Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

IBX SOAP 1.2 Envelope Specification

Document Owner: Christian Druschke
Application Management Central Applications

February 2011

Company Public Information

TOGETHER. FREE YOUR ENERGIES

Table of Contents

1. Introduction

Integrated trading partners can send/receive electronic orders to/from the IBX Connect transaction platform.

The IBX Platform's standard format of electronic business documents is xCBL 3.5 – the standard SOAP envelope version is based on SOAP 1.2.

2. Purpose

The purpose of this document is to provide details about the SOAP 1.2 envelope used within IBX Connect as standard interface for exchanging business documents.

SOAP is an XML based standard. Exhausting documentation for SOAP 1.2 can be found at www.w3.org. In general IBX Connect supports SOAP 1.2 as a standard for exchanging business documents with integrated trading partners.

IBX Connect expects the information provided in this document. The data inside the general structure is partly dependent from the respective type of document.

The documentation consists of three parts.

- Description of the SOAP Envelope for the business document.
- Description of the surrounding MIME-structure if attachments are included
- Description of IBX' synchronous SOAP-Response

3. SOAP Basics

A valid SOAP Message is a well-formed XML document. The XML prolog can be present but, if present, should contain only an XML Declaration (ie. it should not contain any DTD references or XML processing instructions). It should use the SOAP Envelope and SOAP Encoding namespaces and have the following form:

An XML Declaration (optional), followed by

A SOAP Envelope (the root element) which is made up of:

A SOAP Header (optional)

A SOAP Body

3.1. XML Declaration

The XML prolog contains only an XML declaration <?xml version="1.0" encoding="UTF-8" ?> specifying the XML version and the character encoding of the XML message.

Standard encoding type within IBX Connect is UTF-8. If no encoding type is stated in the XML declaration then IBX Connect will interpret messages to be UTF-8 encoded. If the encoding is e.g. "ISO-8859-1", then this needs to be declared as follows:

<?xml version="1.0" encoding="ISO-8859-1"?>

Otherwise special characters like ä,ö,ü,å etc. within the document will be encoded incorrectly during the further processing of the message.

3.2. SOAP envelope

The required SOAP Envelope element is the root element of a SOAP message. This element defines the XML document as a SOAP message.

<SOAP-ENV:Envelope

xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"

xmlns:nseps="urn:schemas-IBX:/docs/endpoint.nsendpoint"

xmlns:nsprop="urn:schemas-IBX:/docs/property.nsproperty"

xmlns:nsproc="urn:schemas-IBX:/docs/process.nsprocess"

xmlns:nsfst="urn:schemas-IBX:/docs/manifest.nsmanifest"

xmlns:nsxcbl35="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd">

3.2.1. The xmlns:soap Namespace

The first namespace in the example references the SOAP schema, which defines the elements and attributes in the SOAP message. Here it defines the Envelope as a SOAP Envelope:

xmlns:SOAP-ENV=" http://www.w3.org/2003/05/soap-envelope"

The namespace identifiers are standard and the SOAP specification requires that these namespaces either be defined correctly or not at all (ie. a SOAP message with missing namespace definitions is correct and processable but one with incorrect, ie. non-standard, definitions is incorrect and discardable).

4.1.1. The SOAP Header Element

The SOAP Header element contains application-specific information; it must be the first child element of the Envelope element.

Note: All immediate child elements of the Header element must be namespace-qualified:

<SOAP-ENV:Header>

Mandatory child elements in SOAP envelopes being exchanged with IBX Connect are:

- *nseps*
- *nsprop*

- *nsproc*
- *nsfst*

4.1.2. The mustUnderstand Attribute

The SOAP mustUnderstand attribute indicates whether a header entry is mandatory or optional for the recipient to process.

If you add mustUnderstand="1" or "true" to a child element of the Header element it indicates that the receiver processing the Header must recognize the element. If the receiver does not recognize the element it will fail when processing the Header.

4.1.3. Child element nseps: endpoints

The child element *nseps* is mandatory since it contains identifiers for the trading partners as the sender respectively receiver of the business document in the payload of the SOAP message.

