papiNet

Global Transaction Standards for the Paper Supply Chain

papiNet Standard - Version 2.00

Interoperability Guidelines
July 2002

Industry Review

Copyright

Copyright 2000 – 2002 papiNet G.I.E ("papiNet"), International Digital Enterprise Alliance, Inc. ("IDEAlliance"), and American Forest & Paper Association, Inc. ("AF&PA"), collectively "Copyright Owner". All rights reserved by the Copyright Owner under the laws of the United States, Belgium, the European Economic Community, and all states, domestic and foreign. This document may be downloaded and copied provided that all copies retain and display the copyright and any other proprietary notices contained in this document. This document may not be sold, modified, edited, or taken out of context such that it creates a false or misleading statement or impression as to the purpose or use of the papiNet specification, which is an open standard. Use of this Standard, in accord with the foregoing limited permission, shall not create for the user any rights in or to the copyright, which rights are exclusively reserved to the Copyright Owner.

papiNet (formerly known as the European Paper Consortium for e-business - EPC), IDEAlliance (formerly known as the Graphic Communications Association - GCA), the parent organisation of IDEAlliance the Printing Industries of America (PIA), the American Forest and Paper Association (AF&PA), and the members of the papiNet Working Group (collectively and individually, "Presenters") make no representations or warranties, express or implied, including, but not limited to, warranties of merchantability, fitness for a particular purpose, title, or non-infringement. The presenters do not make any representation or warranty that the contents of this document are free from error, suitable for any purpose of any user, or that implementation of such contents will not infringe any third party patents, copyrights, trademarks or other rights. By making use of this document, the user assumes all risks and waives all claims against Presenters.

In no event shall Presenters be liable to user (or other person) for direct, indirect, special or consequential damages arising from or related to any use of this document, including, without limitation, lost profits, business interruption, loss of programs, or other data on your information handling system even if Presenters are expressly advised of the possibility of such damages.

Use of Documents in papiNet Implementations

Documents may be used as templates for a papiNet implementation. The Presenters grant the right to modify and edit them to fit an actual implementation project provided all copies retain and display the copyright and any other proprietary notices contained in this document. Such modified documents must not be distributed beyond the trading partners implementing or maintaining a papiNet connection.

Additional Copyright Information

Additional copyrights may be referenced throughout this document in the appropriate section.

Table of Contents

Industry Review

Sponsoring Organisations

papiNet G.I.E.

papiNet started as a project within the Publishing part of the paper industry. A number of major paper suppliers, large German publishers and printers worked together to create a common standard for electronic business transactions. Soon there was a requirement to develop supporting software for handling the transactions via the Internet. This work turned out so well, that it was decided to enlarge the pilot project to encompass other parts of the industry. papiNet G.I.E. currently has 25 suppliers and wide cooperation with customers within the industry.

IDEAlliance

IDEAlliance is the leading global membership organisation for advancing the processes of information interoperability and dissemination of knowledge. IDEAlliance accomplishes this by engaging in and supporting the creation and adoption of globally recognised standards for information definition and exchange. Over the last 15 years, IDEAlliance's B2B Standards Committee has been the recognized e-business and Electronic Data Interchange (EDI) standards development body in the North American paper, print, and publishing industry. More information about the IDEAlliance can be found at http://www.idealliance.org/.

American Forest & Paper Association

The American Forest & Paper Association (AF&PA) is the national trade association in the United States of the forest, paper, and wood products industry. AF&PA members include manufacturers of over 80 percent of the paper, wood and forest products produced in the United States. AF&PA acts as the clearinghouse for statistical information, as the leading force in technical, regulatory and policy issues, and as the national voice for the forestry, wood, and paper industries. More information about the AF&PA can be found at http://www.afandpa.org/.

Preview

Interoperability Guidelines

Interoperability Overview[1]

The papiNet Standards Group has the vision of enterprises of any size and in any geographical location meeting and conducting the business of paper, printing, and publishing with each other through the exchange of XML based messages. The intent is to define a neutral method (one that is open and non-proprietary) for exchanging these electronic business messages. In addition to being neutral the message exchange process has to guarantee safe, secure delivery.

These interoperability objectives can be summarized in the following way:

- Participants in the messaging transfer process should be able to choose the technology they desire to use to communicate messages independent of other participants in the communications network.
- Errors in transmission to the destination must be communicated.
- Security must be assured:
  - Privacy - Protect against information being disclosed or revealed to any entity not authorized to have that information.
  - Authentication - Authenticate the claimed identity of the originator.
  - Authorization - Protect against the threat that unknown entities enter into the system and ensures that an entity performs only authorized actions within the system.
  - Integrity - Protect against the threat that the value of a data item might be changed en route.
  - Non-repudiation - Protect against one party to a transaction or communication later falsely denying that the transaction or communication occurred.

Who should read this document

Users who are designing a messaging service for use within a papiNet environment need to read this document. If you are only constructing the papiNet message (the payload to be sent in a messaging service) you most likely do not need this document.

[1] Portions of this section are extracted from Version 1.08 of the "ebXML Transport, Routing & Packaging – Message Service Specification" and as such are subject to the following copyright communication.

Copyright © UN/CEFACT and OASIS, 2001. All Rights Reserved
This document and translations of it MAY be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation MAY be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself MAY not be modified in any way, such as by removing the copyright notice or references to the ebXML, UN/CEFACT, or OASIS, except as required to translate it into languages other than English.

Historical Perspective

During the initial phases of the papiNet initiative it was felt that messaging standards were not sufficiently robust enough either in their definition or in the availability of neutral components to achieve the papiNet interoperability objectives. So, it was decided to provide a messaging service that contained this functionality.

As part of discussions associated with the joining of the European and North American initiatives[2] this decision was reviewed. It was determined that the ability of participants, in the messaging transfer process, to choose the technology desired, when communicating with other participants, outweighed the benefits of a common messaging service. The papiNet Interoperability Guidelines are a direct result of that decision. All users of the papiNet standard are encouraged to follow these guidelines, suggest improvements, or inform papiNet of additional requirements (www.papinet.org).

The tradition of providing open components to the industry has not been forgotten. In addition to the papiNet schemas we also provide stylesheets and a tool that can be used to communicate your use of the standard to your trading partners, all of these components are freely available.

Envelope Future Direction

While neutral communications components are more available and messaging standards are more robust. There still is a question of their ability to support "industrial strength" communications. Also, which messaging standard will provide long-term viability is still in question, albeit a less open-ended question than it was two years ago.

