XEP-0056: Business Data Interchange

Ulrich Staudinger
mailto:chicago5@gmx.de
xmpp:uls@jabber.org

2002-11-18
Version 0.3

Status	Type	Short Name
Deferred	Standards Track	N/A

This document defines a way to transmit ebXML messages, ANSI X.11, EDIFACT/UN, and SAP iDoc over Jabber/XMPP.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

ANSI X.12, EDIFACT/UN and ebXML are all standards. This document describes a fundamental approach to transmit messages that conform to these standards.

2 Transmitting ebXML via XMPP

2.1 Introduction to ebXML

EbXML [1], a subset of XML that is defined through OASIS.org, is quickly becoming a de-facto standard for electronic, unified, inter and intra business process communication and for business process specification. The direct adjacency to UN/CEFACT EDIFACT, especially object-oriented EDI, make the widespread use of ebXML for automatic business transaction negotiation, process definition, etc. quite likely.

2.2 Protocol

ebXML Messages SHALL be transmitted within Jabber IQ chunks. The value of the 'type' attribute SHALL be 'set' and the chunk SHALL contain a <query/> child element with the namespace declaration set to 'http://jabber.org/protocol/ebxml'. The query element MUST contain the pure ebXML message. No SOAP envelope SHALL exist around the ebXML message. Reception of an ebXML iqed message must be acknowledged with an iq chunk including an empty query tag of the same namespace, the iq tag SHALL have the same ID.

Listing 1: ebXML Envelope

```
<iq from='a@b.c' to='d@e.f' type='set' id='some'>
  <query xmlns='http://jabber.org/protocol/ebxml'>
    [data]
  </query>
</iq>
```

EbXML information is always transmitted in this envelope. No transformation of native ebXML tags into native Jabber tags is performed (e.g., ebXML reception receipt into Jabber reception receipt). The business logic, on top of Jabber transport logic, has to parse incoming IQ chunks and forward received ebXML information to the ebXML Messaging Service. The business logic has as well to pack the ebXML messages into IQ chunks and invoke the message delivery.
Although a complex business transaction between two or more Trading Partners might require a payload that contains an array of business documents, binary images, or other related Business Information' [2], only one ebXML message can be (at the moment) transmitted

[1] http://www.ebxml.org
[2] Walsh. 2002. *ebXML: The Technical Specifications*; p. 69

at a time in one message. However, the ebXML Message MAY contain payload in it's own payload envelope.
It has to be noted: The karma restriction of XMPP, implied on clients, makes transmission of large amounts of payload (at the moment) to services or other clients from client side nearly impossible. However, components' karma is not restrained.

2.3 Examples

Listing 2: ebXML Query from a client to a Gateway

```
<iq to='ebxml.gateway.com' type='set' id='f0a0f0'>
  <query xmlns='http://jabber.org/protocol/ebxml'>
    <RegistryEntryQuery>
      <RegistryEntryFilter>
        objectType EQUAL "CPP" AND status EQUAL "Approved"
      </RegistryEntryFilter>
      <HasClassificationBranch>
        <ClassificationNodeFilter>
          id STARTSWITH "urn:urn:spsc:321118'
␣␣␣␣␣␣␣␣</ClassificationNodeFilter>
␣␣␣␣␣␣</HasClassificationBranch>
␣␣␣␣</RegistryEntryQuery>
␣␣</query>
</iq>
```

The previous example transmits a query to a database with an ebXML interface. A requester searches for all companies dealing with 'Integrated curcuit components', here with UNSPSC code '321118'.

Listing 3: ebXML component OK from component to client

```
<iq to='client@server.com' from='ebxml.gateway.com' type='result' id='
    f0a0f0'>
  <query xmlns='http://jabber.org/protocol/ebxml'/>
</iq>
```

This prior xml code is the result which every recieving unit has to send, after successfully recieving the iq chunk. Note, that this does not mean, the ebxml chunk has been accepted, parsed, or whatever, this is only on jabber level.

3 Transmitting ANSI X.12 via XMPP

3.1 Introduction

ANSI X.12 is a character oriented protocol. X.12 is separated into segments. Segments are separated by an asterisk [3].

3.2 Protocol

ANSI X.12 messages SHALL be delivered in a query tag of an IQ chunk. The IQ chunk's type attribute SHALL be set to 'set'. The query's namespace attribute SHALL be set to 'http://jabber.org/protocol/ansi_x.12'. The query tag shall contain the ANSI X.12 message. The ANSI X.12 message SHALL NOT be Base64 encoded, but SHALL be XML aware.

3.3 Example

```
<iq type='set' id='ANY' to='ANSI_X12Gate@way.com'>
  <query xmlns='http://jabber.org/protocol/ansi_x.12'>
  ST*850*000000101~
  BEG*00*NE*123456789*991125**AC~
  N1*BT***0111213~
  N1*ST***5566789~
  PO1*1*250*KGM*15.3*SR*EAN*MY1001~
  CTT*1*2~
  SE*7*000000101~
  </query>
</iq>
```

4 Transmitting UN/EDIFACT via XMPP

4.1 Introduction

EDIFACT/UN is very similar to X.12 and differs only in the meaning of tags and in some syntactical details.

4.2 Protocol

EDIFACT/UN messages SHALL be delivered in a query tag of an IQ chunk. The IQ chunk's type attribute SHALL be set to 'set'. The query's namespace attribute SHALL be set to 'http://jabber.org/protocol/edifact'. The query tag SHALL contain the UN/EDIFACT message.

[3] Angeli, Streit, Gonfaloniere. 2000. *The SAP R/3 Guide to EDI and interfaces*; p. 133

The UN/EDIFACT message SHALL NOT be Base64 encoded.

5 Transmitting SAP iDoc via XMPP

5.1 Introduction

SAP Systems can create IDocs as receipts of transactions. These receipts can be used within other EDI systems, or within the SAP system. Of course transmission of IDocs can take place through XMPP as well.

5.2 Protocol

SAP IDocs SHALL be transmitted in a query tag of an IQ chunk. The IQ chunk's type attribute SHALL be 'set. The query's namespace attribute SHALL be set to 'http://jabber.org/protocol/sap_idoc'. The query tag SHALL contain the iDoc, XML aware.

5.3 Example

```
<iq type='set' id='ANY' from='sap_idoc_gate@way.com'>
  <query xmlns='http://jabber.org/protocol/sap_idoc'>
    EDI_DC03500000000000000001301230C30USDEL4111SAPI10
    ...
  </query>
</iq>
```