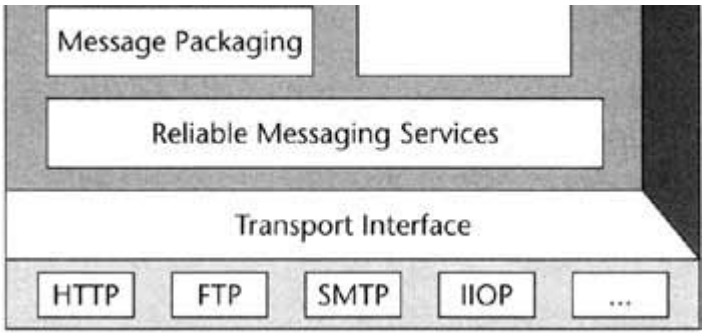
The ebXML message service was designed to work within the overall context of the ebXML initiative. However, the ebXML technical architecture is modular, and the message service can be used independently of ebXML.

The ebXML message service has three logical architectural levels between the business application and the network protocols:

- **The Message Service Interface *MSI*:** It is an application interface for business applications to invoke message handler functionality for sending and receiving messages. Similar to ODBC, JDBC, and other abstract service interfaces, it exposes the message handler functionality as a defined set of APIs for business application developers.
- **The Message Service Handler *MSH*:** It has basic services, such as header processing, header parsing, security services, reliable messaging services, message packing, and error handling.
- **The Message Transport Interface *MTI*:** It is designed to send messages over various networks and application-level communication protocols. The transport interface transforms ebXML specific data to other forms carried by network services and protocols. This involves a complete exchange between two parties, piggybacking on top of existing protocols in the network stack.

The ebXML Messaging Architecture is shown in the following diagram.





Message Formatting:

An ebXML message has to be formatted according to the ebXML message service specification and must conform to the MIME syntax, format, and encoding rules. The definition of the XML elements are provided by an XML schema, which extends SOAP to define the ebXML message header, trace header, manifest, status, and acknowledgment.

Conclusion

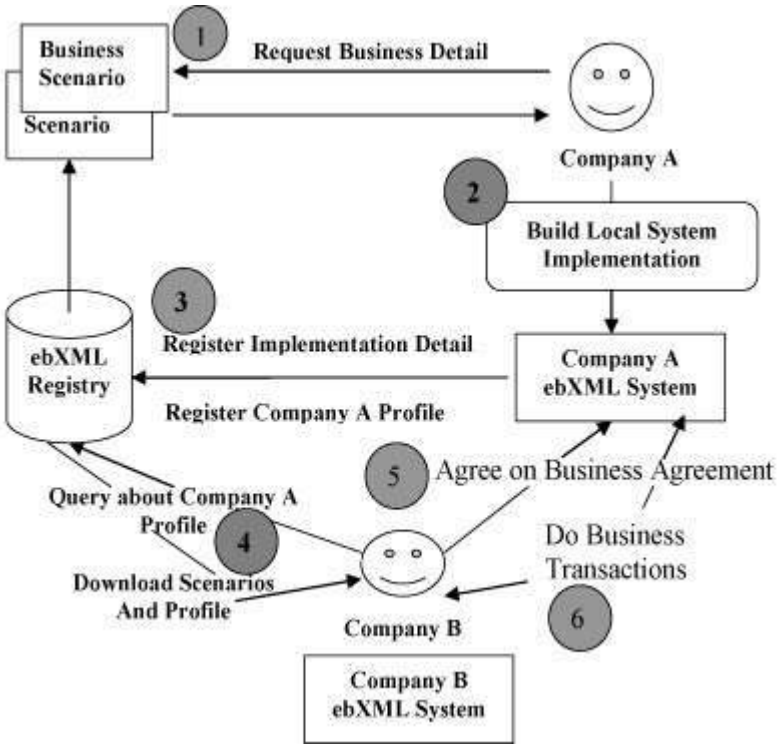
An ebXML message has to be formatted according to the ebXML Message Service Specification and must conform to the MIME syntax, format, and encoding rules. The definition of the XML elements are provided by an XML schema, which extends SOAP to define the ebXML message header, trace header, manifest, status, and acknowledgment.

The ebXML messaging -

- Uses SOAP with Attachments as payload envelope.
- Runs over various communication protocols such as HTTP, SMTP, FTP.
- Supports higher-level semantics needed in business transactions. *SecuirtyandReliability*

EBXML - USAGE EXAMPLE

The following diagram shows an ebXML scenario, which makes it easy to pick up the concept of ebXML. The example is taken from the Technical Architecture Specification.



The example shows how organizations prepare for ebXML, search for new trading partners and then engage in electronic business.

- Company A browses the ebXML registry to see what is available online. At best, company A can reuse all the existing business processes, documents, and core components common to its industry that are already stored in the ebXML registry. Otherwise, company A designs the missing parts, stores them in the ebXML registry and makes them available for its industry partners.
- Company A decides to do electronic business the ebXML way and considers implementing a local ebXML compliant application. An ebXML Business Service Interface *BSI* provides the link between the company and the outside ebXML world. The company has to create a Collaboration Protocol Profile *CPP* which describes the supported business process capabilities, constraints and technical ebXML information such as choice of encryption algorithms, encryption certificates, and choice of transport protocols.
- Company A submits its CPP to ebXML registry. From that point on, company A is publicly listed in the ebXML registry and is likely to be discovered by other companies querying for new trading partners.
- Company B is already registered at the ebXML registry and is looking for new trading partners. Company B queries the ebXML registry and receives the CPP of company A. Company B then has two CPP's: Company A's CPP and its own. The two companies have to come to an agreement on how to do business, which is called a Collaboration Protocol Agreement *CPA* in the ebXML terminology. Company B uses an ebXML CPA formation tool to derive a CPA from the requirements of the two CPPs
- In this scenario, company B communicates with company A directly and sends the newly created CPA for acceptance to company A. Upon agreement of the CPA by company A, both companies are ready for electronic business.
- The companies then use the underlying ebXML framework and exchange business documents conforming to the CPA. This means that both companies follow the business processes defined in the CPA