With this in mind, the papiNet Standards Group will be migrating towards full support for the SOAP – ebXML messaging protocol. At this point in time we are supporting Version 2.0c of the ebXML Message Service (without Digital Signatures but augmented with S/MIME).

papiNet's Approach to Developing Interoperability Guidelines

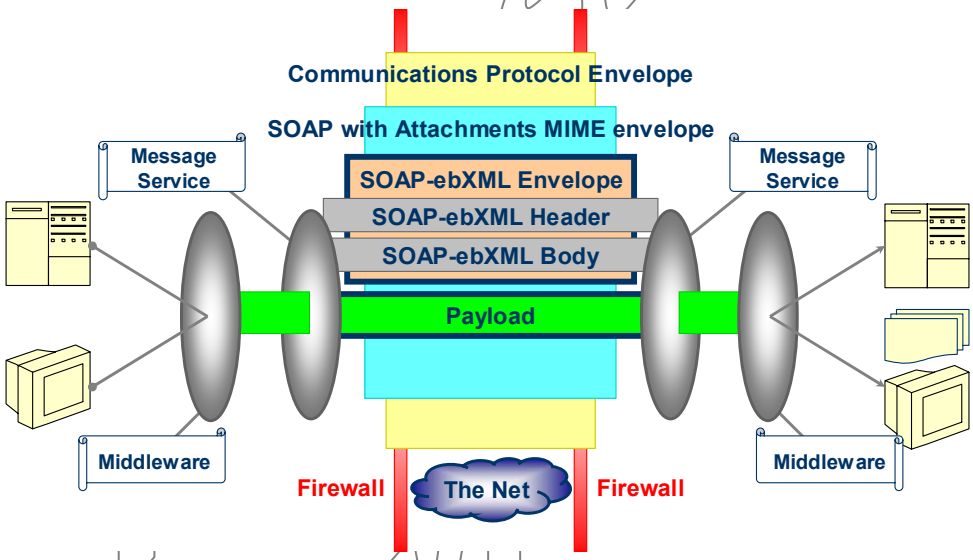
papiNet attempts to balance the benefits provided to the user community by providing a strict interoperability specification with the resultant drag on innovation and loss in functionality delivered to users. While it is our goal to cover the full spectrum of interoperability issues, this will take some time to complete and we anticipate a period of time where papiNet will be in a reactive mode. papiNet will work to minimize the amount of time that it takes to

[2] What began as a European and North American initiative has broadened into a process that is involving more countries and continents, every day.

respond to issues (check www.papiNet.org for Interoperability Guideline updates).

papiNet's Interoperability Scope

While transparency from the sender's application all the way through to the receiver's application would be ideal the papiNet Standards Group feels that the appropriate scope for the interoperability guidelines is assuring that the receiver of a papiNet message (the payload) is able to remove it from its envelope. How the receiver processes the message once it is unwrapped is outside the scope of papiNet interoperability recommendations.



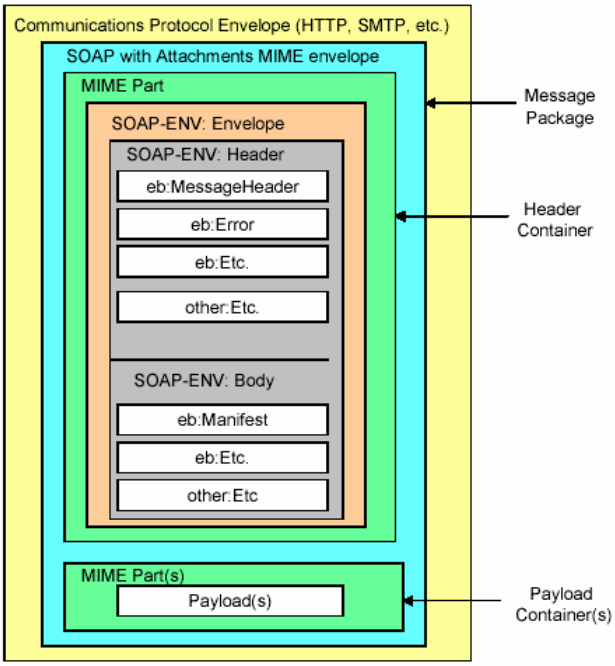
The graphic above[3] illustrates the papiNet interoperability scope. Bounded on either side by the users' particular message service, papiNet interoperability is achieved when the papiNet payload is unwrapped from its various envelopes. The bullets that follow further describe some of the elements of this graphic.

- Message Service
  - The process that receives incoming messages and sends outgoing messages.
- Middleware
  - The process that connects the message with the user's information system. In some situations a single package will serve both a message service and a middleware function.

[3] "Slice" vertically through graphic to determine what wrappings are in place at a given point in the transmission process.

SOAP Packaging Specification

The general structure and composition of a SOAP - ebXML message is illustrated to the right.[4] A complete discussion of this information can be found in the ebXML Message Service documentation (ebMS_v2_0rev_C at this writing). We'll provide some background to this specification so that you can follow the papiNet interpretation. We'll discuss the Header and Payload Container in this section and the Message Package in the following three sections.



In XML, SOAP-ENV would be written as soap:Envelope. This means, "the envelope as described in the soap namespace". This envelope is comprised of header and body elements (designated as SOAP-ENV:Header and SOAP-ENV:Body in the graphic).

In the above graphic soap:Header is comprised of MessageHeader and Error elements both in the ebXML namespace (designated as "eb"). The Error element is optional and in the papiNet implementation would not be necessary. Other elements may also be referenced in the soap:Header.

The graphic indicates that soap:Body contains eb:Manifest. The Manifest element contains information describing the message payload. Notice that there may be multiple payloads each identified as a separate MIME part. This approach serves to isolate the message payload from the routing information, supporting payload encryption, attachments, and other functions.

Message Service Functionality and Envelope Protocols

The mechanism that papiNet has selected to assure safe, secure delivery of business messages in a neutral manner is the SOAP – ebXML protocol. In addition to this standard we have determined that until an approved encryption standard is developed by the W3C/IETF the S/MIME (Version 2) extensions to the MIME standard should be used.

[4] From Message Service Specification, Version 2.0 rev C, OASIS ebXML Messaging Services Technical Committee, 21 February 2002

MIME and its extensions (S/MIME and S/MIME Certificates)

MIME (Multipurpose Internet Mail Extension[5]) is a standard system for identifying the type of data contained in a file based on its extension. MIME is an Internet protocol that permits binary files to be sent across the Internet. These files include graphics, photos, sound and video files, as well as formatted text documents. MIME negotiates many different operating systems and types of software.

The original MIME standard has been extended to include security and certificate functionality (S/MIME and S/MIME Certificates). Various issues are discussed in the documents referenced in the Technical References section with appropriate solutions. MIME has been implemented for decades and issues associated with its adoption have passed.

SOAP and SOAP with Attachments

SOAP is a lightweight protocol for the exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data-types, and a convention for representing remote procedure calls and responses. SOAP can be used in combination with a variety of other protocols.

SOAP is general enough that most implementations will have no problem implementing it. An ebXML compliant Message Service uses a SOAP with Attachments MIME envelope with S/MIME extensions permitted to provide security.

ebXML extensions to SOAP

The ebXML Message Service standard suggests several extensions to the SOAP standard to provide needed functionality. With the exception of the Digital Signature standard papiNet suggest adherence to these extensions.

