Creating A Single Global Electronic Market

ebXML - Complex Content Exchange

Proof of Concept Proposal - Working Draft

Version

ebXML POC Proposal v0.1

Authors

Hicham Bahi	Documentum, Inc.
Una Kearns	Documentum, Inc.

Contributors

***** DRAFT *****

Overview

The main purpose of this Proof of Concept proposal is to validate the ebXML architecture against scenarios which deal with exchange of multiple pieces of related content -- both structured and unstructured. Payloads of this nature address unique requirements such as payload sizes and the ability to correctly manage relationships between both structured and unstructured information.

The business scenarios that support this can be found across multiple applications and verticals, a perfect example is Catalog and Product Information distribution and exchange as seen from the RosettaNet PIPs (e.g. Distributing and Updating Product Information). Most of the examples in the previous POCs appear to have dealt with simpler payloads.

This proposal assumes that it will be merged with other proposals dealing with all aspects of the ebXML architecture including CPP, Registry and Repository, Transport, Packaging and Routing.

Demo Scenario

Business Process

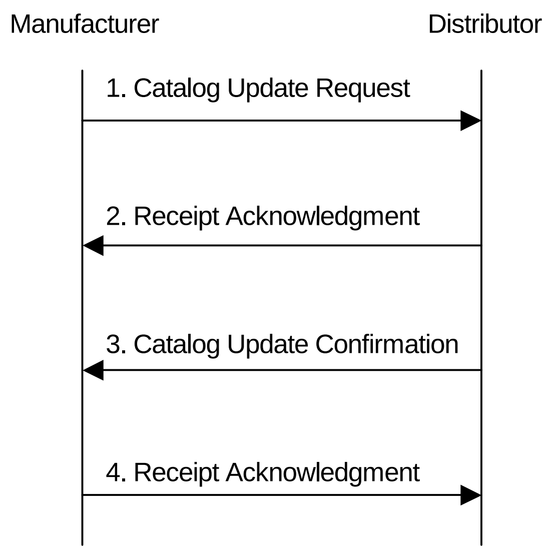
The business process is the exchange of catalog information between a manufacturer and a distributor. This process is common to many verticals. One possible demo scenario is RosettaNet PIP2A1 which focuses on the Information Technology supply chain. Of course, this scenario could easily be extended to other roles and applied to different verticals.

For example:

1. Addition of Roles: Exchange between manufacturers and multiple types of Distribution Channels (e.g. Exchanges, Online Websites, Resellers).

2. Associated CPAs can be developed to accommodate common business scenarios dealing with different payload schemas and formats required for different agreements.

At a minimum, the engagement between the manufacturer and distributor would include the following steps:


Manufacturer
Distributor
1. Catalog Update Request
2. Receipt Acknowledgment
3. Catalog Update Confirmation
4. Receipt Acknowledgment

In step 1, the manufacturer sends a catalog update request to the distributor. The request includes structured information (catalog data in a pre-defined XML schema) and unstructured information (data sheets, pictures, …).

Step 2 is the ebXML transport level acknowledgment.

In step 3, the distributor sends a catalog update confirmation indicating that the distributor has updated his system.

Step 4 is the ebXML transport level acknowledgment.

Business Data

The business data will consist of a structured catalog XML instance that conforms to a pre-defined schema (such as RosettaNet PIP2A1) and supporting unstructured documentation associated with each item described in the catalog (such as a PDF data sheet and a JPEG picture). The catalog XML instance will contain references to the supporting documentation. Different approaches of packaging related content can be explored. For example, base64 encoded inline in the XML instance, or externally referenced and included in the payload.

ebXML Technical Details

Packaging

The goal of the demonstration is to show the ebXML support for complex payloads. The payload will be a nested multipart MIME with the catalog XML instance as the first part followed by the supporting documentation (PDF, JPEG).

Values in ***bold italic blue*** are generated dynamically by the participant according to the pattern shown. Values in **bold** are fixed for a particular message, but may change for different messages. Values in normal text are fixed and common to all messages. An example is shown below:

```
Content-Type: multipart/related;
              type=”application/vnd.eb+xml”;
              charset="iso-8859-1";
              version=0.1;
              boundary=level1_boundary

--level1_boundary
Content-ID: uid@originator-domain
Content-Type: application/vnd.eb+xml

<?xml version="1.0" encoding="UTF-8"?>
<ebXMLHeader>
 ....
</ebXMLHeader>
--level1_boundary
Content-ID: uid@originator-domain
Content-Type: multipart/mixed; boundary=level2_boundary

--level2_boundary
....
--level2_boundary
....
--level2_boundary--


--level1_boundary--
```

Header

Below are sample headers for the request and acknowledgment messages. Values in ***bold italic blue*** are generated dynamically by the participant according to the pattern shown. Values in **bold** are fixed for a particular message, but may change for different messages. Values in normal text are fixed and common to all messages. Note that the term “originator” refers to the creator and sender of an individual message instance and does not necessarily correspond to the “requester” role in the process.