<nseps:endpoints SOAP-ENV:mustUnderstand="true" xmlns:nseps="urn:schemas-IBX:/docs/endpoint.nsendpoint">

<nseps:to>

<nseps:address>receiverID</nseps:address>

</nseps:to>

<nseps:from>

<nseps:address>senderID</nseps:address>

</nseps:from>

</nseps:endpoints>

The element */Envelope/Header/endpoints/to/address* contains usually an IBX Platform unique trading partner ID (TPID) for the business partner which shall finally receive the payload while */Envelope/Header/endpoints/from/address* holds the TPID of its sender. Both have to be registered in IBX Connect to be reckognized when receiving the SOAP message and to be routed accordingly.

4.1.4. Child element nsprop:properties

The *nsprop* element contains important document properties as an unique messageID (*/Envelope/Header/properties/identity*) generated by the sender and used to check for duplicates received in IBX Connect.

<nsprop:properties SOAP-ENV:mustUnderstand="true" xmlns:nsprop="urn:schemas-IBX:/docs/property.nsproperty">

<nsprop:identity>[messageID]</nsprop:identity>

<nsprop:sentAt>[e.g.: 2008-01-01T01:01:01]</nsprop:sentAt>

<nsprop:expiresAt>[e.g.: 2099-12-31T23:59:59]</nsprop:expiresAt>

<nsprop:topic>[e.g.: Order]</nsprop:topic>

</nsprop:properties>

Mandatory are also the date of transmission to IBX Connect: */Envelope/Header/properties/sentAt* and the topic (*/Envelope/Header/properties/topic*) which provides the document type of the payload. Valid values (case sensitive!) on the IBX Connect platform are:

- Order
- ChangeOrder
- OrderResponse
- GoodsReceipt
- Invoice
- CreditNote
- Reminder
- CostObjectInformation
- PurchaseOrderStatus
- TradingPartnerUserInformation
- GLMapping
- SupplierUploadDocument
- AdvanceShipmentNotice

The type in */Envelope/Header/properties/topic* must comply with the type of business document in the payload/body of the SOAP envelope. Based on this the validation and processing of the payload is triggered.

Not mandatory is the element *nsprop:expiresAt.*

4.1.5. Child element nsproc:process

The third child element is *nsproc*:

<nsproc:process SOAP-ENV:mustUnderstand="true" xmlns:nsproc="urn:schemas-IBX:/docs/process.nsprocess">

<nsproc:correlationid>[original order's messageID]</nsproc:correlationid>

<nsproc:relatedDocid > [original order's ordernumber]</nsproc:relatedDocid>

</nsproc:process>

It is optional and may contain process dependent properties such as a *correlationid* and *relatedDocid*.

In case the payload document is of type orderresponse, changeorder, invoice etc. the *correlationid* is supposed to be the messageID of the underlying purchase order document while the *relatedDocid* is supposed to hold its ordernumber.

None of the ID's in this element are used for any business logic or mapping of documents within the IBX OnDemand platform though. The existence of the whole *<nsproc:process>* element in outbound SOAP messages from IBX Connect is due to historical reasons and depends on whether inbound documents from customers are containing this.

It´s not recommended to base any processing logic on the content of this optional tag.

4.1.6. Child element nsfst:manifest

If the SOAP envelope is embedded in a MIME structure including at least one attachment then the child element *nsfst* becomes mandatory:

<nsfst:manifest SOAP-ENV:mustUnderstand="true">

<nsfst:reference>

<nsfst:attachment nsfst:href="cid>[MIME Content-ID]"></nsfst:attachment>

<nsfst:description>urn:documents: [filename]</nsfst:description>

</nsfst:reference>

</nsfst:manifest>

/Envelope/Header/manifest/reference/attachment must then contain a reference to the MIME content part representing the attachment. It needs to always begin with "cid:" + the Content-ID within the MIME structure. */Envelope/Header/manifest/reference/description* holds its filename in the way as shown above.

Here it is important that the manifest description in the envelope header is that same as the the value of the appropriate <AttachmentLocation> tag in the payload!

There is no restriction in number of attachment references in the *nsfst* element.

4.1.1. Child SOAP Body Element

The required SOAP Body element contains the actual SOAP message intended for the ultimate endpoint of the message.