Reliable Messaging

Reliable messaging defines a process that two Message Service Handlers can use to reliably exchange messages, using acknowledgment, retry and duplicate detection and elimination mechanisms, resulting in the *To Party* receiving the message Once-And-Only-Once. The process is flexible, allowing for both store-and-forward and end-to-end reliable messaging.

Reliability is achieved by a Receiving Message Service Handler responding to a message with an *Acknowledgment Message*. An *Acknowledgment Message* is any ebXML message containing an Acknowledgment element. Failure to receive an

[5] Although developed for email MIME is used for a variety of transport mechanisms.

Acknowledgment Message by a Sending Message Service Handler may trigger successive retries until such time as an *Acknowledgment Message* is received or the predetermined number of retries has been exceeded at which time the *From Party* must be notified of the probable delivery failure.

Whenever an identical message may be received more than once, some method of duplicate detection and elimination is appropriate, usually through the mechanism of a persistent storage mechanism. This functionality while desirable in a message service is not necessary for interoperability.

Reliable messaging through an ebXML containing an ***Acknowledgement*** element is the papiNet suggested approach. Users are cautioned to validate how their message service implements duplicate detection.

Industry Review

SOAP – ebXML Envelope Documentation

What follows is the papiNet interpretation of the SOAP – ebXML envelope specification (we will refer to this as the "papiNet-SOAP-ebXML envelope". It is papiNet's goal to be as consistent as possible with the official documentation (refer to the Technical Reference section); please bring any discrepancies to the attention of papiNet.org.

The next several sections review the papiNet-SOAP-ebXML envelope standard. The discussion is organized in the following way:

- General information.
- The SOAP envelope elements.
  - papiNet Interoperability Guidelines require that these elements and attributes must be understood and properly processed.
- The ebXML core envelope elements.
  - papiNet Interoperability Guidelines require that these elements and attributes must be understood and properly processed.
- The ebXML extended envelope elements.
- SOAP-ebXML attributes.

General information

Digital signatures

All digital signature information described in this standard should be considered, at this time, to refer to the equivalent S/MIME facility. As the W3C digital signature standard becomes more defined S/MIME may be augmented by the W3C digital signature process.

A review of the namespaces used

You'll run into a variety of namespaces in these sections:

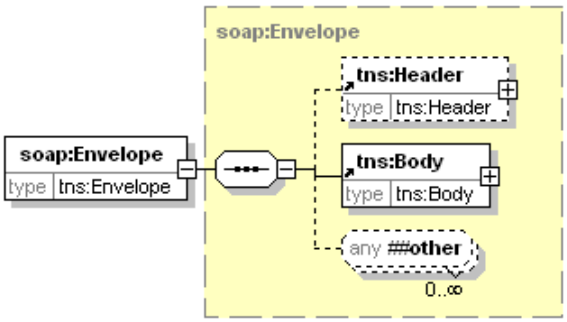
- eb – the ebXML namespace designator
- pn – the papiNet namespace designator
- soap – the SOAP namespace designator
- tns – the "target namespace identifier" this is how SOAP refers to ebXML.

In Review

The SOAP Envelope Elements

soap:Envelope

The ***soap:Envelope*** is a simple structure that provides for an optional ***Header***, a required ***Body***, and enables you to define additional ***Envelope*** children elements.

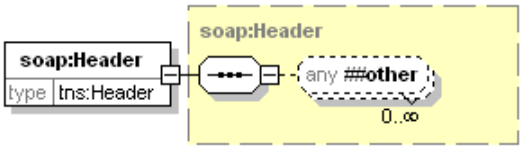


soap:Envelope contains the following elements:

- Header, while this is an optional ebXML extension element is required for the papiNet-SOAP-ebXML envelope.
- Body, required
- other elements are permitted but not validated within the papiNet-SOAP-ebXML envelope.

soap:Header

The SOAP ***Header*** element is the first child element of the SOAP ***Envelope*** element.

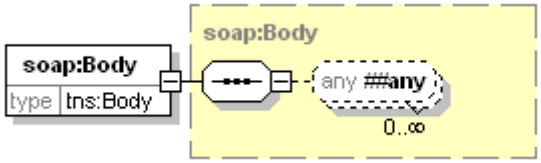


The following ebXML extension elements are substituted for the ##other content of the soap:Header:

- MessageHeader, required for the papiNet-SOAP-ebXML envelope.
- SyncReply, permitted but not required, must be processed if existing.

soap:Body

The SOAP ***Body*** element is the second child element of the SOAP ***Envelope*** element.



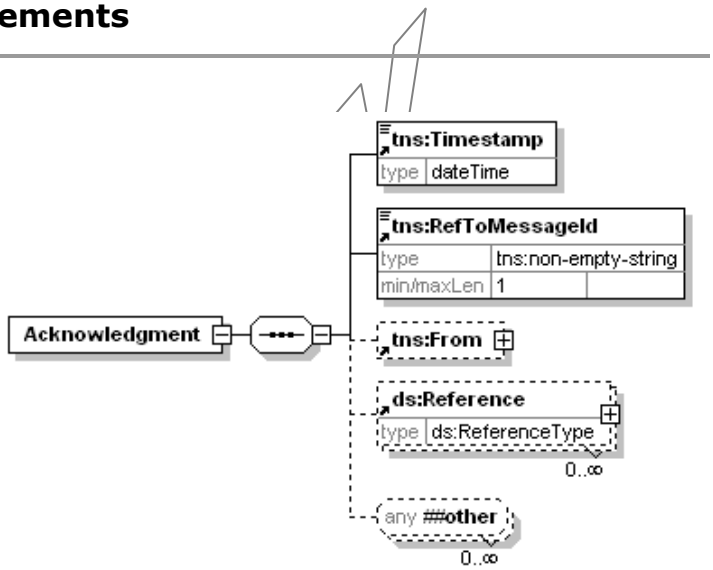
The following ebXML extension elements are substituted for the ##any content of the soap:Body:

- Manifest, optional but required for the papiNet-SOAP-ebXML envelope.
- SyncReply, permitted but not required.

Preview

The ebXML Core Envelope Elements

Acknowledgment



The optional ***Acknowledgment*** element is used by one Message Service Handler to indicate to another Message Service Handler that it has received a message. The ***RefToMessageId*** element in an ***Acknowledgment*** element is used to identify the message being acknowledged by its ***MessageId***.

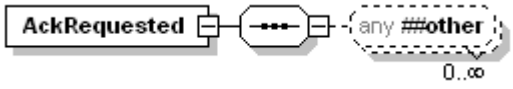
The ***Acknowledgment*** element consists of the following attributes:

- id, optional
- version, required
- soap:mustUnderstand, required with a value of "1"
- soap:actor, optional

The Acknowledgement element consists of the following elements:

- Timestamp, required
- RefToMessageId, required
- From, optional
- ds:Reference, optional
- other elements are permitted but not validated within the papiNet-SOAP-ebXML envelope.