Catalog Update Request

```
<?xml version ="1.0"?>
<!DOCTYPE ebXMLHeader SYSTEM "...">
<ebXMLHeader xmlns = “http://www.ebxml.org/namespaces/messageHeader"
             Version = "1.0"
             MessageType = "Normal">
    <Manifest>
        <Reference id=”part1” xlink:href=”cid:[cid of the payload MIME
part]”>
            <Description>Product Resource Update</Description>
            <Schema location=”http://www.rosettanet.org/pip/pip2a1.dtd”
version=”1.0”>
        </Reference>
        <Reference id=”part2” xlink:href=”cid:[cid of the payload MIME
part]”>
            <Description>Data Sheet</Description>
        </Reference>
        <Reference id=”part3” xlink:href=”cid:[cid of the payload MIME
part]”>
            <Description>Picture</Description>
        </Reference>
        <Reference ...
        ...
         </Reference>
    </Manifest>
    <Header>
        <From>
            <PartyId type="urn:duns.com">[requester-DUNS-
number]</PartyId>
        </From>
        <To>
            <PartyId type="urn:duns.com">[responder-DUNS-
number]</PartyId>
        </To>
        <CPAId>http://regrep.org/cpa/SampleCPA</CPAId>
        <ConversationId>conversation_id</ConversationId>
        <Service type=”RNIF”>PIP2A1</Service>
        <Action>ProductResourceUpdate</Action>
        <MessageData>
            <MessageId>uid@requester-domain</MessageId>
            <TimeStamp>CCYYMMDDThhmmss.sssZ</TimeStamp>
        </MessageData>
        <QualityOfServiceInfo deliverySemantics=”BestEffort”
deliveryReceiptRequested=”UnSigned” syncReplyMode=”False”>
    </Header>
    <RoutingHeaderList>
        <RoutingHeader>
            <SenderURI>Sender URI as defined in CPA</SenderURI>
            <ReceiverURI>Receiver URI as defined in CPA</ReceiverURI>
            <TimeStamp>CCYYMMDDThhmmss.sssZ</TimeStamp>
        </RoutingHeader>
    </RoutingHeaderList>
</ebXMLHeader>
```

Receipt Acknowledgement

```
<?xml version ="1.0"?>
<!DOCTYPE ebXMLHeader SYSTEM "...">
<ebXMLHeader xmlns = “http://www.ebxml.org/namespaces/messageHeader"
             Version = "1.0"
             MessageType = "Normal">
    <Header>
        <From>
            <PartyId type="urn:duns.com">[responder-DUNS-
number]</PartyId>
        </From>
        <To>
            <PartyId type="urn:duns.com">[requester-DUNS-
number]</PartyId>
        </To>
        <CPAId>http://regrep.org/cpa/SampleCPA</CPAId>
        <ConversationId>conversation_id</ConversationId>
        <Service type=”RNIF”>PIP2A1</Service>
        <Action>ProductResourceUpdate</Action>
        <MessageData>
            <MessageId>uid@requester-domain</MessageId>
            <TimeStamp>CCYYMMDDThhmmss.sssZ</TimeStamp>
            <RefToMessageId>MessageId of initial request</RefToMessageId>
        </MessageData>
        <QualityOfServiceInfo deliverySemantics="BestEffort"
deliveryReceiptRequested="UnSigned" syncReplyMode="False">
    </Header>
    <RoutingHeaderList>
        <RoutingHeader>
            <SenderURI>Sender URI as defined in CPA</SenderURI>
            <ReceiverURI>Receiver URI as defined in CPA</ReceiverURI>
            <TimeStamp>CCYYMMDDThhmmss.sssZ</TimeStamp>
        </RoutingHeader>
    </RoutingHeaderList>
    <Acknowledgment type="DeliveryReceipt" signed="False">
        <TimeStamp>CCYYMMDDThhmmss.sssZ</TimeStamp>
    </Acknowledgment>
</ebXMLHeader>
```

Payload

The payload consists of a catalog XML instance and supporting documentation (PDF, JPEG). The catalog XML instance should conform to a pre-defined Schema. One option is to use a schema based on RosettaNet PIP2A1. Other suggestions are welcome.

Transport

Both SMTP and HTTP could be used to demonstrate the scenario.

Collaboration Protocol Agreements

Participants will agree on a CPA to define the details of communication.

Other ebXML Architecture Components

As already mentioned the main purpose of this proposal is the inclusion of complex payloads in an end-to-end ebXML POC -- so it is expected that Registry and Repository POC demonstration would also benefit from this type of scenario.