Immediate child elements of the SOAP Body element may be namespace-qualified.

Insert the business document as XML-document as text/xml as follows:

<SOAP-ENV:Body>

<Order xmlns="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd">

<OrderHeader>

<OrderNumber>

...

4. IBX MIME 1.0

4.1. General

SOAP 1.2 expects MIME 1.0.

The Encoding for binary parts of attachments is base64.

4.2. MIME-Header

The following is an example of a header at the top of the MIME message:

Mime-Version: 1.0

Content-Type: Multipart/Related; boundary="MIME_boundary"; type="text/plain";start="main"

Content-Description: [This is an optional message description.]

The Content-Type is always “Multipart/Related”. Different parts of the Content-Type must be separated by semicolons (“;”).

The only optional elements/attributes in the MIME header are

- *type*
- *start*
- *Content-Description*

If *type* and/or *start* are not present then *type* is presumed to be “text/xml” and *start* defining the main body part is presumed to be the first body part.

Content-Description is a free-text field without any processing relevance.

4.3. MIME-Body

The MIME body is separated from the header by a blank line and.

All the different parts of the MIME structure must be separated by the string “--MIME_boundary”. Field name and field value under “—MIME_boundary” are separated by a colon (“:”) and one blank space.)

4.3.1. Main document

The first MIME part is usually the main document meaning the complete SOAP envelope with header and body containing the actual payload. It is identified as such by the field “Content-ID: main“ in the MIME part’s header:

--MIME_boundary

Content-Type: text/xml

Content-ID: main

Content-Disposition: INLINE

The Content-Type has to be “text/xml”, Content-Disposition must be “INLINE”.

Field names and field values are separated by a colon (“:”) and one blank space.

After the part header, separated by a blank line, the part body begins:

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
…
</SOAP-ENV:Envelope>

4.3.2. Attachment

The mandatory elements of the following mime part headers are described below. All of them are mandatory and of data type “string”.

# Field	Description
Content-Type	MIME-Type of attachment Examples: • application/pdf • application/vnd.ms-excel • image/jpeg
Content-Transfer-Encoding	Fixed value: “base64”
Content-ID	ID of the attachment from manifest and <AttachmentLocation> tag in the payload
Content-Disposition	Fixed value: “ATTACHMENT”

4.3.3. Example:

MIME-Version: 1.0
Content-Type: Multipart/Related; boundary=”MIME_boundary”; type=”text/xml”; start=”main"
Content-Description: This is the optional message description.

--MIME_boundary
Content-Type: text/xml
Content-ID: main
Content-Disposition: INLINE

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:nseps="urn:schemas-IBX:/docs/endpoint.nsendpoint"
xmlns:nsprop="urn:schemas-IBX:/docs/property.nsproperty"
xmlns:nsproc="urn:schemas-IBX:/docs/process.nsprocess"
xmlns:nsfst="urn:schemas-IBX:/docs/manifest.nsmanifest"
xmlns:nsxcbl35="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">


```
<SOAP-ENV:Header>
      <nseps:endpoints xmlns:nseps="urn:schemas-IBX:/docs/endpoint.nsendpoint" SOAP-
ENV:mustUnderstand=”true”>
          ...
      </nseps:endpoints>
      <nsprop:properties xmlns:nsprop="urn:schemas-IBX:/docs/property.nsproperty" SOAP-
ENV:mustUnderstand=”true”>
          ...
      </nsprop:properties>
      <nsproc:process xmlns:nsproc="urn:schemas-IBX:/docs/process.nsprocess" SOAP-
ENV:mustUnderstand=”true”>
          ...
      </nsproc:process>
      <nsfst:manifest xmlns:nsfst="urn:schemas-IBX:/docs/manifest.nsmanifest" SOAP-
ENV:mustUnderstand=”true”>
          <nsfst:reference>
            <nsfst:attachment nsfst:href="cid:4452B5B1F076726BE1000000C729347F"/>
            <nsfst:description>file1.pdf</nsfst:description>
          </nsfst:reference>
          <nsfst:reference>
            <nsfst:attachment nsfst:href="cid:44ABD026565A0CCAE1000000C72934C4"/>
            <nsfst:description>files2.xls</nsfst:description>
          </nsfst:reference>
      </nsfst:manifest>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
              <SOAP-ENV:Body>
         <Order xmlns="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd">
                        ....
                  <Attachment>...<AttachmentLocation>file1.pdf</AttachmentLocation> ...
</Attachment>
                  .....
                  <Attachment> .... <AttachmentLocation>file2.xls</AttachmentLocation> ...
</Attachment>
                        ...
          </Order>
              </SOAP-ENV:Body>
            </SOAP-ENV:Envelope>

--MIME_boundary
Content-Type: application/pdf
Content-Transfer-Encoding: base64
Content-ID: 4452B5B1F076726BE1000000C729347F
Content-Disposition: ATTACHMENT

AQBTAP0ACgAsAAIAHwAQAAAAvgAQACwAAwAmAFAAVAAnAGQABwC+ABYALQAAAFAAUA
BQAFAAUABAFAAVQAHAP0ACgAuAAAAFwAKAAAAvgAUAC4AAQBQAFAAUABQAFAAUABVA
AcAvgAWAC8AAABQAFAA          ( ...base64 coded file1.pdf...)

--MIME_boundary

Content-Type: application/vnd.ms-excel

Content-Transfer-Encoding: base64
```


