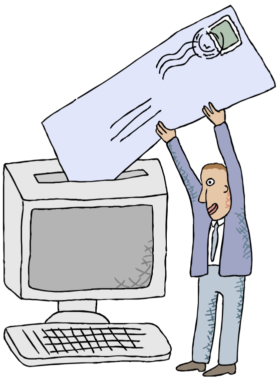
SFTI Transport profile: Bas



Source text	Translation data
Version: 2.0	Translation version: 2.0 – 0
Status: Standard	Source language: Swedish
Date: 2005-11-30	Translation date: 2006-01-07

Authors:

Anders W. Tell	Financial Toolsmiths AB
Martin Forsberg	Amnis Consulting
Sören Lennartsson	Ooi Data

Revision history

Note. Changes and amendments are documented only with respect to adopted versions of this transport profile, i.e. versions of profile documentation indicated as ”Standard”.

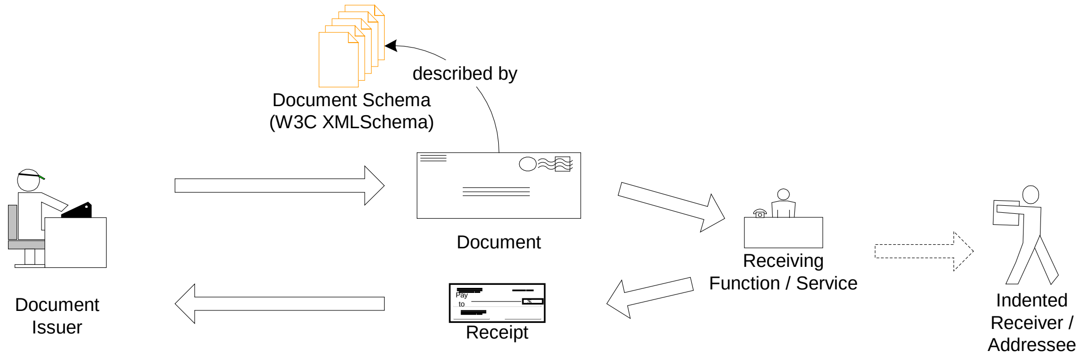
Source text version	**Source text date**	**Description of amendments made**	**Made by**
1.0.0	2004-06-16	Original document published	The authors
1.1	2004-11-08	Corrections of identified errors, and some ameliorations of the text. History of amendments added. Version indicated by two figures (x.y) system	S Lennartsson
2.0	2005-10-26	Revised principles for acknowledgment of receipt. Example corrected. In this version acknowledgment of receipt according to BPSS ha been substituted for a similar receipt according to ebMS 2.0. Several documents may now be transferred in one message. This version is not backward compatible with previous versions 1.0 and 1.1.	M Forsberg

Table of Contents

1. Introduction

1.1. *Executive summary*

SFTI Transport Profile Bas is a specification for simple electronic communication between business partners and/or intermediaries acting on their behalf. The profile “Bas” is designed for simplified communication of a message, that contains one or more documents, and one message in return as acknowledgment of receipt.
(Note. In the source text the word “Bas” carries the meaning of “basic”.)



The profile is based on the ebXML framework and its ebXML Message Service Specification [ebMS] Version 2.0. The ebXML Framework has been developed by UN/CEFACT in collaboration with OASIS and their constituent parts are ISO standards. UN/CEFACT is also responsible for UN/EDIFACT, and ebXML can be viewed as a successor to UN/EDIFACT in terms of technology.

In the profile “Bas” the documents are transferred using the HTTP/S protocol [SSL3].

<table>
<tr><td colspan="3">ebXML Message Service Handler</td></tr>
<tr><td colspan="3">SOAP with Attachment</td></tr>
<tr><td>HTTP/S</td><td>MIME</td><td>XML</td></tr>
<tr><td colspan="3">TCP/IP</td></tr>
<tr><td colspan="3">...</td></tr>
</table>

”Collaboration Protocol Profile and Agreement” [ebCPPA] and related other ebXML specifications are not part of the ”Bas” profile.

This specification of profile ”Bas” shall be understood as a strict subset of the ebXML MSH specification [ebMS]. This document is constructed as an amendment to the complete [ebMS] specification, and the documents should be read together

The ”Bas” profile is derived from the ebXML Framework with several of its options excluded. It does not require a full [ebMS] message handling service implementation, but may be implemented with limited resources based on commonly available knowledge and technologies. Open source implementations are available. For more information, see the annex on realisation technologies.

As the ”Bas” profile is a strict sub-set of the ebXML Framework, parties who wish to utilize additional functionality in their business communications can easily upgrade *within* the ebXML Framework, for example by adding reliability and security features.

It should be noted that this “Bas” profile can be used with XML formatted documents as well as with UN/EDIFACT formatted documents.

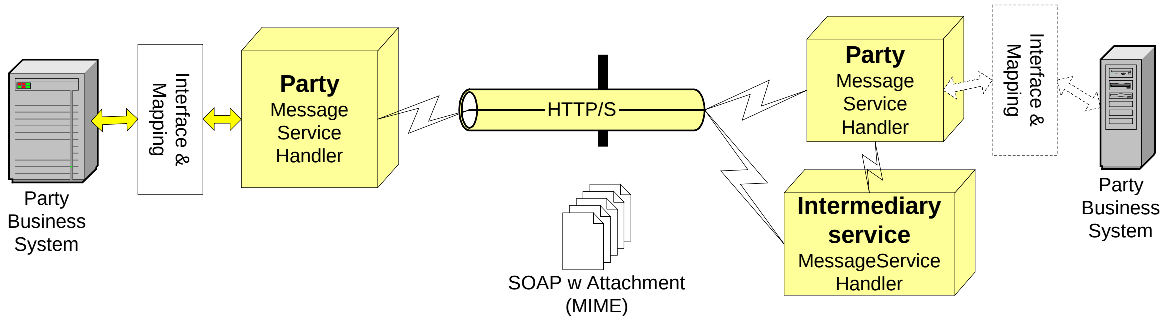
If a receiving Message Service Handler – MSH – identifies a deviation from this profile, e.g. an [ebMS] option that has been excluded in this profile, the receiving MSH may choose, depending on built-in functionality, whether to accept the incoming message or to report an error due to the fact that the sending MSH has exceeded the constraints of this profile. See the section on error handling for further information on how to deal with this kind of incompatibility issues.