AckRequested



The optional ***AckRequested*** element is used by the Sending Message Service Handler to request a Receiving Message Service Handler, acting in the role of the actor URI identified in the SOAP actor attribute, returns an Acknowledgment Message.

The ***AckRequested*** element contains the following attributes:

- id, optional
- version, required
- soap:mustUnderstand, required with a value of "1"
- soap: actor, optional
- signed, required

The ***AckRequested*** element is used to indicate to a Receiving Message Service Handler, acting in the role identified by the SOAP actor attribute,

In Review

whether an *Acknowledgment Message* is expected, and if so, whether the message should be signed by the Receiving Message Service Handler.

An ebXML Message may have zero, one, or two instances of an ***AckRequested*** element. A single Message Service Handler node should only insert one ***AckRequested*** element. If there are two ***AckRequested*** elements present, they MUST have different values for their respective SOAP ***actor*** attributes. At most one ***AckRequested*** element can be targeted at the ***actor*** URI meaning Next Message Service Handler and at most one ***AckRequested*** element can be targeted at the ***actor*** URI meaning To Party Message Service Handler for any given message.

Action

The required ***Action*** element identifies a process within a ***Service*** that processes the message. ***Action*** should be unique within the ***Service*** in which it is defined. The value of the ***Action*** element is specified by the designer of the service.

Action		
type	tns:non-empty-string	
min/maxLen	1	

In the context of the papiNet Standard ***Action*** is a value from ***pn:Document@DocumentName***.

ConversationId

The required ***ConversationId*** element is a string identifying the set of related messages that make up a conversation between two parties. It must be unique within the context of the specified ***CPAId***. The party initiating a conversation determines the value of the ***ConversationId*** element that shall be reflected in all messages pertaining to that conversation.

ConversationId		
type	tns:non-empty-string	
min/maxLen	1	

The ***ConversationId*** remains constant for all messages within a conversation. The value used for a ***ConversationId*** is implementation dependent.

In the context of the papiNet-SOAP-ebXML envelope the ***ConversationId*** is initiated with a papiNet message and follows through to the BusinessAcknowledgement after which the ***ConversationId*** is reinitiated. The ***ConversationId*** would encompass the original message, the ebXML acknowledgement, and the papiNet BusinessAcknowledgement. The implication of this approach is that multiple papiNet message documents in an envelope would not be allowed.

CPAId

The required ***CPAId*** element is a string that identifies the parameters governing the exchange of messages between the parties. The recipient of a

CPAId		
type	tns:non-empty-string	
min/maxLen	1	

In Review

message must be able to resolve the ***CPAId*** to an individual set of parameters, taking into account the sender of the message.

papiNet publishes a series of generic CPAs that can be referenced at www.papiNet.org

CPAId	URL	Discussion
BasicCPA	www.papiNet.org/CPA/BasicCPA.xml	Minimal CPA designed to support general papiNet communications.

Description

The ***Description*** element may be present zero or more times. Its purpose is to provide a human readable description of the purpose or intent of the message. The language of the description is defined by a required ***xml:lang*** attribute. Each occurrence should have a different value for ***xml:lang***.

In reference to the ***ErrorList*** element the content of the ***Description*** element provides a narrative description of the error. The XML parser or other software validating the message typically generates the message. The content is defined by the vendor/developer of the software that generated the ***Error*** element.

DuplicateElimination

The ***DuplicateElimination*** element, if present, identifies a request by the sender for the receiving Message Service Handler to check for duplicate messages.

The DuplicateElimination element must be used by the From Party Message Service Handler to indicate whether the Receiving Message Service Handler must eliminate duplicates. If the value of ***duplicateElimination*** in the CPA is "never", DuplicateElimination must not be present.

If ***DuplicateElimination*** is present:

- The To Party Message Service Handler must persist messages in a persistent store so duplicate messages will be presented to the To Party Application At-Most-Once
- The To Party Message Service Handler must adopt a reliable messaging behaviour causing duplicate messages to be ignored.

If ***DuplicateElimination*** is not present

- The To Party Message Service Handler is not required to maintain the message in persistent store and is not required to check for duplicates.
- A Receiving Message Service Handler is not required to check for duplicate message delivery. Duplicate messages might be delivered to an

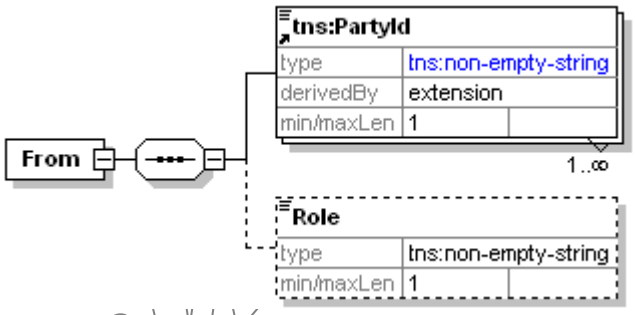
application and persistent storage of messages is not required - although elimination of duplicates is still allowed.

If the To Party is unable to support the requested functionality, or if the value of ***duplicateElimination*** in the CPA does not match the implied value of the element, the To Party should report the error to the From Party using an **errorCode** of "Inconsistent" and a ***Severity*** of "Error".

The papiNet-SOAP-ebXML envelope standard mandates that ***DuplicateElimination*** be present.

From

The required ***From*** element identifies the party that originated the message. Both the ***To*** and ***From*** elements can contain logical identifiers, such as a DUNS number, or identifiers that also imply a physical location such as an email address.



The ***From*** element contains the following elements:

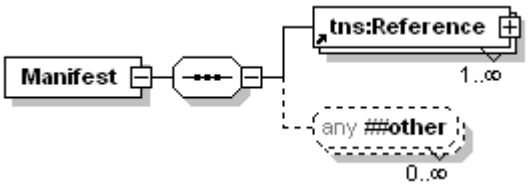
- PartyId, required
- Role, permitted but not validated in the papiNet-SOAP-ebXML envelope

If either the ***From*** or ***To*** elements contains multiple ***PartyId*** elements, all members of the list must identify the same organization. Unless a single ***type*** value refers to multiple identification systems, the value of any given ***type*** attribute must be unique within the list of ***PartyId*** elements contained within either the ***From*** or ***To*** element.

When used in the context of an ***Acknowledgment*** element, the ***From*** element contains the identifier of the party generating the Acknowledgment Message. If the ***From*** element is omitted then the party sending the element is identified by the ***From*** element in the ***MessageHeader*** element.

Manifest

An element pointing to any data present either in the Payload Container(s) or elsewhere, e.g. on the web.



In the context of the papiNet-SOAP-ebXML envelope the ***Manifest*** element must be present because there is information associated with the message not present in the ***Header***.

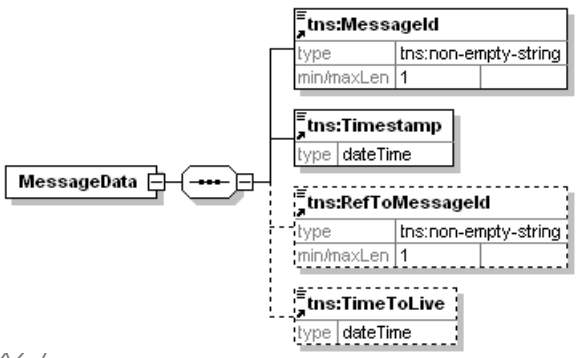
The Manifest element contains the following elements:

- Reference, required

- other elements are permitted but not validated within the papiNet-SOAP-ebXML envelope.

MessageData

The required ***MessageData*** element provides a means of uniquely identifying an ebXML Message.

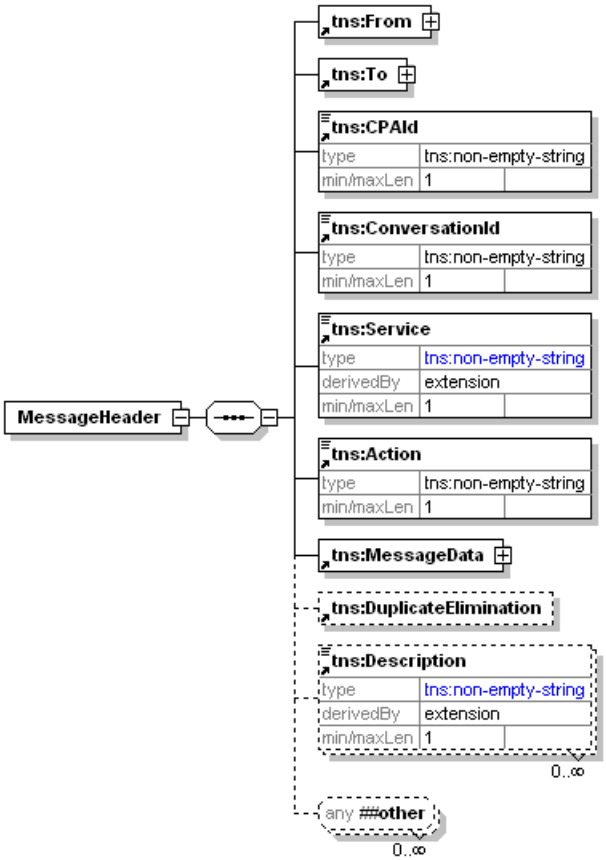


MessageData contains the following elements:

- MessageID, required
- Timestamp, required
- RefToMessageID, optional, must be processed if existing in the message
- TimeToLive, optional, must be processed if existing in the message

MessageHeader

The required ***MessageHeader*** element contains routing information for the message as well as other context information about the message.



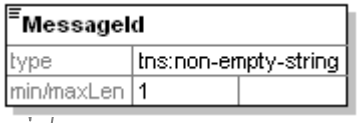
MessageHeader contains the following attributes:

- id, optional
- version, required
- soap:mustUnderstand, required

MessageHeader contains the following elements:

- From, required
- To, required
- CPAId, required
- ConversationId, required
- Service, required
- Action, required
- MessageData, required
- DuplicateElimination, optional
- Description, optional
- other elements are permitted but not validated within the papiNet-SOAP-ebXML envelope.

MessageId



The required element ***MessageId*** is a globally unique identifier for each message.

In the context of the papiNet-SOAP-ebXML envelope the ***MessageID*** would be equivalent to the deprecated ***TransferID*** which was "a unique ID provided by the communication software that identifies the transfer. This is used to provide a transactional transfer of the document in case of guaranteed delivery."

PartyId

The ***PartyId*** element is the identifier for the party that originated the message (in the case of ***From/PartyId***) or the receiver of the message (in the case of ***To/PartyId***).

PartyId contains the following attribute:

- ***type*** - The type attribute indicates the domain of names to which the string in the content of the PartyId element belongs.

  In the context of the papiNet-SOAP-ebXML envelope papiNet recognizes ***PartyID@type*** that conform to four different conventions:

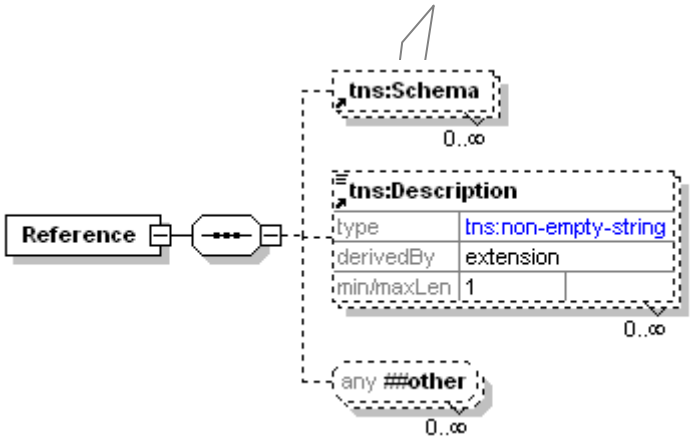
  - Content that is equivalent to the content of the global papiNet attribute ***pn:@PartyIdentifierType***.
  - The ebXML recommended ***type*** content, mentioned below.
  - An omitted type attribute as certain ***PartyID***s communicate their type explicitly (usually those ***PartyID***s that follow proper URI naming conventions).

ebXML recommends that the value of the ***type*** attribute be a URI and that these values be taken from either the EDIRA (ISO 6523), EDIFACT ISO 9735 or ANSI ASC X12 I05 registries. papiNet is developing recommendations that follow this approach.

In the context of the papiNet-SOAP-ebXML envelope the ***From/PartyId*** is equivalent to the deprecated ***SenderURI*** and the ***To/PartyId*** is equivalent to the deprecated ***ReceiverURI***.

Preview

Reference



The ***Reference*** element is used to communicate information about the content of the message.

The ***Reference*** element contains the following attributes:

- id, optional
- xlink:type – optional, this attribute defines the element as being an XLINK simple link. The value of 'simple' is fixed in the schema and cannot be changed.
- xlink:href – required, has a value that is the URI of the payload object referenced. It shall conform to the XLINK specification criteria for a simple link.
- xlink:role – this attribute identifies some resource that describes the payload object or its purpose. If present, then it shall have a value that is a valid URI in accordance with the XLINK specification,
- Any other namespace-qualified attribute may be present. A Receiving Message Service Handler may choose to ignore any foreign namespace attributes other than those defined above.

The ***Reference*** element contains the following elements:

- Schema, optional
- Description, optional
- other elements are permitted but not validated within the papiNet-SOAP-ebXML envelope.

RefToMessageId

The ***RefToMessageId*** element has a cardinality of zero or one. When present, it must contain the ***MessageId*** value of an earlier ebXML Message to which this message relates. If there is no earlier related message, the element must not be present.