Content-ID: 44ABD026565A0CCAE1000000C72934C4

Content-Disposition: ATTACHMENT

AQBTAP0ACgAsAAIAHwAQAAAAvgAQACwAAwAmAFAAVAAnAGQABwC+ABYALQAAAFAAUA
BQAFAAUABAFAAVQAHAP0ACgAuAAAAFwAKAAAAvgAUAC4AAQBQAFAAUABQAFAAUABVA
AcAvgAWAC8AAABQAFAA (… base64 coded file2.xls)

--MIME_boundary

4.3.4. Attachment restrictions

The maximum allowed size of order/changeorder messages is 5 MB. This applies for the whole message in MIME structure if attachments are included as well as for simple SOAP envelopes.

In practice this means that the size of attachments should not increase much more than 4 MB since the other parts' size can vary depending on number of line items etc.

4.4. Verification

- If a MIME structure is present IBX Connect expects at least one attachment and rejects messages if no attachment is found.
- The message gets rejected if an attachments in the manifest is not present as a MIME-Part.
- The message gets rejected if a MIME-Part with an attachment is not provided in the manifest.
- The message gets rejected if its total size exceeds 5 MB.

5. Validation

5.1. General

Each document sent to IBX Connect results in a synchronous response message back to the calling application. The error messages are dependant from the process.

In case of successful transmission a HTTP 2xx is returned to the sender. The SOAP-Response can hold warnings or information.

The response is:

- HTTP-Header with a standard response code (2xx, 4xx, 5xx)
- HTTP-Body: The response is SOAP based and describes the result of the operation on a detailed level with message id of received document.
- If an error is detected (validation, duplicate check), the response message describes what went wrong.

The inbound validation is done in two steps.

- Basic validation
- Schema validation

Every process has additional asynchronous checks and validations, but those feedback channels are not described in this chapter.

Error messages in this chapter are just samples.

5.2. Response Success

An HTTP 200 OK response code is returned if the message has successfully passed the inbound validation at IBX Connect (basic- and schema validation).

5.2.1. Sample Response Success

 # HTTP HEADER:

HTTP/1.0 200 OK
Content-Type: text/xml
Set-Cookie: ssnid=; path=/;
Connection: Close
Content-Length: 1021

#HTTP-Body:

<?xml version="1.0" encoding="UTF-8"?>

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-ENC="http://www.w3.org/2003/05/soap-encoding">

<SOAP-ENV:Body>

<nssuc:success xmlns:nssuc="urn:schemas-IBX:/docs/success.nssuccess">

<nssuc:code>200</nssuc:code>

<nssuc:description>Order *messageid* accepted</nssuc:description>

</nssuc:success>

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>

5.3. Step One: Basic validation

In general a HTTP 5xx returns to sender if server problems occur. The response is driven by the used basic software and will be usually not formatted as described here.

In the basica validation the receiver tries to recognize and save the document at all.