2. Transport protocol

Messages, each containing *a document (or documents)* or a *receipt*, are transported between parties by means of a limited application of the HTTP/S protocol, i.e. [SSL3].

A message from a document issuer is sent from that party's Message Service Handler (MSH) or from an intermediary service MSH acting on behalf of that document issuer.

A message to an intended document receiver is sent to the addressee's Message Service Handler (MSH) or to an intermediary service MSH acting on behalf of that intended message receiver.



The communication between document issuer/sender and document addressee/receiver is conducted synchronously through their message handling services. This means that a [SSL3], i.e. HTTP/S, connection shall not be closed, or disconnected, before a response has been received or a timeout has occurred. An acknowledgment of receipt must be sent on the same connection as the document itself. Irrespective if the MSH is being operated under the party's own management or by an intermediary service, the communication between a party's business system and its Message Service Handler is under the sole responsibility of that party and, consequently, may be implemented according to the party's own preferences.

In order to define that synchronous communication shall be used, <`syncReplyMode`> must be defined in the SOAP header for each message. See [ebMS] B.2.5 for more information.

Messages must be formatted according to the MIME principles defined in section 19.4 in [RFC2616]. It is recommended that implementations interpret presence of the mandatory HTTP header value 'SOAPAction' liberally, according to the guidelines in the SOAP 1.1 specification, i.e. a message not containing 'SOAPAction' should not be considered incompatible with SOAP 1.1.

If a message is transmitted successfully to a receiving MSH, a HTTP response 2XX shall be returned. If an error occurs, HTTP status codes 3XX, 4XX or 5XX shall be returned. If a SOAP error occurs a '`500 "Internal Server Error"`' according to the SOAP 1.1 specification shall be reported back to originator.
See the error handling section for more information on error handling and reporting.

See [ebMS] appendix B for normative information and details regarding the transport protocol.

2.1. Security

The [ebMS] security profile corresponding to the "Bas" profile is Security profile no. 5. Only server certificates shall be used for [SSL3] communication.

The access control mechanism "Basic and Digest Access Authentication" is not part of the "Bas" profile. See [ebMS] section B.2.6 for more information.

Note. Server certificates are used in this transport profile for encryption, not for authentication. For this purpose either own or third party certificates may be used, but be aware of the fact that own server certificates generates warnings/alerts by the client software which may confuse an inexperienced user.

3. Messages

The Bas profile is designed for sending one or more document in a message. An acknowledgment of receipt is returned when the message (incl the document(s) it contains) has been received.

3.1. Message envelope and document packaging

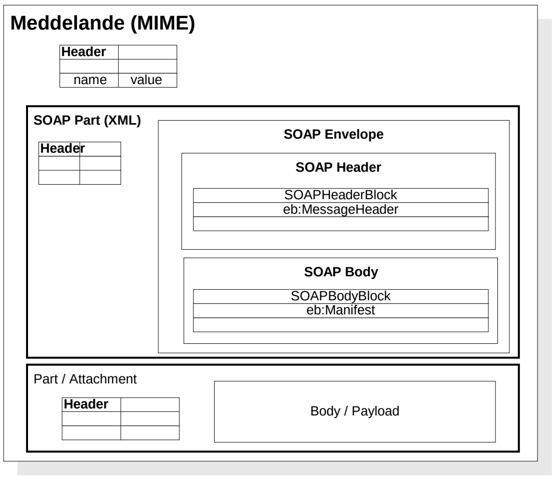
The documents are to be packaged according to MIME principles. A message consists of two or more MIME Parts. The format of a document should be defined in a format specification, a so called Document Schema.

The parts are

1. a SOAP v1.1 Envelope, in XML format
2. Payload, i.e. a document in any well defined format, for example XML, UN/EDIFACT.

See section 3.1 in [ebMS] and [SOAP] for detailed specifications, and below for an example.

The rules for message packaging and for the construction of SOAP Envelopes are given below.



3.1.1. Identification of sender and recipient of messages

The message sender and the intended recipient are identified in the SOAP header by character strings uniquely identifying them.

In the “Bas” profile one may use at least one party identifier for message sender and intended message recipient. If more than one identifier is allocated to a party then all identifiers must identify the same party.

The detailed rules for party identification are given <u>in the Collaboration Process specification</u> (see separate document).

```
<eb:From>
    <eb:PartyId eb:type=’countrycode:organizationid’> SE1234567890 </eb:PartyId>
</eb:From>
<eb:To>
    <eb:PartyId eb:type=’countrycode:organizationid’> SE9876543210 </eb:PartyId>
    <eb:PartyId eb:type=’operatorid’> QWERTYUIOP </eb:PartyId>
</eb:To>
```

At least one <`From`> and one <`To`> element shall be present in each message and its SOAP Header.

Role descriptions for message sender and receiver are <u>not</u> part of the ”Bas” profile and shall <u>not</u> be included, neither in the <`From`> nor in the <`To`> element.

The option to specify more than one party identifier may be used by third party service providers.

See [ebMS], section 3.1.1 for more information.