In reference to the ***Acknowledgement*** element the required ***RefToMessageId*** element contains the MessageId of the message whose delivery is being reported.

Preview

Service

The required ***Service*** element identifies the service that acts on the message and it is specified by the designer of the service. The designer of the service may be a standards organization, or an individual or enterprise.

In the context of the papiNet-SOAP-ebXML envelope ***Service*** is either:

- Production
- Test

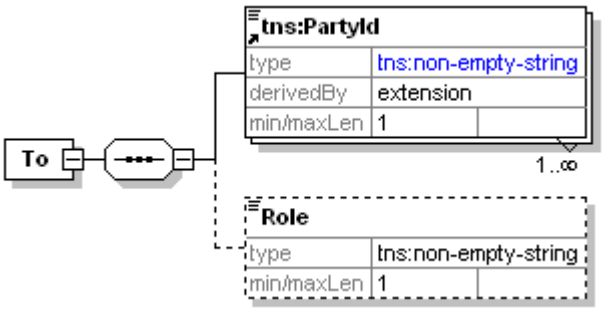
Timestamp

The required ***Timestamp*** is a value representing the time that the message header was created conforming to a ***dateTime*** and must be expressed as UTC. Indicating UTC in the Timestamp element by including the 'Z' identifier is optional.

In reference to the ***StatusResponse*** the ***Timestamp*** element contains the time the message, whose status is being reported, was received. This must be omitted if the message, whose status is being reported, is "NotRecognized" or the request was "UnAuthorized".

To

The required ***To*** element identifies the party that is the intended recipient of the message. Both ***To*** and ***From*** can contain logical identifiers, such as a DUNS number, or identifiers that also imply a physical location such as an email address.



The ***To*** element contains the following elements:

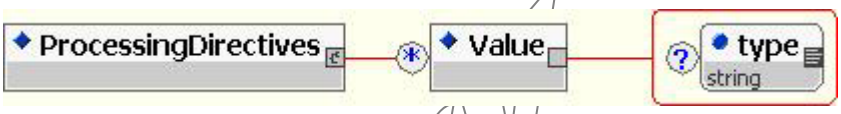
- PartyId, required
- Role, permitted but not validated in the papiNet-SOAP-ebXML envelope

If either the ***From*** or ***To*** elements contains multiple ***PartyId*** elements, all members of the list must identify the same organization. Unless a single ***type*** value refers to multiple identification systems, the value of any given ***type*** attribute must be unique within the list of ***PartyId*** elements contained within either the ***From*** or ***To*** element.

Preview

634 ### any ##other

635 The following
636 structure is included
637 in the
638 papiNet/ProcessingDirectives namespace for backwards compatibility and
639 extensibility.



@type	Description of Value content	
AttachmentDescription	The description of the attachment.	optional
Compression	Specifies the compression method.	optional
ConverterProductName	The product name of the software used to convert the internal ERP message to the XML format.	optional
ConverterVendorName	The name of the vendor of the converter software.	optional
ConverterVersion	The version of the converter software used.	optional
DTDSet	A grouping of DTDs.	optional
DTDVersion	The papiNet DTD version number as published in the reference. (V1R00, V1R10, or V2R00)	optional
ERPProductName	The product name of the ERP software used for generating the source of the message.	optional
ERPVendorName	The name of the vendor of the ERP software.	optional
ERPVersion	The version number of the ERP software.	optional
KeyLength	The length of the encryption key in bits.	optional
LogInfo	The sender may enter an arbitrary value here that has a meaning only in the local domain to help retrieving documents based on the value assigned	optional
MessengerProductName	The product name of the software used to transfer the message from the sender to the receiver.	optional
MessengerVendorName	The name of the vendor of the messenger software used.	optional
MessengerVersion	The version of the messenger software used.	optional
TransmissionProtocol	The protocol used for the transmission.	optional

640

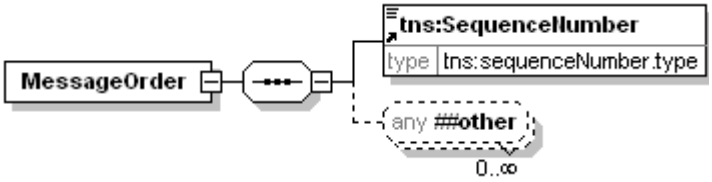
In Review

The ebXML Extended Elements

Error and ErrorList

The *ebXML Message Service* error reporting and handling module is a layer of processing above the SOAP processor layer and as such is not considered within the scope of the papiNet-SOAP-ebXML envelope documentation.

MessageOrder

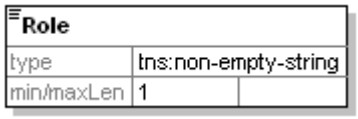


The ***MessageOrder*** element is an optional extension to the SOAP Header requesting the preservation of message order in this conversation. While ***MessageOrder*** is permitted within the papiNet-SOAP-ebXML envelope this type of processing is outside the scope of the papiNet-SOAP-ebXML envelope documentation.

MessageOrder contains the following element:

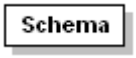
- SequenceNumber. required

Role



The ***Role*** element identifies the authorized role (***fromAuthorizedRole*** or ***toAuthorizedRole***) of the party sending (when present as a child of the ***From*** element) and/or receiving (when present as a child of the ***To*** element) the message. The value of the ***Role*** element is a non-empty string, which is specified in the CPA.

Schema

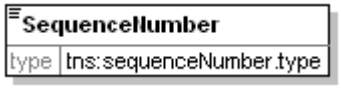


If the item being referenced has schema(s) of some kind that describe it then the ***Schema*** element should be present as a child of the ***Reference*** element. It provides a means of identifying the schema and its version defining the payload object identified by the parent Reference element.

The Schema element contains the following attributes:

- location - the required URI of the schema
- version - a version identifier of the schema

SequenceNumber

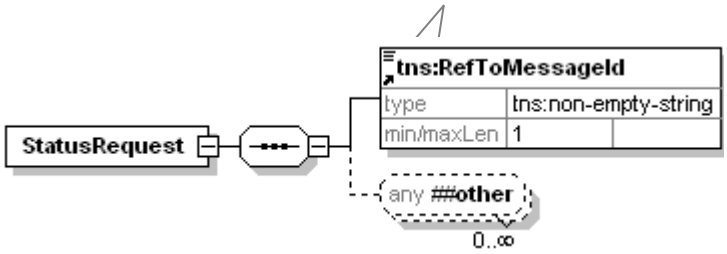


The required ***SequenceNumber*** element indicates the sequence a Receiving Message Service Handler must process messages. The SequenceNumber is unique within the ConversationId and Message Service Handler. It is used with ***MessageOrder***.

Preview

StatusRequest

The optional ***StatusRequest*** element is an immediate child of a SOAP Body and is used to identify an earlier message whose status is being requested.

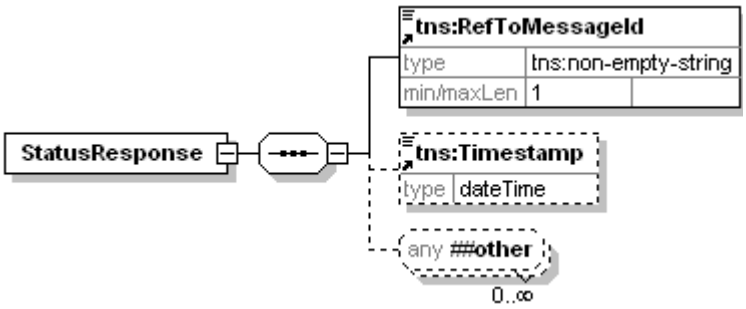


StatusRequest contains the following element:

- RefToMessageId, required

StatusResponse

The optional ***StatusResponse*** element is an immediate child of a SOAP Body and is used by one Message Service Handler to describe the status of processing of a message.



The StatusResponse element contains the following attributes:

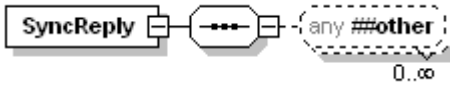
- id, optional
- version, required
- messageStatus, required

The ***StatusResponse*** element contains the following elements:

- RefToMessageId, required
- Timestamp, optional

SyncReply

The optional ***SyncReply*** element may be present on any outbound message sent using synchronous communication protocol. The ***SyncReply*** element may be present as a direct child descendant of the SOAP Header element. It contains the following attributes:



- id
- version
- soap:actor, required with the value of "http://schemas.xmlsoap.org/soap/actor/next"
- soap:mustUnderstand - with a value of "1"

If present, this element indicates to the receiving SOAP or ebXML Message Service Handler node the connection over which the message was received

In review

should be kept open in expectation of a response message to be returned via the same connection.

This element must not be used to override the value of ***syncReplyMode*** in the CPA. If the value of ***syncReplyMode*** is "none" and a ***SyncReply*** element is present, the Receiving Message Service Handler should issue an error with ***errorCode*** of "Inconsistent" and a ***severity*** of "Error".

TimeToLive

If the ***TimeToLive*** element is present, it must be used to indicate the time, expressed as UTC, by which a message should be delivered to the To Party Message Service Handler.

Industry Review

SOAP-ebXML Attributes

id

Each of the ebXML SOAP extension elements defined in this specification has an id attribute which is an XML ID that may be used to provide for the ability to uniquely identify the element within the message. This may be used when applying a digital signature to the papiNet-SOAP-ebXML message as individual ebXML SOAP extension elements can be targeted for inclusion or exclusion by specifying a URI of "#<idvalue>" in the Reference element.

messageStatus

The required messageStatus attribute identifies the status of the message identified by the RefToMessageId element. It shall be set to one of the following values:

- UnAuthorized – the Message Status Request is not authorized or accepted
- NotRecognized – the message identified by the RefToMessageId element in the StatusResponse element is not recognized
- Received – the message identified by the RefToMessageId element in the StatusResponse element has been received by the MSH
- Processed – the message identified by the RefToMessageId element in the StatusResponse element has been processed by the MSH
- Forwarded – the message identified by the RefToMessageId element in the StatusResponse element has been forwarded by the Message Service Handler to another Message Service Handler.

signed

The required signed attribute is used by a *From Party* to indicate whether or not a message received by the *To Party Message Service Handler* should result in the *To Party* returning a signed *Acknowledgment Message* – containing a XMLDSIG[6] Signature element. Valid values for signed are:

- true - a signed Acknowledgment Message is requested, or
- false - an unsigned Acknowledgment Message is requested.

Before setting the value of the signed attribute in AckRequested, the *Sending Message Service Handler* should check if the *Receiving Message Service Handler* supports *Acknowledgment Messages* of the type requested.

When a *Receiving Message Service Handler* receives a message with signed attribute set to true or false then it should verify it is able to support the type of *Acknowledgment Message* requested.

[6] Until XMLDSIG (digital signatures) is embraced by the papiNet-SOAP-ebXML envelope. S/MIME signatures will be the approach taken.

If the *Receiving Message Service Handler* can produce the *Acknowledgment Message* of the type requested, then it must return to the Sending *Message Service Handler* a message containing an Acknowledgment element.

If the *Receiving Message Service Handler* cannot return an *Acknowledgment Message* as requested it must report the error to the *Sending Message Service Handler* using an errorCode of "***Inconsistent"*** and a severity of either "***Error"*** if inconsistent with the CPA, or "***Warning"*** if not supported.

soap:actor

The URI urn:oasis:names:tc:ebxml-msg:actor:nextMSH when used in the context of the SOAP actor attribute value shall be interpreted to mean an entity that acts in the role of an instance of the ebXML Message Service Handler conforming to this specification.

- This actor URI has been established to allow for the possibility that SOAP nodes that are NOT ebXML Message Service Handler nodes may participate in the message path of an ebXML Message. An example might be a SOAP node that digitally signs or encrypts a message.
- All ebXML Message Service Handler nodes must act in this role.

The URI urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH when used in the context of the SOAP actor attribute value shall be interpreted to mean an instance of an ebXML Message Service Handler node, conforming to this specification, acting in the role of the Party identified in the MessageHeader/To/PartyId element of the same message. An ebXML Message Service Handler may be configured to act in this role.

- The Message Service Handler that is the ultimate destination of ebXML messages must act in the role of the To Party Message Service Handler actor URI in addition to acting in the default actor as defined by SOAP.

soap:mustUnderstand

The REQUIRED SOAP mustUnderstand attribute on SOAP Header extensions, namespace qualified to the SOAP namespace, indicates whether the contents of the element must be understood by a receiving process or else the message must be rejected in accordance with SOAP. This attribute with a value of '1' (true) indicates the element must be understood or rejected. This attribute with a value of '0' (false), the default, indicates the element may be ignored if not understood.

version

The REQUIRED version attribute indicates the version of the ebXML Message Service Header Specification to which the ebXML *SOAP* Header extensions conform. Its purpose is to provide future versioning capabilities. For conformance to this specification, all of the version attributes on any SOAP extension elements defined in this specification must have a value of "2.0".

Preview

802 An ebXML message MAY contain SOAP header extension elements that have a
803 value other than "2.0". An implementation conforming to this specification
804 that receives a message with ebXML SOAP extensions qualified with a version
805 other than "2.0" may process the message if it recognizes the version
806 identified and is capable of processing it. It must respond with an error if it
807 does not recognize the identified version.
808

Industry Review

Technical references associated with these standards.