In case of grave errors a HTTP 400 with a simple message is returned to sender. The document will not be stored at IBX side.

Reasons can be for instance:

- incorrect encoding
- incorrect syntax in the SOAP envelope
- unknown document type ((*/Envelope/Header/properties/topic*)
- total size of message exceeding 5 MB limit

5.3.1. Sample Response Grave Error

```
<?xml version="1.0" encoding="UTF-8"?>

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://www.w3.org/2003/05/soap-encoding">

  <SOAP-ENV:Body>

    <SOAP-ENV:Fault>

      <SOAP-ENV:Code>

        <SOAP-ENV:Value>SOAP-ENV:Sender</SOAP-ENV:Value>

      </SOAP-ENV:Code>

      <SOAP-ENV:Reason>

        <SOAP-ENV:Text xml:lang="en-US">error in document</SOAP-ENV:Text>

      </SOAP-ENV:Reason>

      <SOAP-ENV:Detail>

        <nsresp:listoferror xmlns:nsresp="urn:schemas-IBX:/docs/response.nsresponse">

          <nsresp:error>

            <nsresp:code>0</nsresp:code>

            <nsresp:description>##ERROR PATH: xxx  ##ERROR MESSAGE:grave error in
document</nsresp:description>

          </nsresp:error>

        </nsresp:listoferror>

      </SOAP-ENV:Detail>

    </SOAP-ENV:Fault>

  </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

5.4. Step Two: xCBL Schema Validation

Depending on the type of business document in the payload the second validation step refers to different XML schemas. If the payload violates the respective schema definition then an HTTP 400 response code is returned with a more detailed error description in the SOAP body.

5.5. Response Error

In the error response there are two parts of information

- Information about the rejected messageID and the kind of rejection
- The error list of all errors which were detected. Each error consists of a code and a description

5.5.1. Sample Response Error)

```
<?xml version="1.0" encoding="UTF-8"?>

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://www.w3.org/2003/05/soap-encoding">

 <SOAP-ENV:Body>

  <SOAP-ENV:Fault>

   <SOAP-ENV:Code>

    <SOAP-ENV:Value>SOAP-ENV:Sender</SOAP-ENV:Value>

   </SOAP-ENV:Code>

   <SOAP-ENV:Reason>

    <SOAP-ENV:Text xml:lang="en-US">Order messageid validation failed</SOAP-ENV:Text>

   </SOAP-ENV:Reason>

   <SOAP-ENV:Detail>

    <nsresp:listoferror xmlns:nsresp="urn:schemas-IBX:/docs/response.nsresponse">

     <nsresp:error>

      <nsresp:code>DT-012</nsresp:code>

      <nsresp:description>##ERROR PATH: /Order/OrderHeader/OrderIssueDate  ##ERROR
MESSAGE:[ISC.0082.9469] Value does not match pattern(s)</nsresp:description>

     </nsresp:error>

    </nsresp:listoferror>

   </SOAP-ENV:Detail>

  </SOAP-ENV:Fault>

 </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```


5.6. Response Duplicate Check

The IBX Connect receiver is also validating against already received transactions. Duplicate documents are not accepted. The duplicate check builds on a unique combination of

- Document type
- SenderID
- MessageID

and

- Document type
- SenderID
- BuyerOrderNumber (in case of purchase orders) / InvoiceNumber (in case of invoices)

If a document is not unique referring to these criterias then the default response code is HTTP 400.

In case of orders and invoices the duplicate check based on ordern- resp. invoicenumber will only apply if a previously received document with the same attributes has been processed successfully.

5.6.1. Configurable Duplicate Response

To avoid automatic resends of duplicate messages due to a specific sender side interpretation of a received HTTP 400 response codes it´s possible to configure a customer specific response code and message at IBX Connect.

To configure a non-standard http code, the sender's trading partner profile in IBX Connect needs to be updated.

Options:

DuplicateCode - Numeric HTTP response code that should be retured in case of dupl. failure

DuplicateMessage – Customer specific standard message to be returned in case of dupl. failure

Sample

The sample below is a SOAP response sent to a partner that has specified "999" as the duplicate failure code and "999 - Customized error message" as error message.