3.1.2. Identification of technical agreement

The ”Bas” profile does not require that the parties establish a technical agreement in writing, instead an agreement on electronic the exchange of information and documents may be formed over the phone or at the time of goods collection/delivery, or concluded when one party acts according to an offer by the other party to accept documents electronically. With regard to the communication rules, this means that the mandatory <CPAId> must not be semantically relevant in terms of the interpretation of transmitted documents. If a CPA is used, its parameters must not contradict any rules of this profile. A <CPAId> element should be constructed according to the following definition:

```
CPAId ::=  <date> ’:’ <fromPartyId> ’:’ <toPartyId>
```

```
<eb:CPAId>20040510:SE1234567890:SE9876543210</eb:CPAId>
```

A <CPAId> element shall be present in each message and its SOAP Header.

See [ebMS], section 3.1.2 for more information.

3.1.3. Conversation identification

In order to relate the messages of a collaboration process to each other in so called conversations, the [ebMS] requires a unique conversation identifier to be present in each message. The sender of the first message in a conversation defines, according to that party’s own principles, the ID for the conversation. It is recommended that the following rule is applied:

> Concatenate the date, when the first message of a conversation is sent, to a unique number. The number must be unique per date, for example a serial number starting from zero each day at 00.00.00.

```
ConversationId ::=  <dateOfInitialMessageTransmission>’:’
                    <uniqueNumber>’:’ <fromPartyId>
```

```
<eb:ConversationId>20040510:4567:SE1234567890</eb:ConversationId>
```

A <ConversationID> element shall be present in each message and its SOAP Header.

See [ebMS], section 3.1.3 for more information.

3.1.4. Identification of the message receiving service

The name of a receiving service is defined in the Collaboration Process specification (see separate document). The service and action names used in conjunction with this profile should be based on the pattern below, which is dependent on the collaboration process name and the name of the document(s) transferred.

```
Service ::= ’urn:’ <organization> ’:services:documentprocessing’
<samverkansProcess>
Action ::= ’incoming’ <documentName>
```

```
<eb:Service>urn:sfti:services:documentprocessing:BasicInvoice</eb:Service>
<eb:Action>incomingBasicInvoice</eb:Action>
```

See [ebMS], section 3.1.4 and 3.1.5 for more information.

3.1.5. Unique identification of a message

In ebMS it is possible to attribute a message with a globally unique identifier. The identifier must comply with the principles described in [RFC2822] and the timestamp must be expressed as UTC.

```
<eb:MessageData>
  <eb:MessageId>20040510-102030-28572@company.se</eb:MessageId>
  <eb:Timestamp>2004-05-15T11:12:12</eb:Timestamp>
</eb:MessageData>
```

A <MessageData> element shall be present in each message and its SOAP Header. When sending messages according to the ”Bas” profile, the following elements are excluded: <RefToMessageId> and <TimeToLive>, which describes the UTC time by which a message should be received by the intended addressee. <RefToMessageId> must however be present in Error messages.

See [ebMS], section 3.1.6 for more information.

3.1.6. Message manifest

Each ebXML message may contain a description of the payload of the message (i.e. the document or documents that it contains). In this ”Bas” profile, all messages containing documents (i.e. excluding the receipts) shall have the <Manifest> element in its SOAP Body. The ’id’ attribute shall not be specified. The ’version’ attribute shall be set to constant “2.0” (Note: this refers to ebMS SOAP header extension, version 2.0).

The <Manifest> element shall contain one <Reference> element pointing to the document (or, elements pointing to each document, in case of more than one) that has been packaged according to MIME principles. The <Reference> element is a simple link [XLINK] where the attribute ’type’ is a constant: ’xlink:type="simple"’. The attributes ’id’ and ’role’ are not part of the ”Bas” profile and shall not be specified.

The attribute ’href’ references the payload document, inside a MIME part and the reference shall be a URI conforming to the principles of [XLINK]. ’href’ shall follow the ’cid:’ naming conventions, which means that the MIME part containing the document shall have a MIME header ’content-id’ corresponding to the value of the ’href’ attribute.

```
<SOAP:Body>
  <eb:Manifest eb:version="2.0">
    <eb:Reference xlink:href="cid:ebxmlpayload1@sender.com" xlink:type="simple">
       <eb:Description xml:lang=”en-GB”> Textual description of message content </eb:Description>
       <eb:Schema eb:location=’urn:se:sfti:collaborationprocesses:BasicInvoice.xsd’
eb:version=’1.0’> </eb:Schema>
    </eb:Reference>
  </eb:Manifest>
</SOAP:Body>
```

If ’href’ does not point to an existing ’content-id’, the receiving MSH shall report this through a response message with an <Error> element containing “errorCode=MimeProblem” and “severity=Error”.

See [ebMS], section 3.2 for more information.

Schema:
Documents that are exchanged in an electronic commerce context are usually described by a document schema. For example, documents in XML-format are commonly described by a W3C XML Schema document or an ISO Relax NG document. A document sent by means of the ”Bas” profile should be

described by a schema document. The reference to this format definition document is specified in the child <`Schema`> element of the <`Reference`> element. Two attributes shall be specified: ’`location`’ and ’`version`’:

- **location** – contains a URI referring to the schema document.
- **version** – refers to the version of the schema document.

```
<eb:Schema eb:location=’urn:se:sfti:collaborationprocesses:BasicInvoice.xsd’
eb:version=’1.0’></eb:Schema>
```

These two values can be obtained from the documentation of the Collaboration Process defining the document exchange in question.

See [ebMS], section 3.2.1.1 for more information.

It should be noted that this ”Bas” profile can also be used for the exchange UN/EDIFACT- formatted documents. The <`Schema`> element should then refer to a UN/EDIFACT message format description.

Description
It is possible to provide a human readable description of a message by including a <`Description`> element inside the <`Reference`> element. This element is provided for in the “Bas” profile, but its use is not mandatory and it shall <u>not</u> influence the interpretation and handling of messages sent and received.

```
<eb:Description xml:lang=”en-GB”> Free text description of the message </eb:Description>
```

Attribute:

- ’xml:lang’ is mandatory and must be specified in each <`Description`> element.

See [ebMS], section 3.1.8 for more information.

3.1.7. Synchronisation of message and response over same connection

In order to signal to the receiver that the receipt shall be returned over the same connection a <`SyncReply`> element shall be present in each message and its SOAP Header.

```
<eb:SyncReply eb:id=’’ eb:version=’2.0’ SOAP:mustUnderstand=’1’
     SOAP:actor="http://schemas.xmlsoap.org/soap/actor/next">
</eb:SyncReply>
```

Attributes:

- ’`id`’ is optional; when used it is to uniquely identify the element
- ’`version`’ is mandatory and shall be set to ’`2.0`’
- ’`SOAP:mustUnderstand`’ is mandatory and shall be set to ’`1`’
- ’`SOAP:actor`’ is mandatory and shall be set to ’`http://schemas.xmlsoap.org/soap/actor/next`’

See [ebMS], section 4.3 for more information.

3.1.8. Request for acknowledgment of receipt

The sender of a message requests the receiver to return an acknowledgment of receipt by including, in the initial message, an `<AckRequested>` element in its SOAP Header. AckRequested is mandatory in this transport profile, and shall occur only once in per message. For responses by the receiving MSH, see section 3.2.

Signed acknowledgments of receipt shall not be requested in this transport profile.

```
<eb:AckRequested SOAP:mustUnderstand="1" eb:version="2.0" eb:signed="false"
     SOAP:actor="urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH">
</eb:AckRequested>
```

Attributes:

- ’`id`’ is optional; when used it is to uniquely identify the element
- ’`version`’ is mandatory and shall be set to ’`2.0`’
- ’`SOAP:mustUnderstand`’ is mandatory and shall be set to ’`1`’
- ’`signed` ’ is mandatory and shall be set to ’false’
- ’`SOAP:actor`’ is mandatory and shall be set to "`urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH`”

See [ebMS] section 6.3.1 for more information.

3.2. Response messages

A response message, containing either an acknowledgment of receipt or an error message, shall be returned in the same connection as the initial document.

If a message has been successfully transferred to a receiving MSH, a HTTP/S response 2XX shall be returned in a response message that contains an ebMS Header and an acknowledgment of receipt, see section 3.2.1. Otherwise, an error message is sent according to 3.2.2.

3.2.1. Acknowledgment of Receipt from intended recipient

An acknowledgment of receipt is represented by an `<Acknowledgment>` element in the SOAP header.

The initial message for which acknowledgment of receipt is issued shall satisfy the following:

1. All information in the message shall be available to the intended addressee.
2. The message shall be formatted according to [ebMS] message and packaging principles.

The verification and sending of an acknowledgment of receipt may occur in the intended receiver’s MSH or in an third party MSH acting on behalf of the intended addressee.

A document is considered to be **at the recipient’s disposal** once an acknowledgment of receipt has been **dispatched**.

An acknowledgment of receipt document shall satisfy the following criteria:

1. All information in the initial message shall be available to the intended addressee.
2. The response message shall be formatted according to [ebMS] packaging and message principles.
3. The acknowledgment of receipt shall have a format that, according to an agreement, the intended recipient can handle.

Rules for acknowledgment of receipt document:
The acknowledgment of receipt consists of only a SOAP v1.1 Envelope in XML format. The rules in 3.1.1 – 3.1.5 applies also to the receipt, with the following amendments.

For acknowledgment of receipt

- The <Service> element shall be set to" urn:oasis:names:tc:ebxml-msg:service", and
- The <Action> element shall be set to "Acknowledgment".

```
    <eb:Service>urn:oasis:names:tc:ebxml-msg:service</eb:Service>
    <eb:Action>Acknowledgment</eb:Action>
```

The `<ConversationId>` element shall contain the same value as `<ConversationId>` in the message being acknowledged.

The `<Acknowledgment>` element shall be included in the SOAP header of the acknowledgment of receipt message.

```
<eb:Acknowledgment  SOAP:mustUnderstand="1" eb:version="2.0"
          SOAP:actor="urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH">
  <eb:Timestamp>2005-10-28T17:29:01Z</eb:Timestamp>
  <eb:RefToMessageId>20040510-102030-28572@myCompany.se</eb:RefToMessageId>
  <eb:From>
    <eb:PartyId eb:type="countrycode:organizationid">SE9876543210</eb:PartyId>
  </eb:From>
</eb:Acknowledgment>
```

Attributes:

- '`id`' is optional; when used it is to uniquely identify the element
- '`version`' is mandatory and shall be set to '`2.0`'
- '`SOAP:mustUnderstand`' is mandatory and shall be set to '`1`'
- '`SOAP:actor`' is mandatory and shall be set to `"urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH"`

The mandatory `<Timestamp>` element shall contain the time, as seen by the recipient, when the initial message was received by the recipients. This timestamp is taken to be the time of receipt also for the document (or documents) contained in the message.

The mandatory `<RefToMessageId>` element shall contain the same message identifier as the initial message being acknowledged.

The `<From>` element shall contain the identity of the party creating the acknowledgment of receipt. If the element is missing, the `<From>` element in `<MessageHeader>` applies.

In this transport profile signed acknowledgments of receipt are not included, and the `<Reference>` element shall not be used.

See [ebMS] section 6.3.2 for more information.

3.2.2. Error messages and error reporting

Error handling and reporting can be divided into four primary areas:

1. Communication protocol errors (HTTP/S)
2. Packaging problems (MIME)
3. SOAP problems
4. Message handling problems [ebMS].

See section 4.2 [ebMS] for details.

3.2.2.1 HTTP/S errors

If an error occurs, HTTP status 3XX, 4XX, or 5XX shall be returned.

3.2.2.2 MIME errors

These types of errors are reported by returning a SOAP message with an <ErrorList> element.

See [ebMS], section 2.1.6 for more information.

3.2.2.3 SOAP Errors

If SOAP errors occur '500 "Internal Server Error"' shall be returned in accordance with the SOAP 1.1 specification. A sending MSH shall be prepared to accept and process SOAP Fault values.

3.2.2.4 Message handling errors

Errors in message handling are reported by returning an <ErrorList> element in the SOAP header of the response message. <ErrorList> shall not be included in an initial message but only in response messages reporting the error(s).

See [ebMS], section 4.2 for more information.

3.2.2.5 Reporting of profile incompatibility

Reporting of incompatibility with this profile shall be made by the receiving MSH by returning an <ErrorList> element with the error code ”NotSupported”.

3.3. Optional [ebMS] features that shall not be used in messages

The ebXML framework provides a number of optional features. This section describes features that shall not to be used by message handling services applying this “Bas” profile.

3.3.1. <DuplicateElimination>

By inclusion of a <DuplicateElimination> element in the SOAP header, the sender may request that the receiver shall check if an incoming message is a copy of an message received earlier and, if so, shall not forward such an message to the business system. If the <DuplicateElimination> element is not present then a message is to be delivered in a Best-Effort behaviour.
This optional SOAP Header element shall not be specified in the ”Bas” profile.

See [ebMS], section 3.1.7 for more information.

3.3.2. <Signature>

A message may be signed digitally by the inclusion of a <Signature> element in the SOAP header.
This optional SOAP Header element shall not be specified in the ”Bas” profile.

See [ebMS], section 4.1.1 for more information.

3.3.3. <MessageOrder>

The sender may request that the receiving MSH shall forward messages to the business system of the <To> party with preservation of the message order.
This optional SOAP Header element shall not be specified in the ”Bas” profile.

See [ebMS], section 9 for more information.

3.4. Other Message Handler Services

MSH implementations of this profile must not require of other MSH software to support the Status or Ping-Pong services.

See [ebMS], section 7 and 8 for more information.

4. A message example

The following example consists of a message with HTTP header followed by a SOAP Envelope and a document in two MIME parts.

```
POST /servlet/ebXMLhandler HTTP/1.1
Host: www.receiver.se
SOAPAction: "ebXML"
Content-type: multipart/related; boundary="BoundarY"; type="text/xml";
start="<ebxhmheader1@sender.com>"

--BoundarY
Content-ID: <ebxhmheader1@sender.com>
Content-Type: text/xml

<?xml version="1.0" encoding="UTF-8"?>
<SOAP:Envelope xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:eb="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd"
    xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://www.oasis-open.org/committees/ebxml-msg/schema/envelope.xsd
    http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd
    http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd">
  <SOAP:Header>
    <eb:MessageHeader SOAP:mustUnderstand="1" eb:version="2.0">
      <eb:From><eb:PartyId eb:type="countrycode:organizationid">SE1234567890</eb:PartyId></eb:From>
      <eb:To>  <eb:PartyId eb:type="countrycode:organizationid">SE9876543210</eb:PartyId>  </eb:To>
      <eb:CPAId>20040510:SE1234567890:SE9876543210</eb:CPAId>
      <eb:ConversationId>20040510:4567:SE1234567890</eb:ConversationId>
      <eb:Service>urn:sfti:services:documentprocessing:BasicInvoice</eb:Service>
      <eb:Action>incomingBasicInvoice</eb:Action>
      <eb:MessageData>
        <eb:MessageId>20051108-102030-28572@foretag.se</eb:MessageId>
        <eb:Timestamp>2005-11-08T11:12:12</eb:Timestamp>
      </eb:MessageData>
    </eb:MessageHeader>
    <eb:AckRequested SOAP:mustUnderstand="1" eb:version="2.0" eb:signed="false"
SOAP:actor="urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH"/>
    <eb:SyncReply eb:id="" eb:version="2.0" SOAP:mustUnderstand="1"
          SOAP:actor="http://schemas.xmlsoap.org/soap/actor/next">
    </eb:SyncReply>
  </SOAP:Header>

  <SOAP:Body>
    <eb:Manifest eb:version="2.0">
      <eb:Reference xlink:href="cid:ebxmlpayload1@sender.com" xlink:type="simple">
         <eb:Description xml:lang="se">Free form description of message content</eb:Description>
         <eb:Schema eb:location="urn:se:sfti:collaborationprocesses:BasicInvoice.xsd"
eb:version="1.0"> </eb:Schema>
      </eb:Reference>
    </eb:Manifest>
  </SOAP:Body>

</SOAP:Envelope>

--BoundarY
Content-ID: <ebxmlpayload1@sender.se>
Content-Type: text/xml

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

```
<Invoice xmlns:sfti=”urn:se:sfti”>
</Invoice>

--BoundarY––
```

4.1. Acknowledgment of receipt

```
HTTP/1.1 200 OK
SOAPAction: "ebXML"
Content-Type: text/xml

<?xml version="1.0" encoding="UTF-8"?>
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/ http://www.oasis-
open.org/committees/ebxml-msg/schema/envelope.xsd" xmlns:eb="http://www.oasis-
open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd">
            <SOAP:Header xsi:schemaLocation="http://www.oasis-open.org/committees/ebxml-
msg/schema/msg-header-2_0.xsd http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-
2_0.xsd">

  <eb:MessageHeader SOAP:mustUnderstand="1" eb:version="2.0">
    <eb:From>
      <eb:PartyId eb:type="countrycode:organizationid">SE9876543210</eb:PartyId>
    </eb:From>
    <eb:To>
      <eb:PartyId eb:type="countrycode:organizationid">SE1234567890</eb:PartyId>
    </eb:To>
    <eb:CPAId>20040514:SE1234567890:SE9876543210</eb:CPAId>
    <eb:ConversationId>20040514:4567:SE1234567890</eb:ConversationId>
    <eb:Service>urn:oasis:names:tc:ebxml-msg:service</eb:Service>
    <eb:Action>Acknowledgment</eb:Action>
    <eb:MessageData>
      <eb:MessageId>20051108-102030-28572@abc.se</eb:MessageId>
      <eb:Timestamp>2005-11-08T11:12:16</eb:Timestamp>
      <eb:RefToMessageId>20051108-102030-28572@organization.se</eb:RefToMessageId>
    </eb:MessageData>
  </eb:MessageHeader>
  <eb:Acknowledgment SOAP:mustUnderstand="1" eb:version="2.0"
SOAP:actor="urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH">
    <eb:Timestamp>2005-11-08T11:12:16</eb:Timestamp>
    <eb:RefToMessageId>20051108-102030-28572@organization.se</eb:RefToMessageId>
      <eb:From>
        <eb:PartyId eb:type="countrycode:organizationid">SE9876543210</eb:PartyId>
     </eb:From>
  </eb:Acknowledgment>
  </SOAP:Header>

  <SOAP:Body xsi:schemaLocation="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-
header-2_0.xsd http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd"/>
</SOAP:Envelope>
```

5. Transport profile Bas and CPA

The “Bas” profile does not presuppose that the parties use an ebXML Collaboration Protocol Agreement [ebCPPA] document. However, in certain situations it can be practical for either party – or both of them – to register its details regarding communication in a technical profile. The data in the profile can then be used for automatic processing as well as for documentation. The “Bas” profile provides an option for such a document through the profile description below. The publication of and sharing such a document is only an indication that the organization is capable of handling the process, document, etcetera, as defined, and shall not be legally binding, unless otherwise agreed.

The following information may be registered in the transport profile

- Identifier [0..1] – Issuing party’s identification of the transport profile
- Start date [0..1] – date upon which the profile information takes effect (begins at 00.00.00)
- End date [0..1] – date upon which the profile information (expires at end of the date)
- Information relating to the party [1..1] – the party for which the process and technical information of this technical profile applies
  - Identifier [1..1] of the party publishing the profile (used in elements <`From`> or <`To`>, depending on the roll of the party in the collaboration process)
  - Additional Identifiers [0..*] of the party publishing the profile (if more identifiers are needed in the <`From`> or <`To`> elements)
  - Contact details [0..*]
    - Contact function [0..1] – the function of the contact at the profile issuing party
    - Name [1..1] – name of contact point, contact person or department handling any queries regarding the sending and receiving of documents relating to this profile
    - Communication details [0..*]
      - URI [0..1]
      - Means of contact [0..1] – e.g. telephone, fax, e-mail, web
      - Contact number/address string [0..*]
- Collaboration Process specification [0..*]
  - Version [0..1] – version of collaboration process specification: VersionID ::= <major> ’.’ <minor> ’.’ <revision>
  - Status [0..1] – status of collaboration process specification: [draft | standard]
  - Name [1..1] – identification of the collaboration process specification
  - Role [1..1] – role in the collaboration process specification [sender | receiver]
  - Communication address of receiving MSH [1..1] (if recipient role)
  - Document schema [1..1] reference to document schema for the collaboration process
  - Service [1..1] – name of receiving service (if recipient role)
  - Action [1..1] – name of specific activity within receiving service (if recipient role)

See the section 7.5, Core Components definitions, for more information.

6. Realisation technologies

6.1. Java – JAXM, SAAJ, Java Server Pages (JSP)

The Java framework contains built-in support for ebXML communication through the ”Java Web Services Developer Pack”, see < http://java.sun.com/webservices/webservicespack.html> for more information.

Information on the use of JAXM Servlets as a simple solution for sending and receiving messages according to this profile: [JAXM] <http://java.sun.com/xml/jaxm/>
API and specifications: < http://java.sun.com/xml/downloads/jaxm.html>
See also ”SOAP with Attachments API for Java (SAAJ)”, <http://java.sun.com/xml/saaj/index.jsp>

6.2. Code example in Java

The following code example shows a simple arrangement for sending a document using the Open source software Hermes.

```
package DemoEbxml;
import hk.hku.cecid.phoenix.message.handler.*;
import hk.hku.cecid.phoenix.message.packaging.*;
import java.net.URL;
import java.util.Date;
import javax.activation.DataHandler;

public class SftiSendEbxmlMessage implements MessageListener{

    public void SendEbXMLMessage(){
      try{
      ApplicationContext ac;
      ac = new ApplicationContext("20040514:SE1234567890:SE9876543210",
                 "20040514:4567:SE1234567890",
                 "urn:sfti:services:documentprocessing:BasicInvoice",
                 "incommingBasicInvoice");

      // Make a request
      Request mshReq = new Request(ac, new URL("http://localhost:8080/msh/"),  this, "HTTP");

      // Create a message
      EbxmlMessage ebxmlMessage = new EbxmlMessage();
      MessageHeader msgHeader = ebxmlMessage.addMessageHeader();
      msgHeader.addFromPartyId("SE1234567890", "countrycode:organizationid");
      msgHeader.addToPartyId("SE9876543210", "countrycode:organizationid");
      msgHeader.setCpaId("20040514:SE1234567890:SE9876543210");
      msgHeader.setConversationId("20040514:4567:SE1234567890");
      msgHeader.setService("urn:sfti:services:documentprocessing:BasicInvoice");
      msgHeader.setAction("incommingBasicInvoice");
      String messageId = Utility.generateMessageId(new Date(),ebxmlMessage);
      msgHeader.setMessageId(messageId);
      msgHeader.setTimestamp(Utility.toUTCString(new Date()));

      // Link a document to the message:
      AttachmentDataSource ads = new AttachmentDataSource("sfti.xml", "text/xml");
      DataHandler dataHandler = new DataHandler(ads);
      ebxmlMessage.addPayloadContainer(dataHandler, "contentId",
                                      "Free text message description");
      ebxmlMessage.addSyncReply();

      // Save:
      ebxmlMessage.saveChanges();

      // Send message.
      // Collect the response by calling onMessage()
      mshReq.send(ebxmlMessage);

      // Wait for response through onMessage()
      Thread.sleep(30000);
      }

    // Error handling
     catch(Exception e){
       System.err.println(e);
     }
   }

   public URL getClientUrl(){ return null;  }

   // Response to the so called Callback
   public void onMessage(EbxmlMessage ebxmlMessage){
      System.err.println("<<Message recived>>");
   }

   /**/
   public static void main(String[] args){
     SftiSendEbxmlMessage sendMessage = new SftiSendEbxmlMessage();
     sendMessage.SendEbXMLMessage();
   }
}
```

7. References

7.1. Normative references

[CC] UN/CEFACT Core Components Technical Specification – Part 8 of the ebXML Framework, 15 November 2003, Version 2.01

[ebCPPA] Oasis ebXML Collaboration-Protocol Profile and Agreement Specification Version 2.0, 23 September 2002

[ebMS] ebXML Message Service Specification Version 2.0, 1 April 2002. OASIS ebXML Messaging Service Technical Committee

[ISO3166] ISO 3166-1:1997 Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes. As amended by the ISO 3166 Maintenance agency

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee,"Hypertext Transfer Protocol, HTTP/1.1", June 1999.

[SSL3] A. Frier, P. Karlton and P. Kocher, "The SSL 3.0 Protocol", Netscape Communications Corp., Nov 18, 1996. 2780

[SOAP] W3C-Draft-Simple Object Access Protocol (SOAP) v1.1, Don Box, DevelopMentor; David Ehnebuske, IBM; Gopal Kakivaya, Andrew Layman, Henrik Frystyk Nielsen, Satish Thatte, Microsoft; Noah Mendelsohn, Lotus Development Corp.; Dave Winer, UserLand Software, Inc.; W3C Note 08 May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

[SOAPAttach] SOAP Messages with Attachments, John J. Barton, Hewlett Packard Labs; Satish Thatte and Henrik Frystyk Nielsen, Microsoft, Published Oct 09 2000 <http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211>

[XLINK] W3C XML Linking Recommendation, <http://www.w3.org/TR/2001/REC-xlink-20010627/>

7.2. Non-normative references

[ebCPP] ebXML Collaboration Protocol Profile and Agreement specification, Version 1.0, published 10 May, 2001, <http://www.ebxml.org/specs/ebCCP.doc>

[JAXM] Java API for XML Messaging, <http://java.sun.com/xml/jaxm/>

[XMLSchema] W3C XML Schema Recommendation,
<http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/>
<http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>
<http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>

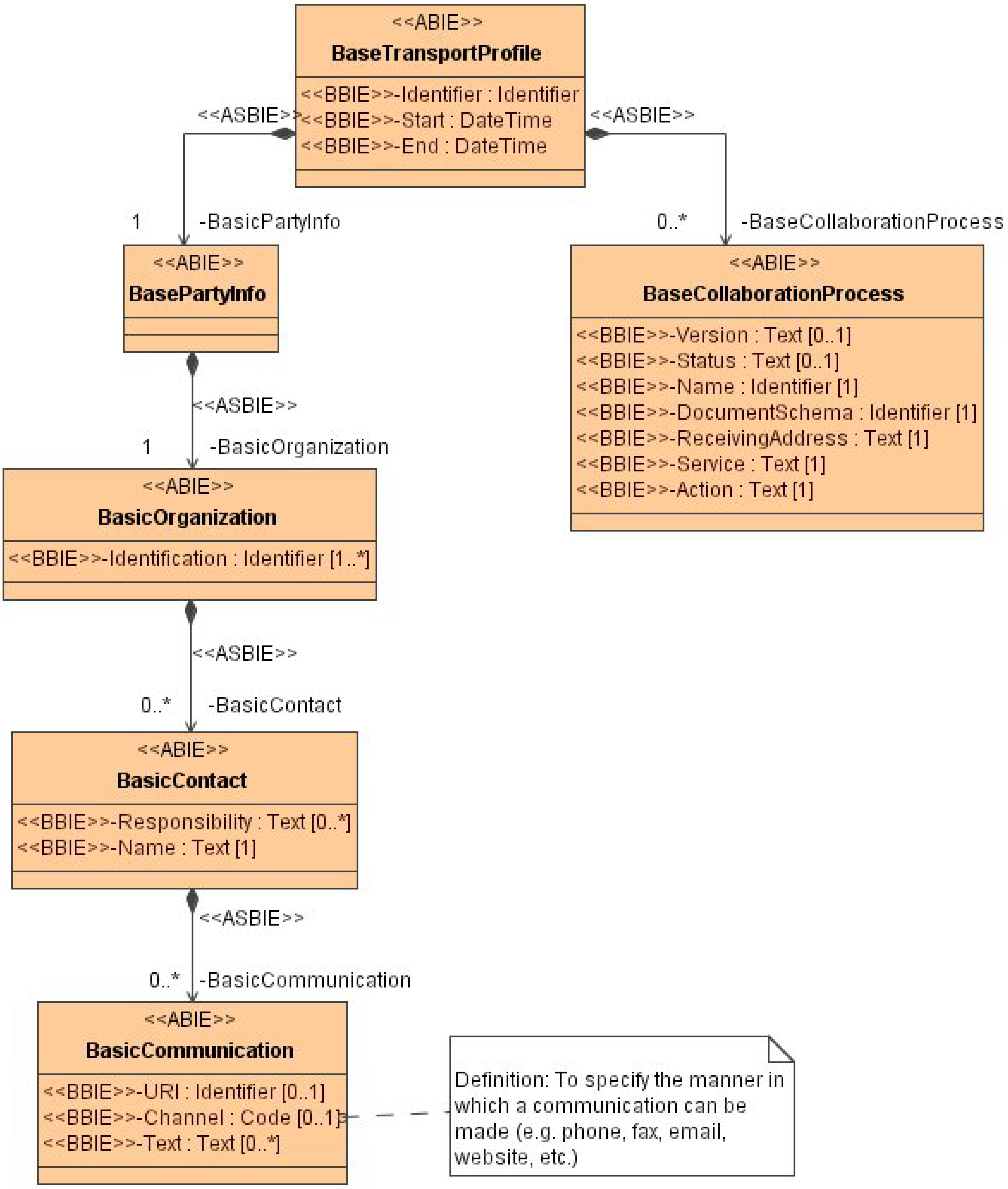
7.3. *The ebMS Security profiles*

Present in baseline MSH		Persistent digital signature	Non-persistent authentication	Persistent signed receipt	Non-persistent integrity	Persistent confidentiality	Non-persistent confidentiality	Persistent authorization	Non-persistent authorization	Trusted timestamp	**Description of Profile**
✓	Profile 0										no security services are applied to data
✓	Profile 1	✓									Sending MSH applies XML/DSIG structures to message
	Profile 2		✓						✓		Sending MSH authenticates and Receiving MSH authorizes sender based on communication channel credentials.
	Profile 3		✓				✓				Sending MSH authenticates and both MSHs negotiatea secure channel to transmit data
	Profile 4		✓		✓						Sending MSH authenticates, the Receiving MSH performs integrity checks using communications protocol
	Profile 5		✓								Sending MSH authenticates the communication channel only (e.g., SSL 3.0 over TCP/IP)
	Profile 6	✓					✓				Sending MSH applies XML/DSIG structures to message and passes in secure communications channel
	Profile 7	✓		✓							Sending MSH applies XML/DSIG structures to message and Receiving MSH returns a signed receipt
	Profile 8	✓		✓			✓				combination of profile 6 and 7
	Profile 9	✓								✓	Profile 5 with a trusted timestamp applied
	Profile 10	✓		✓						✓	Profile 9 with Receiving MSH returning a signed receipt
	Profile 11	✓					✓			✓	Profile 6 with the Receiving MSH applying a trusted timestamp
	Profile 12	✓		✓			✓			✓	Profile 8 with the Receiving MSH applying a trusted timestamp
	Profile 13	✓				✓					Sending MSH applies XML/DSIG structures to message and applies confidentiality structures (XMLEncryption)
	Profile 14	✓		✓		✓					Profile 13 with a signed receipt
	Profile 15	✓		✓						✓	Sending MSH applies XML/DSIG structures to message, a trusted timestamp is added to message, Receiving MSH returns a signed receipt
	Profile 16	✓				✓				✓	Profile 13 with a trusted timestamp applied Profile
	Profile 17	✓		✓		✓				✓	Profile 14 with a trusted timestamp applied
	Profile 18	✓						✓			Sending MSH applies XML/DSIG structures to message and forwards authorization credentials [SAML]
	Profile 19	✓		✓				✓			Profile 18 with Receiving MSH returning a signed receipt
	Profile 20	✓		✓				✓		✓	Profile 19 with the a trusted timestamp being applied to the Sending MSH message
	Profile 21	✓		✓		✓		✓		✓	Profile 19 with the Sending MSH applying confidentiality structures (XML-Encryption)
	Profile 22					✓					Sending MSH encapsulates the message within confidentiality structures (XML-Encryption)

7.4. Definitions of some Core Components

7.4.1. Formalised description of the details of transport profiles

Chapter 5 outlines how formalised ”business cards” of transport profile data could be organised, as an optional feature for implementing transport profile data maintenance. It is suggested that those who consider implementing it should take [ebCPPA] as a starting point, and reduce it to the needs of this profile. The following description is based on draft Core Components published by UN/CEFACT. (Note. The example is intended for the role of “receiver”.)



<<ABIE>>
BaseTransportProfile
<<BBIE>>-Identifier : Identifier
<<BBIE>>-Start : DateTime
<<BBIE>>-End : DateTime
<<ASBIE>>
<<ASBIE>>
1
-BasicPartyInfo
0..*
-BaseCollaborationProcess
<<ABIE>>
BasePartyInfo
<<ABIE>>
BaseCollaborationProcess
<<BBIE>>-Version : Text [0..1]
<<BBIE>>-Status : Text [0..1]
<<BBIE>>-Name : Identifier [1]
<<BBIE>>-DocumentSchema : Identifier [1]
<<BBIE>>-ReceivingAddress : Text [1]
<<BBIE>>-Service : Text [1]
<<BBIE>>-Action : Text [1]
<<ASBIE>>
1
-BasicOrganization
<<ABIE>>
BasicOrganization
<<BBIE>>-Identification : Identifier [1..*]
<<ASBIE>>
0..*
-BasicContact
<<ABIE>>
BasicContact
<<BBIE>>-Responsibility : Text [0..*]
<<BBIE>>-Name : Text [1]
<<ASBIE>>
0..*
-BasicCommunication
<<ABIE>>
BasicCommunication
<<BBIE>>-URI : Identifier [0..1]
<<BBIE>>-Channel : Code [0..1]
<<BBIE>>-Text : Text [0..*]
Definition: To specify the manner in which a communication can be made (e.g. phone, fax, email, website, etc.)