To implement the papiNet Interoperability Guidelines you may need to reference the following documents:

- SOAP
  - http://www.w3.org/TR/SOAP
- SOAP with Attachments
  - http://www.w3.org/TR/SOAP-attachments
- S/MIME
  - http://www.imc.org/rfc2311
- S/MIME certificates
  - http://www.imc.org/rfc2312
- ebXML Messaging Service Version 2.0
  - http://www.oasis-open.org/committees/ebxml-msg/documents/ebMS_v2_0rev_c.pdf
- Collaboration Protocol Agreement
  - http://www.ebxml.org/specs/ebCCP.doc
- XLink
  - http://www.w3.org/TR/xlink/

Any questions you might have should first be addressed by reviewing the standards documents and then by enquiring for technical assistance at www.papinet.org.

Industry Review

Sample XML with SOAP Envelope

```
SOAPAction: "ebXML"
Content-type: multipart/related; boundary="BoundarY"; type="text/xml";
      start="<papiNet_envelope_example>"

--BoundarY
Content-ID: < papiNet_envelope_example >
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://schemas.xmlsoap.org/soap/envelope/">
   <soap:Header xmlns:eb = "http://www.oasis-open.org/committees/ebxml-
msg/schema/msg-header-2_0.xsd" 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 eb:version = "2.0" soap:mustUnderstand = "1">
         <eb:From>
            <eb:PartyId>oid:1.3.6.1.4.1.13099</eb:PartyId>
         </eb:From>
         <eb:To>
            <eb:PartyId eb:type = "DunsNumber">125811117</eb:PartyId>
         </eb:To>
         <eb:CPAId>papiNetBasicCPA</eb:CPAId>
         <eb:ConversationId>oid:1.3.6.1.4.1.13099.999.1</eb:ConversationId>
         <eb:Service>PurchaseOrder</eb:Service>
         <eb:Action>Test</eb:Action>
         <eb:MessageData>
            <eb:MessageId>oid:1.3.6.1.4.1.13099.998.1</eb:MessageId>
            <eb:Timestamp>2002-05-14T14:51:00</eb:Timestamp>
         </eb:MessageData>
         <eb:DuplicateElimination/>
      </eb:MessageHeader>
   </soap:Header>
   <soap:Body xmlns:eb = "http://www.oasis-open.org/committees/ebxml-
msg/schema/msg-header-2_0.xsd" 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:Manifest eb:version = "2.0">
         <eb:Reference xmlns:xlink = "http://www.w3.org/1999/xlink" xlink:href =
"cid:papiNet_message_fragment.xml"/>
         <eb:Reference xmlns:xlink = "http://www.w3.org/1999/xlink" xlink:href =
"//www.papinet.org/envelope/#DTDVersion">
            <eb:Description xml:lang = "en-US">V2R00</eb:Description>
         </eb:Reference>
```

industry review

```
        </eb:Manifest>
    </soap:Body>
</soap:Envelope>
--BoundarY
Content-ID: <papinet_message_fragment.xml>
Content-Type: text/xml

<?xml version="1.0" encoding="UTF-8"?>
<PurchaseOrder xmlns="http://www.papiNet.org/papiNet"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=http://www.papiNet.org/papiNet PurchaseOrderV2R00.xsd
    <PurchaseOrderHeader>
…
        </PurchaseOrderHeader>
    <PurchaseOrderLine>
…
        </PurchaseOrderLine>
    <PurchaseOrderSummary>
…
        </PurchaseOrderSummary>
</PurchaseOrder>

--BoundarY--
```

Mapping the V1R10 Envelope to the V2R00 Envelope

papiNet Envelope	S/MIME, SOAP-ebXML mapping, or comments
TransmissionInformation/ TransmissionCharacteristics/	
@TransmissionProtocol	MessageHeader/ProcessingDirectives with @type = 'TransmissionProtocol' and appropriate content.
@TransmissionMode	MessageHeader/MessageMode/DuplicateElimination
TransferID	MessageHeader/MessageData/MessageID
SenderURI	MessageHeader/From/PartyId
ReceiverURI	MessageHeader/To/PartyId
TransmissionTimeStamp	MessageHeader/MessageData/Timestamp
TransmissionInformation/ TransmissionOrganisationIdentifiers/	
SenderOrganisation	MessageHeader/From/PartyId
SenderOrganisationUnit	Not supported through envelope content.
ReceiverOrganisation	MessageHeader/To/PartyId

Industry Review

papiNet Envelope	S/MIME, SOAP-ebXML mapping, or comments
ReceiverOrganisationUnit	Not supported through envelope content.
TransmissionInformation/ TransmissionSecurityCharacteristics/	
@HashAlgorithm	S/MIME – Content-Type micalg parameter
@SignatureAlgorithm	S/MIME – DigestEncryptionAlgorithmIdentifier
@CryptoAlgorithm	S/MIME – ContentEncryptionAlgorithmIdentifier
KeyLength	S/MIME oid
CommunicationSoftware/	
ConverterProductName	MessageHeader/ProcessingDirectives with @type = 'ConverterProductName' and appropriate content.
ConverterVendorName	MessageHeader/ProcessingDirectives with @type = 'ConverterVendorName' and appropriate content.
ConverterVersion	MessageHeader/ProcessingDirectives with @type = 'ConverterVersion' and appropriate content.
MessengerProductName	MessageHeader/ProcessingDirectives with @type = 'MessengerProductName' and appropriate content.
MessengerVendorName	MessageHeader/ProcessingDirectives with @type = 'MessengerVendorName' and appropriate content.
MessengerVersion	MessageHeader/ProcessingDirectives with @type = 'MessengerVersion' and appropriate content.
Payload/MessageMetaData/Amendments/	
Ammendment@Element	Handled by the ConversationId concept. This type of tracking is removed from the envelope and is managed using persistant storage.
Ammendment@KindOfChange	
Ammendment@DateTime	
Ammendment/PreviousValue	
Payload/MessageMetaData/DocumentInfo	
@MessageName	MessageHeader/Service
@TestFlag	MessageHeader/Action
@Type	MessageHeader/Service@type
LogInfo	MessageHeader/ProcessingDirectives with @type = 'LogInfo' and appropriate content.
DTDVersionNumber	MessageHeader/ProcessingDirectives with @type = 'DTDVersionNumber' and appropriate content.

papiNet Envelope	S/MIME, SOAP-ebXML mapping, or comments
ApplicationSoftware/	
ERPProductName	MessageHeader/ProcessingDirectives with @type = 'ERPProductName' and appropriate content.
ERPVendorName	MessageHeader/ProcessingDirectives with @type = 'ERPVendorName' and appropriate content.
ERPVersion	MessageHeader/ProcessingDirectives with @type = 'ERPVersion' and appropriate content.
DTDSet	MessageHeader/ProcessingDirectives with @type = 'DTDSet' and appropriate content.
Compression	MessageHeader/ProcessingDirectives with @type = 'Compression' and appropriate content.
AttachmentDescription	MessageHeader/ProcessingDirectives with @type = 'AttachmentDescription' and appropriate content.

902

903

In Public Review