HTTP HEADERS:

HTTP/1.0 999 OK

Content-Type: text/xml

Set-Cookie: ssnid=; path=/;

Connection: Close

Content-Length: 1021

SOAP FAULT:

```
<SOAP-ENV:Envelope... >
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <SOAP-ENV:Code>
        <SOAP-ENV:Value>SOAP-ENV:Sender</SOAP-ENV:Value>
      </SOAP-ENV:Code>
      <SOAP-ENV:Reason>
        <SOAP-ENV:Text xml:lang="en-US">999 - Customized error message</SOAP-ENV:Text>
      </SOAP-ENV:Reason>
      <SOAP-ENV:Detail>
        <nsresp:listoferror xmlns:nsresp="urn:schemas-IBX:/docs/response.nsresponse">
          <nsresp:error>
            <nsresp:code>IBX301</nsresp:code>
            <nsresp:description>##ERROR PATH: N/a  ##ERROR MESSAGE:The message ID for the
document was not unique. Document ID: dupl_001. Document type:
SOAP_Invoice_xcbl35</nsresp:description>
          </nsresp:error>
        </nsresp:listoferror>
      </SOAP-ENV:Detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6. IBX SOAP 1.2 Envelope Specification in Table structure

The following tables give an overview of the already described elements of the IBX SOAP 1.2 envelope with or without MIME structure.

Legend - IBX platform requirements:

Legend symbol	Meaning
M	Field is mandatory.
C	Field is conditionally mandatory (depends on presence of attachments)
O	Field is optional.

Element	IBX Platform Requirements
MIME-Version: 1.0	C
Content-Type: Multipart/Related; boundary=”MIME_boundary”; type=”text/xml”; start=”main"	C
Content-Description: This is the optional message description.	O
--MIME_boundary	
Content-Type: text/xml	C
Content-ID: main	O
Content-Disposition: INLINE	C
<?xml version="1.0" encoding="utf-8"?>	M
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"	M
xmlns:nseps="urn:schemas-IBX:/docs/endpoint.nsendpoint"	O
xmlns:nsprop="urn:schemas-IBX:/docs/property.nsproperty"	O
xmlns:nsproc="urn:schemas-IBX:/docs/process.nsprocess"	O
xmlns:nsfst="urn:schemas-IBX:/docs/manifest.nsmanifest"	O
xmlns:nsxcbl35="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd"	O
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">	O
<SOAP-ENV:Header>	M
<nseps:endpoints SOAP-ENV:mustUnderstand="1" xmlns:nseps="urn:schemas-IBX:/docs/endpoint.nsendpoint">	M
<nseps:to>	M
<nseps:address>	M

<nseps:from>	M
<nseps:address>	M
<nsprop:properties SOAP-ENV:mustUnderstand="true" xmlns:nsprop="urn:schemas-IBX:/docs/property.nsproperty">	M
<nsprop:identity>messageID</nsprop:identity>	M
<nsprop:sentAt>2008-01-01T01:01:01</nsprop:sentAt>	M
<nsprop:expiresAt>2099-12-31T23:59:59</nsprop:expiresAt>	O
<nsprop:topic>	M
<nsproc:process SOAP-ENV:mustUnderstand="true" xmlns:nsproc="urn:schemas-IBX:/docs/process.nsprocess">	O
<nsproc:correlationid>	O
<nsproc:relatedDocid>	O
<nsfst:manifest SOAP-ENV:mustUnderstand="false">	M
<nsfst:reference>	M
<nsfst:attachment nsfst:href=	M
<nsfst:description>urn:documents:	M
<SOAP-ENV:Body>	M
<Order xmlns="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd">	M
<OrderHeader> …	M
--MIME_boundary	C
Content-Type: application/pdf	C
Content-Transfer-Encoding: base64	C
Content-ID: 4452B5B1F076726BE1000000C729347F	C
Content-Disposition: ATTACHMENT	C
AQBTAP0ACgAsAAIAHwAQAAAAvgAQACwAAwAmAFAAVAAnAGQ (…base64 encoded attachment...)	C

7. SOAP 1.2 Schema definitions (XSD)

Find embedded useful schema definition files for validation of the SOAP 1.2 envelope structure and its elements: