ROSETTANET
POWERED BY GS1 US

Multiple Messaging Services (MMS) Profile for ebMS 3.0

Specification Information	
Name	Multiple Messaging Services Profile for ebMS 3.0
Version	R12.00.00A
Date	30 October 2009

Table of Content

Legal Disclaimer

Copyright

Trademarks

Document Version History

Version	Date	Document History
R12.00.00A	30 Oct 2009	Published as Released Version

Acknowledgement

This document has been prepared by RosettaNet (www.RosettaNet.org) from requirements gathered during the Foundational Program and in conformance with the methodology. Listed below are the legal entities that contributed to the design and development of this specification.

MMS ebMS Team

Members	Company
Durand, Jacques	OASIS
Moberg, Dale	Axway
Stojanovic, Nikola	RosettaNet
Chew, Jasmine	RosettaNet
Hussam El-Leithy	RosettaNet

1. Introduction

1.1. Document Conventions

This document occasionally uses terms that appear in capital letters. When the terms "MUST", “REQUIRED”, “SHALL”, "SHOULD", "RECOMMENDED", “MAY”, “OPTIONAL”, "MUST NOT", “NOT REQUIRED”, “SHALL NOT” and "SHOULD NOT" appear capitalized, they are being used to indicate particular requirements of this profiling specification. The meaning of these terms is to be interpreted as defined in [RFC2119].

1.2. General Intent and Scope

1.2.1. Intent

RosettaNet implementations currently require users to buy a messaging system capable of running the RosettaNet Implementation Framework (RNIF). Although such RNIF systems are robust and widely adopted for XML payloads in the high-tech industry, this solution has the following drawbacks:

- Such RNIF messaging systems are not commonly used by other vertical markets. As a result, companies who support both the high tech industry and other verticals are forced to support more than one messaging standard for e-business transactions.
- They do not represent a viable solution for small and medium businesses (SMBs).

To alleviate the problem, RosettaNet created a Multiple Messaging Services (MMS) initiative that included three additional messaging systems, ebMS, AS2 and Profile for Web Services WS

In a first phase of the MMS project, a profile for ebMS V2.0 has been defined. Since then, a new version of the ebMS standard has been completed in OASIS (ebMS V3.0) which has the following advantages compared to ebMS V2.0:

- Message pulling feature. This supports partners with limited connectivity (intermittent connectivity, lack of static IP addresses) who can only behave as pure clients, pushing messages to a server, and pulling messages from this server.
- Integration with back-end Web Services. The format of ebMS V3 messages is fully compatible with protocol-level Web services standards. As a result, integration with back-end Web services is facilitated, while still ensuring B2B decoupling between partners: WSDL definitions and their subsequent upgrades only need to be known from the ebMS gateway or the Enterprise Service Bus, while the messaging middleware – connecting message handlers over the Internet - supports all QoS and connectivity modes.
- Native support for advanced security features, such as non-repudiation of receipts and service-level authorization.

The purpose of this document is to recommend to users and developers how best to use an ebMS messaging V3 handling system to transport RosettaNet PIP business messages between trading partners. Including ebMS V3 in this way will add another messaging option for trading partners transporting RosettaNet business payloads and may lower the infrastructure investments required to exchange RosettaNet payloads across trading networks that employ ebMS V3.

1.2.2. In-Scope of this Profile

This document is concerned with the profiling and the configuration of an ebXML framework, for carrying RosettaNet PIPs according to the requirements identified in the MMS - Abstract Message Definition (AMD) document.

The approach we have taken is to focus on ebMS 3.0, include part of the CPA and limit choreography to the lowest level covering one action PIPs.

- **ebXML version**: V 3.0. The ebXML context for this profiling document includes ebBP2.0 (in RN PIPs: BPSS 1.01, though a customized version – validate against 1.01 at least. Some references will be made to ebBP 2.0 when appropriate). Profiling of CPPA 2.0 / CPPA 2.1. is added in appendix, although it is expected that CPPA V3 should be used instead when complete.
- **Choreography**: The approach is here to cover simple choreographies, involving one or two business messages (action messages in RosettaNet), augmented with ebMS signals message (e.g. Receipts, Errors, PullRequest) when QoS requirements demand it.
- **Subset of Agreements**: Although ebMS is the target, the intent of this profiling goes beyond just wire interoperability and addresses some agreement aspects (CPA). Defining CPA templates or guidelines is the best way to represent the out-of-band agreement required for a practical deployment of PIPs. Only the part of the CPA that relates to messaging and maps to PIP definition data, will be profiled.

1.2.3. Out-Of-Scope of this Profile

- **Multi-hop**. Part 2 of ebMS V3 is still in the design process at the time this V3 profile is written. This includes routing functions in multi-hop environments.

1.3. General Methodology

- The Trading Partner Profiles (TPP) and resulting Abstract Trading Partner Agreement (ATPA) is a good starting point for users. Although the core of the ebXML profiling described here is defined solely based on mapping RNIF features into ebXML features (via the MMS-AMD [AMD] requirements), the ATPA represents parameters that need to be defined in order to complete a user-specific profiling of ebXML for a PIP deployment.
- From the TPP info, a CPP (Collaboration Protocol Profile) template can be filled for each partner, and a resulting agreement (TPA) can be mapped to a subset of the CPA. However, the suggested approach is for a business entity to directly define a partially-filled CPA with its capabilities and communication requirements (a “CPA template”), then share this CPA template with its business partner(s) who will complete it. The resulting document is a CPA instance.
- This CPA instance will be somehow orthogonal to PIPs: the deployment of several PIPs may share the same CPA instance. Conversely, several instances of the same PIP may use different CPAs, based on requirements that are specific to the nature of the document contents and other business considerations. Tools exist for producing the Collaboration Role part of a CPP or CPA directly from a 2.0 ebBP Business Collaboration specification.
- This document describes **profiling rules** for ebMS V3 and for CPA 2.1. These rules, when applied to data that is specific to business partners (TPA) and specific to targeted PIPs, will define specific **messaging profiles**.
- ebMS can be used with ebBP2.0 (and former BPSS versions as well) or CPPA, and so both ebBP2.0 and CPPA can also be used when ebMS is used for RosettaNet. No

attempt will be made to produce a complete profile for CPPA or ebBP2.0 for RosettaNet in this document. However, it will occasionally be explained what CPPA or ebBP2.0 features would need to be in a CPPA or ebBP2.0 that governed ebMS messaging when used for RosettaNet. These features can be understood as configuration input for the ebXML MSH mode of operation, as they may affect the MSH behavior without necessarily affecting the message header. For example, for GS1 EDIINT there exists a standard ebBP set of templates that comply with GS1 recommended configurations.

Some general rules of ebMS profiling:

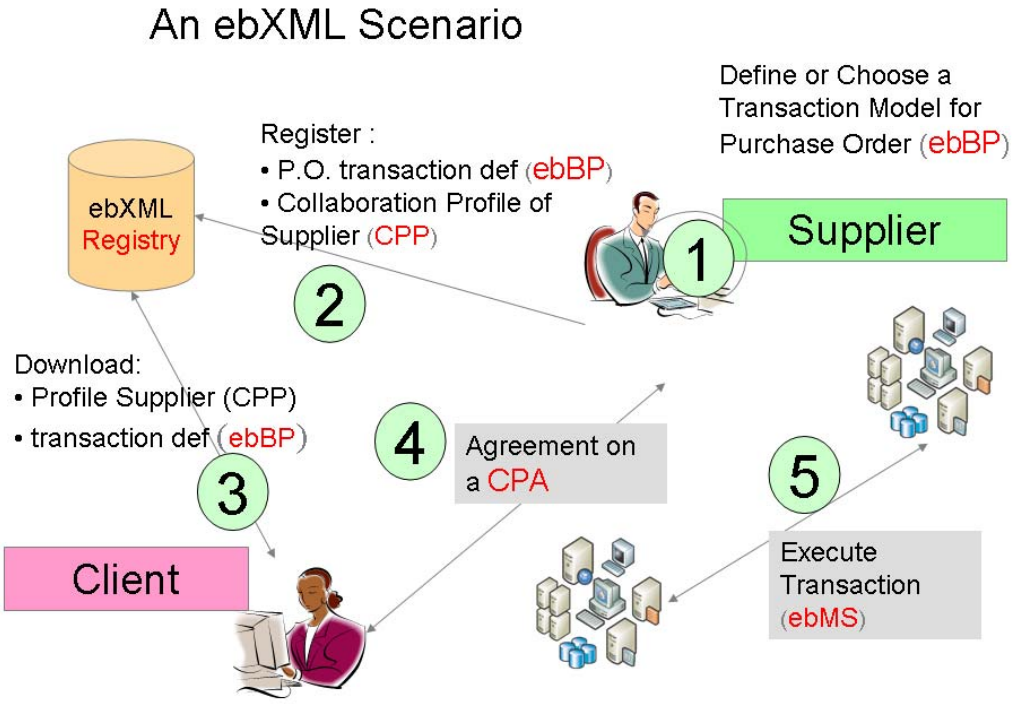
- The proposed profiling in this document does not make use of customization points in the ebMS3 header – in particular MessageProperties – so that a broader set of MSH implementations can be used.
- The ebMS header will fulfill the functions of the RNIF Delivery header, although it does not contain all the information present in the Delivery header. Some elements of the ebMS header will also map to elements from the Service header.
- Clearly there is more data in Standard Business Document Header (SBDH) and/or RNIF Service Header than can be represented in ebMS headers (without using extensions). In case the SBDH needs be preserved as is – e.g. for reuse of legacy back-end integration -, then the entire SBDH header can be added as a distinct payload part (referenced by a separate eb:PartInfo element in the header). It is RECOMMENDED to add it in the SOAP Body instead of as an attachment.

2. ebXML Messaging Overview

2.1. The ebXML Framework

The ebXML specifications support the exchange of business messages required to conduct electronic trading relationships between business partners. These capabilities logically separate but allow coordinated use of five key technologies important for eBusiness:

- Communicating data in common terms using a defined methodology (Core Components)
- Defining business processes and assembling business transactions (ebXML Business Process Specification Schema, ebBP)
- Providing secure and reliable transport (ebXML Messaging Service [ebMS])
- Registering and making available key eBusiness artifacts and services (ebXML Registry Services [ebRS] and Registry Information Model [ebRIM])
- Providing a technical configuration contract between business partners (Collaboration Protocol Profile and Agreements [CPP/CPA])



The business case and requirements that prompted the development of the set of ebXML specifications were to:

- Provide a migration path for and leveraging of EDI-compatible technologies.
- Develop openly accessible technologies for Small-Medium Enterprises whether in a managed or non-managed environment.

- Enable all supply-chain partners using XML technologies.
- Provide an integrated eBusiness approach focused on interoperability needs, while maintaining loose-coupling to backend systems.

The heterogeneous nature of eBusiness transactions require a flexible infrastructure and architectural framework that can support service calls (catalog status requests) and more complex document exchange (offers and acceptance).

The business document processing was decoupled from the messaging layer. The consumer of these documents may be a service, a business process instance, or middleware or business application interested in the business document contents. Such documents cannot be directly associated with an application service in a predefined way. The coupling between the messaging system and the consumers of these messages must be supported in an adaptable way.

2.2. ebXML Messaging

Messaging Overview

The ebMS protocol runs over several transport protocols, and provides bindings for HTTP and SMTP**.** It supports several message exchange patterns including push and pull that map to different types of business transactions. ebMS V3 improves on ebMS V2 on several aspects such as message pulling, non-repudiation of receipt, and full compliance with Web service protocols such as WS-ReliableMessaging and WS-Security. V3 is also compliant with related WS-I Basic Profiles [WS-I].

Status of ebXML Messaging Standards

- The ebMS V2 was approved in 2002, and, in May 2004, submitted and accepted an ISO standard, ISO/TS 15000-2.
- The ebMS v3.0 OASIS Standard was approved in September 2007 [ebMS3]. An adjunct document on “Conformance Profiles for ebMS V3” [ebMS3-CP] has been defined. A second part to ebMS V3 is scheduled for end of 2009, which defines multi-hop and routing support as well as message bundling rules.

3. Trading Partner Agreement

3.1. TPA and Infrastructure Deployment Parameters

This table shows the relationships between TPA and ebXML components. It shows how elements of a TPA relate to the ebXML components. A dependency in the table indicates that some features of the ebXML component are concerned by the TPA item, or even further, that the feature must comply with the profiling described in this document.

The elements of a TPA that do not directly affect the standardized features of an ebXML component may still concern this component, but its impact will be at implementation, administration or deployment level, i.e. will affect the operational aspect, which is outside the scope of this profiling guideline.

- An "F" means that the answer to the question will affect features that are specified in the related standard (ebMS, CPPA or ebBP2.0). In other words, a compliant implementation to these standards explicitly supports answers to this question.
- An "X" means that depending on the user answer to the question, the usage of this ebXML component will be affected in a way that must comply with the profiling defined in this document. In other words, you need to be aware of this profiling in order to implement the answer in a compliant way.
- No mark means no direct effect or dependency on features that are specified in the related standard. The answer to the question becomes more a product implementation or deployment issue (how well this is handled depends more on additional product features than on conformance to the standard.)

TPA Item	***ebMS V3***	***CPPA*** ***(partial profiling here)***	***ebBP*** ***(no profiling defined here)***
General Operation Parameters			
Specify the standard you will use to identify trading partners. (DUNS, GLN, other authority name)	FX	FX	
What is the maximum volume of messages you expect to exchange with any specific trading partner? What is the average?			
What is your peak time interval?			
What is the average size of the message during the peak time interval?			
How many messages do you receive and send during the peak time interval?			
Will you use peer-to-peer routing or will you be using a messaging service (MS)(temporary message store)?			

Will you operate via a message Hub, or directly point-to-point?	FX	F	
If using a Hub, which message data will be used for doing routing?			
Which of these Message Exchange Patterns will you be using? One-Way, Request-Response, Notification, Solicit-Response	FX	FX	F
Will you need to correlate messages from a single PIP instance, or will you need to correlate messages from several PIP instances, as belonging to a same long-lasting conversation (e.g. for monitoring purpose)	FX	FX	F
Will you be using a specific registry? If Yes, please specify			
Will you be defining specific roles for each party of each business process?	FX	FX	F
Indicate the level of Management Services you will require for your messaging system	F		
Do you need to track messages based on their participation in complex business processes?	FX (maybe)		
Do you need status services?	F (maybe)		
Do you need remote management?			
Do you need monitoring?	FX (maybe)		
Do you need BI support?			
Do you require disaster recovery?			
Do you require Debugging capabilities?			
Indicate the level of Connectivity you will have with the Internet			
Are you occasionally connected (dialup with modem?)	FX	F	
Are you permanently connected with a permanent IP address (T1, DSL)?	FX	F	

Are you permanently connected without a permanent IP address (Cable)?	FX	F	
Indicate the Security Services you will require for your messaging system			
Do you need the headers to be encrypted?	FX	FX	
Do you need the payload to be encrypted? Using which method? (S/MIME, XML Encryption)	FX	FX	
Do you need non-repudiation of origin?	F	F	
Do you need non-repudiation of receipt?	FX	FX	
If need non-repudiation of receipt, is it required to have a digest of the acknowledged message in the acknowledgement, or is a simple reference to MessageID sufficient?	FX		
Do you need to digitally sign your messages (i.e.:X.509)?	FX	FX	
Do you want to transport level encryption? (i.e.: TLS/SSL)?	F	F	
Do you require timestamp of your messages?	F		
Do you subscribe to authority domains? (i.e.: DUNS/GLN/EAN-UCC/DUNS+4/Vertical)	FX	F	
Indicate the Reliability you will require for your messaging system			
Do you need guaranteed message delivery? (include ACK signals)	FX	FX	
Do you need de-duping?	FX	FX	
Do you require Ordered Delivery?			
Do you require a Manifest?	F		
Do you require expiration control?	F		
Indicate what you will need to compress in your messages			
Do you need to compress headers?	F		
Do you need to compress payload?			
Do you need to compress attachments?			

Indicate the Error Handling you will require for your messaging system			
Do you need Message Service error handling (too many retries, etc.)?	F	F	
Do you need Message Content error handling (invalid message, etc.)?			
Do you need out of band error notification?			
Indicate the Payload Capabilities you will require for your messaging system			
Do you need Globalization/I18?	F		
Do you need attachments?	F		
Do you need payload validation?			
What type of payload will you be exchanging, ASCII, Binary or both?			
Will you need to support different versions of PIPs?			
Indicate the requirements imposed by integration with existing software			
Do you need to preserve (some) previous message header structures (not native to ebMS) as is, so that you can reuse back-end binding technology?	FX		
Do you need to move message header data into your back-end?	FX		

3.2. CPP and CPA Profiling

Only CPPA V2.1 has been profiled here (Appendix 9). Use of CPP V3.0 once completed, is RECOMMENDED. Many profiling aspects of CPPA V2.1 defined here, can be transposed to subsequent CPPA versions.

4. Message Description

4.1. General approach for ebMS 3.0 profiling

4.1.1. ebMS message headers and RosettaNet message headers

RNIF Headers:

The existing RNIF headers are dealt with in the following way:

- The RNIF Preamble header is not supposed to appear anymore in the ebMS message.
- The RNIF Delivery header is not supposed to appear anymore in the ebMS message. It is replaced by the ebMS header. Although not all the information encoded in the Delivery header will be translated into the ebMS header, this header is not relevant anymore in the ebXML context.
- The RNIF Service header may still be present in the ebMS message, in case it is needed for binding a message to existing PIP software that needs to be reused with ebXML. Some but not all of its elements map to the ebMS header. The Service header may be preserved as a separate attachment in the ebMS payload. However, in case of conflict between data in Service header and analogous data in ebMS header, the ebMS header will prevail as long as the messaging transfer is still in progress

Standard [Business] Document Headers (SBDH):

Not all information in the SBDH will map to ebMS headers. In case where an SBDH structure is present and used in the binding of the message with backend processes, it is recommended to keep the SBDH in the payload. However, in case of conflict between data in SBDH and analogous data in ebMS header, the ebMS header will prevail as long as the messaging transfer is still in progress (this includes routing in multi-hop environments).There are two options to consider:

1) The PIP document is defined using XML schema. In this case the SBDH is bound to the PIP document, and both are represented in the SOAP Body. The ebMS3 message has then only one MIME part: the SOAP envelope in which the eb:Messaging SOAP header contains the ebMS2 header, and the SOAP Body contains the payload (SBDH + PIP).
2) The PIP document is defined using DTD. In general, no SBDH instance is defined. Additional meta-data that may be needed and that is not in ebMS headers, is supposed to be found in the Service header -, and in FromRole + ToRole elements of PIPs. The Service Header can be preserved as a distinct payload part in the ebMS payload if needed. In case the SBDH is defined for this PIP and added to the message, it should also be sent as a distinct payload part in the ebMS payload. In both cases, it is RECOMMENDED to add it in the SOAP Body instead of as an attachment.

ebMS header and its extensions:

For better interoperability between users, a deployment that conforms to this messaging profile MUST NOT use such extensions on any element under the eb:Messaging element. It is NOT RECOMMENDED to use the eb:MessageProperties child element of eb:Messaging.

Integrating with existing infrastructures:

In order to preserve the ability to reconstruct the former RNIF structure (minus preamble and delivery headers), in case some users consider this a good integration approach with existing systems, the Service header MAY be included as a separate MIME part in the ebMS3 message, i.e. as an additional SOAP attachment element of the ebMS payload.

4.1.2 Message Bundling and Batching

- **Payloads Bundling** is the grouping of several payloads inside the same ebMS User Message unit (the eb:UserMessage element will refer to everyone of these payloads) . There is a single set of ebMS V3 “business headers”, i.e. a single eb:Messaging SOAP header block.
- **User Message Bundling** is the grouping of several ebMS User Message units (eb:UserMessage elements) inside the same eb:Messaging SOAP header block. There is still a single SOAP envelope, but several ebMS business headers. This practice simply consists of packaging several ebMS messages in the same SOAP envelope (and leaving all additional payloads as separate attachments of the same MIME envelope). Except for security and reliability that will be processed once per SOAP envelope, the receiver MSH will process each User Message unit individually as if it had been received separately. The difference with payload bundling is that a set of well-formed ebMS messages is bundled, not just their payloads.
- **Message batching** is the ability to nest several SOAP ebMS messages (each one with its own SOAP envelope and individual headers) within the same MIME message. This option is not considered here.

4.2. ebMS V3 Header Profiling

The tables below are borrowed from the Deployment Profile Template 1.1 for ebMS 2.0, [DPT-ebMS2] a document guide for deploying ebMS that has been developed by the ebXML Implementation, Interoperability and Conformance OASIS TC.

4.2.1. Profile Requirement Item eb:PartyId

Specification Feature	Header elements: eb:UserMessage/eb:PartyInfo/eb:From/eb:PartyId eb:UserMessage/eb:PartyInfo/eb:From/eb:PartyId/@type eb:UserMessage/eb:PartyInfo/eb:To/eb:PartyId eb:UserMessage/eb:PartyInfo/eb:To/eb:PartyId/@type
Specification Reference	ebMS 3 [ebMS3], section 5.2.2.2
Profiling	One instance of PartyId (in case several exist) must have as value either a DUNS (or DUNS+4) or Global Location Number (GLN). Both should not be found at the same time under the same From or To element. When several PartyId are present, the one above should be the first PartyId element. It is allowed to have additional PartyId elements in eb:From or in eb:To (they just need to have different @type values) • PartyID values MUST comply with ISO 6523 values, when applicable. That includes DUNS and GLN identifiers. • PartyID type attribute MUST be used to represent the Domain name and ICD (International Code Designator) according to section 24 “PartyID” of the CPPA V2.1 specification, found in: http://www.oasis-open.org/committees/ebxml-cppa/documents/ebCPP-2_1.pdf (April 2005)

More generally:

- the type value of the type attribute MUST be a URN. If the type attribute is present, then it provides a scope or namespace for the content of the PartyId element.
- if the type attribute is not present, the content of the PartyId element MUST be a URI that conforms to [RFC2396].

If an abbreviated name is described in the item titled “Name of Coding System” within the ICD list, it should be used, followed by the ICD value.

Example:

<tp:PartyId

tp:type="urn:oasis:names:tc:ebxml-cppa:partyid-type:D-U-N-SNumber:0060">123456789</tp:PartyId>

Where “0060” is the ICD value of D-U-N-S Number.

A GLN value can be used instead (see at the end of this section) e.g.:

tp:type="urn:oasis:names:tc:ebxml-cppa:partyid-type:EANLocationCode:0088"> … </tp:PartyId>

Alignment

- MUST map to (RNIF, Delivery Header) element: PartnerIdentification / GlobalBusinessIdentifier.
- MUST map to (Standard Bus. Doc Header - SBDH) element: PartnerIdentification (choice of Duns, Duns+4 or GLN), when applicable.
- MUST map to ebXML CPPA 2.0 or 2.1 element: PartyInfo/PartyId, when used.
- Value and type MUST conform to ISO 6523 when applicable, and the type attribute to section 24 “PartyID” of the ebXML CPPA V2.1 specification.

Test References

An implementation SHOULD provide support, and be able to accommodate, the usage of the type values standardized herein:

- If an abbreviated name is described in the item titled “Name of Coding System” within the ICD list (see ISO 6523), a type attribute can be constructed by prepending: “urn:oasis:names:tc:ebxml-cppa:partyid-type:” to the abbreviated name and appending a colon “:” followed by the ICD value. For example, using the abbreviated name D-U-N-S Number:

  Abbreviated Name: “D-U-N-S Number

  Upper-camel-case resultant string: “D-U-N-SNumber”

  tp:type=" urn:oasis:names:tc:ebxml-cppa:partyid-type:D-U-N-SNumber:0060"

  Note: “0060” is the ICD value of D-U-N-S Number.

  To be consistent with v2 CPP/A, the value that follows remains a valid type attribute value or content for the PartyId element:
  “urn:oasis:names:tc:ebxml-cppa:partyid-type:duns”.

- Because an abbreviated name may be omitted from the ICD list, the type attribute can always contain the string derived from “Name of Coding System” expressed in upper-camel-case. A value can always be constructed by pre-pending “urn:oasis:names:tc:ebxml-cppa:partyid-type:” to the upper-camel-case name and appending a colon “:” followed by the ICD value. For example, using the formal name of the Name of Coding System: “Data Universal Numbering System”:

  Transformed Camel-case: “DataUniversalNumberingSystem”

  tp:type=" urn:oasis:names:tc:ebxml-cppa:partyid-type:DataUniversalNumberingSystem:0060"

- Punctuation marks in these formal names (such as, “/”, “-“or “’”) should be included unless they are not allowed in URNs [RFC2141]. If the punctuation characters are not allowed in URNs, then the hexadecimal escaping convention explained in [RFC2141] should be followed for characters

Given the directives in ISO 6523 related to Global Location Numbers (GLN) (**ICD**: 0088, **Name of coding** : EAN Location Code system, the above example yields a GLN type value of:

urn:oasis:names:tc:ebxml-cppa:partyid-type: EANLocationCode:0088

4.2.2. Profile Requirement Item eb:Role

Specification Feature	Header elements: eb:UserMessage/eb:PartyInfo/eb:From/eb:Role eb:UserMessage/eb:PartyInfo/eb:To/eb: Role
Specification Reference	ebMS 3, section 5.2.2.3, 5.2.2.5

Profiling	Role Name of the partner in this business transaction (in Partner Role Description of the PIP)
	Example:
	`<eb:To>`
	`<eb:PartyId eb:type="`
	`urn:oasis:names:tc:ebxml-cppa:partyid-type:D-U-N-SNumber:0060">`
	`myDUNS</PartyId>`
	`<eb:Role>Seller</eb:Role>`
	`</eb:To>`

Alignment	• MUST map to (RNIF, Service Header) element: from{to}Role/GlobalPartnerRoleClassificationCode • MUST map to ebXML ebBP2.0: the role value maps to the corresponding BinaryCollaboration/Role/@name in ebBP2.0 (or BPSS 1.*) definition, when used. • MUST map to ebXML CPPA 2.0 or 2.1 element: CollaborationRole/Role/@name, when used.
Test References	

4.2.3. Profile Requirement Item eb:Service

Specification Feature	Header element: eb:UserMessage/eb:CollaborationInfo/eb:Service
Specification Reference	ebMS 3, Section 5.2.2.8
Profiling	This value MUST be the same as the one used in the ebBP2.0 instance (or BPSS) for the PIP (*). Its format must be: **"urn:rosettanet:specification:interchange:PIP" +<alphanumeric name of PIP> + ":xml:ebbp:"+ <PIP VersionIdentifier>.** The Service element MUST have same value for all messages involved in a single PIP or TPIR-PIP (whether it is an Action, Confirmation or Signal message). **Example:** If in ebBP2.0: nameID="**urn:rosettanet:specification:interchange:PIP7C7:xml:ebbp:v11_00**" version="**V11.00**" **In ebMS header:** <eb:Service> urn:rosettanet:specification:interchange:PIP7C7:xml:ebbp:v11_00</eb:Service>
Alignment	• MUST map to ebXML ebBP 2.0 or BPSS 1.* element when used: ProcessSpecification/@uuid or if not present, ProcessSpecification/@NameId • MUST map to ebXML CPPA 2.0 or 2.1 element, when used: Service/@name
Test References	

(*) XSD (Modular) PIPs have BPSS documents defined while DTD (Monolithic) PIPs do not.

4.2.4. Profile Requirement Item eb:Action

Specification Feature	Header element: eb:UserMessage/eb:CollaborationInfo/eb:Action
Specification Reference	ebMS 3, section 5.2.2.9
Profiling	In case of an Action message, the value MUST be consistent with Global Business Action Code (camel case version of this value). eb:Action value needs to correspond to ServiceHeader/ProcessControl/ActivityControl/MessageControl/Manifest/ ServiceContentControl/ActionIdentity/GlobalBusinessActionCode In case of a RosettaNet signal, the value MUST be consistent with the following: If Signal is positive (ReceiptAcknowledgment):

	Value= "ReceiptAcknowledgment" If Signal is negative (Exception): **Value= "Exception"** **Examples:** <eb:Action>Purchase Order Request</eb:Action> <eb:Action>Purchase Order Change</eb:Action> <eb:Action>Shipment Receipt Notification</eb:Action>
Alignment	• MUST map to ebXML ebBP2.0 or BPSS 1.* element when used: **RequestingBusinessActivity/@nameId, RespondingBusinessActivity/@nameId** • Map to ebXML ebBP2.0 element when used: BinaryCollaboration//@name or BusinessCollaboration/@name, Or, more precisely, the no-space version of these values.
Test References	

4.2.5. Profile Requirement Item eb:ConversationId

Specification Feature	Header element: eb:UserMessage/eb:CollaborationInfo/eb:ConversationId
Specification Reference	ebMS 3, section 5.2.2.10
Profiling	It is RECOMMENDED that ConversationId represents the PIP instance ID value, i.e. has same value for all messages related to the same PIP instance. In other words, messages from the same PIP instance MUST have same ConversationID, and it is recommended that this ConversdationID be unique to this PIP instance (not shared with other PIP instances).
Alignment	• MUST map to Standard Business Doc Header (SBDH**)** element when applicable**: RequestingDocumentInformation / BusinessProcessInstanceIdentifier** • MUST map to (in RNIF Service header) element: `ServiceHeader/ProcessControl/pipInstanceId/InstanceIdentifier`
Test References	

4.2.6. Profile Requirement Item eb:RefToMessageId

Specification Feature	Header element: eb:UserMessage/eb:MessageInfo/eb:RefToMessageId
Specification Reference	ebMS 3, section 5.2.2.1

Profiling	As a reminder, it MUST be used only for: (1) Relating business signals messages to action messages, by referencing the ebMS ID. (2) Relating a business response to a business request. Every message involved in a PIP instance MUST refer to another previous message of this instance (except for the initial message of the instance, which MUST NOT have a RefToMessageId element.) If several PIP instances must be correlated, this MUST NOT be achieved by this value (the first message of a PIP instance MUST NOT refer to another PIP).
Alignment	• MUST map to RNIF Service header element, in the sense it plays a similar role: `ServiceHeader/ProcessControl/ActivityControl/MessageControl/inReplyTo/messageTrackingID`
Test References	

4.2.7. Profile Requirement Item eb:MessageId

Specification Feature	Header element: eb:UserMessage/eb:MessageInfo/eb:MessageId
Specification Reference	ebMS 3, section 5.2.2.1
Profiling	Used for uniquely (globally) identifying a message (either signal or action). Normally, this identifier is automatically generated by an MSH, and out of control from applications. (However, an MSH provides visibility to applications on this value, so that an application can use it for referencing (see eb:RefToMessageId).
Alignment	• MUST map to RNIF Delivery header, in the sense it plays a similar role: `DeliveryHeader/messageTrackingID.InstanceIdentifier`.
Test References	

4.2.8. Profile Requirement Item eb:AgreementRef

Specification Feature	Header element: eb:UserMessage/eb:CollaborationInfo/eb:AgreementRef (NOTE: in ebMS V2, this corresponds to the element eb:MessageHeader/eb:CPAId)
Specification Reference	ebMS 3, section 5.2.2.7
Profiling	See Section “CPA Profiling and Sample” in Appendix A, for recommended profiling.

Alignment	
Test References	

4.2.9. Profile Requirement Item eb:PayloadInfo

Specification Feature	Header element: eb:UserMessage/eb:PayloadInfo/eb:PartInfo eb:UserMessage/eb:PayloadInfo/eb:PartInfo/eb:Schema
Specification Reference	ebMS 3, section 5.2.2.12, 5.2.2.13

Profiling	Element eb:PartInfo: SHOUD|LD include an eb:Schema element when the eb:PartInfo element is referring to the service content part (main XML document) of a PIP payload, or is referring to an XML RosettaNet signal. When present it MUST use an URN identifying the schema or DTD that applies to the part. The schema URN identifier MUST comply with **[RN-NameSpaces].**
	Example for a PIP service content with XML schema:
	`urn:rosettanet:specification:interchange:PIP3A4PurchaseOrderRequest:xsd:schema:1.0`
	When a legacy RNIF header (such as Service header) is included in the message, it must be added as a single attachment. The eb:Reference element SHOULD contain an eb:Schema element to identify it, which conforms to **[RN-NameSpaces].**
	Example for a Service header:
	`urn:rosettanet:specification:system:ServiceHeader:dtd:schema:2.0`
	NOTE: The use of an XLINK processor should not be required.

Alignment	
Test References	

5. Message Processing

5.1. The Processing Modes (P-Modes)

A P-Mode (Processing Mode) can be seen as configuration data that controls the way each message of a kind is processed by the Message Handler, when sent or received.

Because different messages may be subject to different types of processing, an MSH generally supports several P-Modes.

On a Sending MSH, together with the information provided by the application layer for each submitted message, the P-Mode fully determines the content of the message header. For example, the "security" part of the P-Mode will specify certificates and keys, as well as which messages will be subject to these. This in turn will determine the content of the Security header.

The association of a P-Mode with a message may be based on various criteria, usually dependent on header data (e.g. Service/Action, Conversation ID, or other message properties). Which security and/or which reliability protocol and parameters, as well as which Message Exchange Pattern (MEP) is being used when sending a message, is determined by the P-Mode associated with this message.

5.2. Packaging

The packaging of the message headers and payloads, including security headers, follows the ebMS 3.0 specification. It is automatically implemented by conforming ebMS V3 MSH implementations. The content of ebMS header elements is largely controlled by PMode parameters. These parameters in turn reflect profiling decisions made in section 4 (for business headers) and in section 6 (for the MEP).

5.3. Un-packaging

The un-packaging of the message headers and payloads, including security headers, follows the ebMS 3.0 specification. It is automatically implemented by conforming ebMS MSH implementations.

6. Supported IT Scenarios and Message Exchange Patterns

Partners implementing a RosettaNet PIP may not always have advanced infrastructure or persistent Internet connection. This Profile supports two kinds of business partners:

- A **pure-client business partner** does not run a messaging server (e.g. does not run an HTTP server), does not have a static IP address, and cannot receive incoming request messages (e.g. HTTP requests). It can have varying QoS capabilities (reliability, security).
- A **server-enabled business partner** is running a messaging server, is an addressable endpoint to which messages can be sent directly (e.g. HTTP requests). It can have varying QoS capabilities (reliability, security), generally more complete than a pure-client partner.

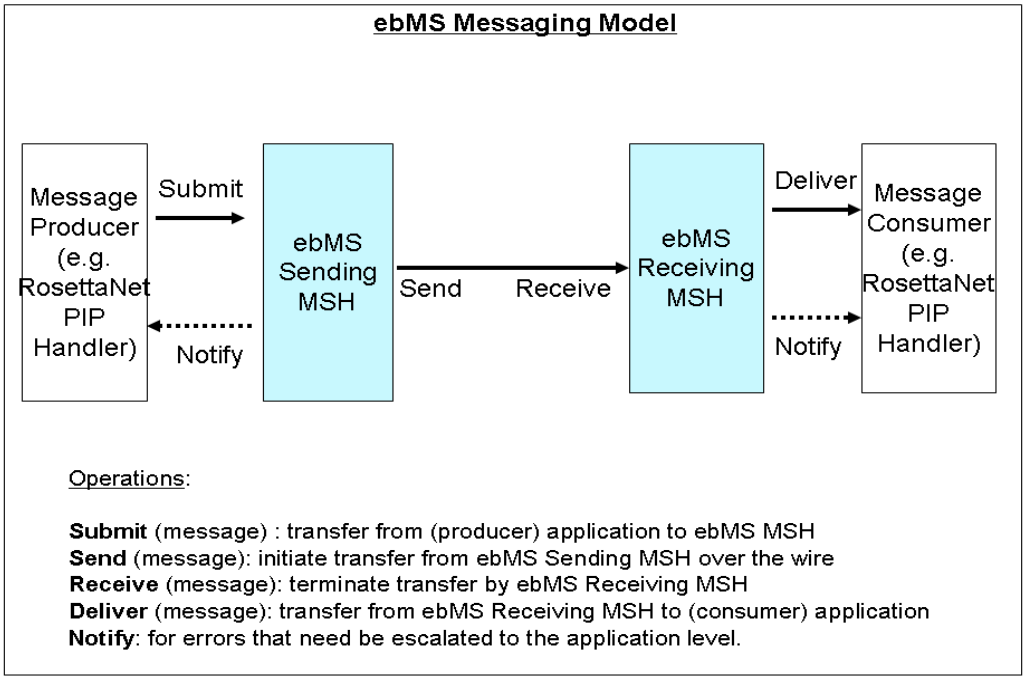
Accordingly, PIP interactions are supported in two basic IT scenarios:

- **Server to server**: Interactions between two server-enabled business partners
- **Pure-client to server**: Interactions between a pure-client business partner and a server-enabled business partner.

6.1. Message Exchange Patterns (MEPs) in ebMS 3.0

This section describes how ebMS MEPs map to PIPs involving action message(s), and potentially some RosettaNet signal message(s).

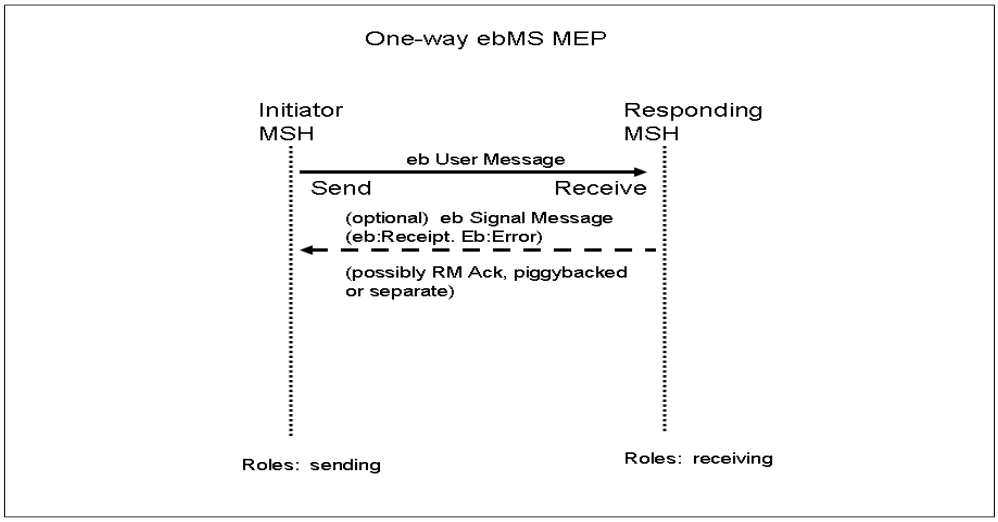
The general messaging model in ebMS V3 is illustrated in the following figure. The model of integration with RosettaNet "PIP handler" is of a layering: PIP handling code is acting as the "application layer" for an ebMS Message Service Handler, communicating with the ebMS MSH via abstract operations (Submit, Deliver). An MSH will act in either one or both of the following roles: Sending and Receiving.



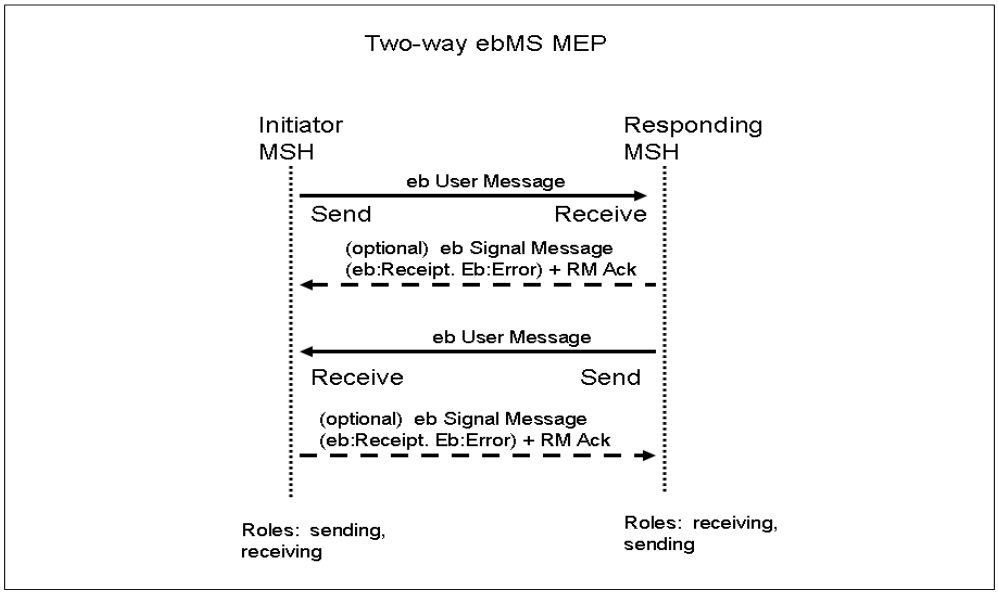
NOTE: The ebMS operation "Notify" is not bound to a business action, like Notifications often are in RosettaNet. Instead, they fulfill a QoS contract between the messaging layer and the application layer, about the reporting of errors.

Two basic MEPs are supported in ebMS 3.0.:

- **One-way MEP**: involves the transfer of a single business message (or action message), followed by some optional message(s) flowing in the other direction, that provide some status about the initial message.



- **Two-way MEP**: involves the transfer of a business (or action) message in one direction, followed by the transfer of another business message in the opposite direction (the "response"). Each one of these business messages may be followed by some optional message(s) flowing in the other direction, that provide some status about it. The following figure shows an asynchronous Two-way MEP (synchronous cases are shown in later sections).



.

6.2. Message Types and Terminology

6.2.1. Different Types of ebMS Messages

In the following, a distinction is made between ebMS signal messages and RosettaNet signal messages.

- **ebMS User message**: Such a message has a business payload, and direct significance for the application layer. It is subject to the "submit" and "deliver" operations mentioned in the messaging model, section 6.1. It is subject to ebMS Receipts.
- **ebMS Signal message**: Such a message has no direct significance for the application layer: instead, it is a message that facilitates the exchange or informs on the status of other ebMS messages. Such signals can be "piggy-backed" on other messages as they are in form of SOAP header elements. The following message types are in this category:
  - ebMS Error message
  - ebMS Pull message
  - ebMS Receipt messages (eb:Receipt) are also considered as ebMS signals: Although they do have application relevance, they usually are not intended to be delivered directly to applications the same way a User message is.
- **Accessory signals**: These are SOAP messages that do not have an ebMS header, but assist in some way the transfer of ebMS messages, e.g. from a QoS point of view. Such messages may be supported by external specifications accessory to ebMS, such as Reliable Messaging. Such signals can often be "piggy-backed" on other messages as some of them are in form of SOAP headers (e.g. RM Acks). It is not intended to be visible beyond the ebMS messaging layer (although it may generate some form of notification to the upper layer, e.g. an error notification). The following message types are in this category:
  - Mapping Reliable Messaging protocol messages (management of RM sequences, etc)
  - Reliable Messaging acknowledgements. That is for the exclusive usage of the ebMS protocol. It would generally be at lower level than message choreographies defined by RosettaNet PIPs.
  - SOAP Faults or HTTP errors (with error status codes)

6.2.2. Mapping of RosettaNet messages to ebMS Messages

The following rules define how RosettaNet messages map to ebMS Messages:

- **A RosettaNet Action message** is always mapped to an ebMS User message: These are the messages to be consumed by applications, with a rich ebMS "business header" (in eb:Messaging element) the profiling of which is defined in section 4.
- **A RosettaNet signal message** (receipt, exception) is, from an ebMS perspective, closer to application than to ebMS signaling. It maps to an ebMS User message too. ebMS V2 does not handle these any differently from action messages. The signal content is treated in ebMS V2 as any other application payload.
  - NOTE: In some cases where non-repudiation is not required or does not involve payload validation (see 6.3), the ebMS V3 Receipt - which is an ebMS Signal message (as opposed to an ebMS User message) – will be used instead

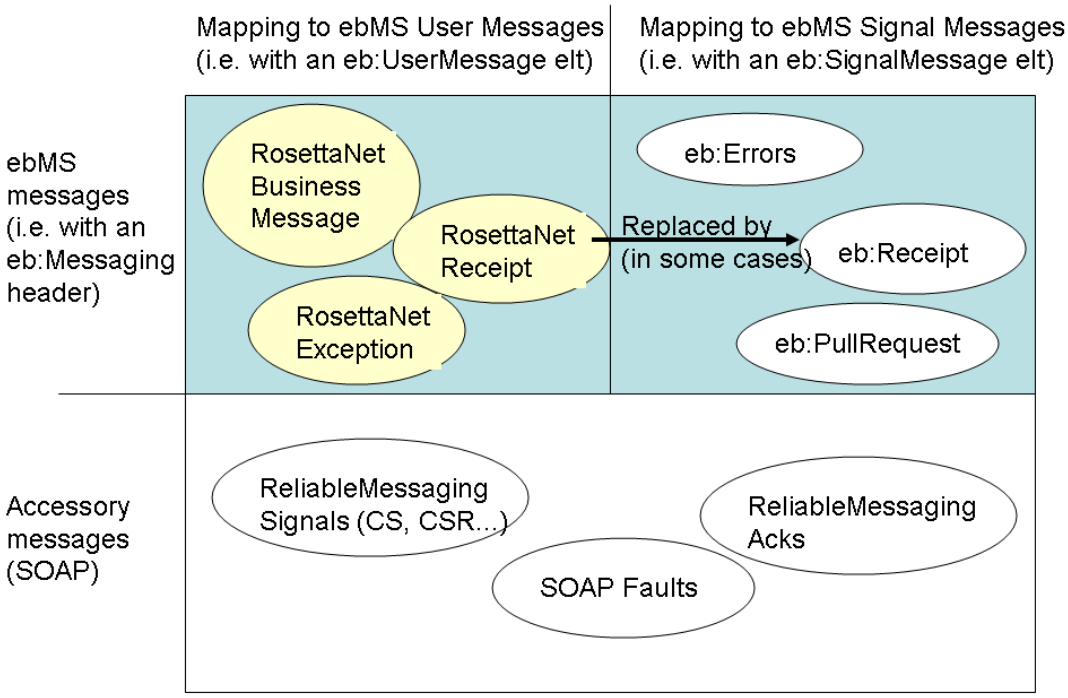
of the RosettaNet Receipt. In other cases when Reliable Messaging is used, RM Acks will play a similar role.

- It is possible to piggyback ebMS signal messages or accessory message (acknowledgements, errors, Receipt) on ebMS User messages, merging into one message what could have been considered as two separate messages.

The following figure shows how RosettaNet message types map to ebMS V3. They all map to ebMS User Messages, except in some cases the RosettaNet Receipt Acknowledgement is substituted by the ebSMS Receipt signal (eb:Receipt).

- White-colored ovals show all types of messages involved in ebMS exchanges, that do not have RosettaNet (RNIF) counterparts but assist the exchange in various functions.
- Peach-colored ovals show RosettaNet messages and how they are categorized in terms of ebMS messages.

RosettaNet Messages and ebMS Messages



6.3. Receipt Semantics: Simple and Validating Non-Repudiation

We consider here the basic message sequence of sending an Action message, and getting back a Receipt Acknowledgement or an Exception. As ebMS V3 provides both (a) a Reliable Messaging (RM) feature, and (b) a Receipt signal message, it appears that in most cases these ebMS features make the RosettaNet Receipt Acknowledgement redundant:

- A retry mechanism is specified in RN, triggered by not receiving a Receipt Acknowledgement in time. It is controlled with a RetryCount parameter, and Time to Acknowledge.
- ebMS 3.0 has a similar mechanism of retries, until an ebMS acknowledgement (Reliable Messaging - or RM - Ack) is received. A maximum number of retries as well as a retry interval, are specified.

The mapping of the parameters of RM features to RosettaNet equivalent has been described in Appendix A.

The RN Retry and Acknowledgement mechanisms can be largely supported by ebMS 3.0 Reliable Messaging mechanism, except for two aspects:

- **Document Validation**: The meaning of an RN Receipt Acknowledgement usually goes beyond message reception, to include document validation (grammar level). The ebMS acknowledgement does not have this semantics.
- **Non-repudiation of Receipt**. Receipt Acknowledgements are used for non-repudiation of receipt. ebMS RM acknowledgements cannot be used for this purpose.

On these same functions, the ebMS Receipt signal compares as follows with the RN Receipt Acknowledgement:

- **Document Validation**: The ebMS Receipt signal is generally sent back by the MSH before payload validation occurs. It cannot be counted on to implement this semantics.
- **Non-repudiation of Receipt**. The (signed) ebMS Receipt signal can be used for this purpose and replace here the RN Receipt Acknowledgement.

In case non-repudiation of receipt is required, the signed ebMS Receipt signal (eb:Receipt) MUST be used – whether or not RM is also used. This ebMS Receipt signal must include a digest of the original message. It MUST include the ebbpsig:NonRepudiationInformation child element, as defined in the ebBP Signal Schema [ebBP-SIG].

Two variants of non-repudiation of receipts are supported in this profile:

- **Simple non-repudiation**: In this variant, the signed eb:Receipt is sent back before document validation occurs. The eb:Receipt only means that the message has been well received and that the receiving endpoint is taking responsibility for further processing (including payload validation).
- **Validating non-repudiation**: In this variant, the signed eb:Receipt is sent back only after the document validation occurs. The eb:Receipt means that the message has been well received and that it is considered as valid for further business processing.

The recommended profiling is as follows:

When non-repudiation of receipt is not required	Do NOT use the RosettaNet Receipt Acknowledgement (positive) signals. Instead, use the ebMS3 Receipt signal message. In that case, the ebbpsig:NonRepudiationInformation MAY be absent. No other element than ebbpsig:NonRepudiationInformation is allowed by this profile as child of eb:Receipt. If this element is not used, then eb:Receipt MUST be empty. NOTE: In case Reliable Messaging is used (generating RM Acks and automatic resends), the use of ebMS3 Receipt signal is optional, as the RM Ack will provide reception awareness. **In case of invalid payload**: only then would an RosettaNet exception message (type: Receipt Acknowledgement Exception) be sent back, as the result of a validation check occurring at higher level than the messaging layer. The sending of this exception SHOULD use Reliable Messaging supported by ebMS3. It MUST be sent as a regular ebMS User Message. The absence of such a signal tells the sender that the payload was valid. Note: a 0A1 PIP could be used too.

When non-repudiation of receipt is required

Often there are several steps in a non-repudiation mechanism (or layers). Validation of the payload may not belong to the initial step. Also, it appears that different users give different meaning to non-repudiation, e.g. regarding the degree of payload validation. For these reasons, two options are available to users, depending on the precise semantics of non-repudiation that is required.

Option 1: **Simple non-repudiation**: Payload validation is NOT a precondition to sending the receipt. In that case, an ebMS Receipt signal is sent back (eb:SignalMessage/eb:Receipt header element), including a digest of the received payload. The value of eb:MessageInfo/eb:RefToMessageId MUST refer to the message for which this signal is a receipt. Reliable Messaging may or may not be used.

Option 2: **Validating non-repudiation**: Payload validation is a precondition to sending the receipt. In that case, an RN Receipt Acknowledgement will be sent back as a regular ebMS User Message, bundled with the eb:SignalMessage/eb:Receipt header element that includes a digest of the received valid payload. The value of eb:MessageInfo/eb:RefToMessageId MUST refer to the message for which this signal is a receipt. Reliable Messaging may or may not be used.

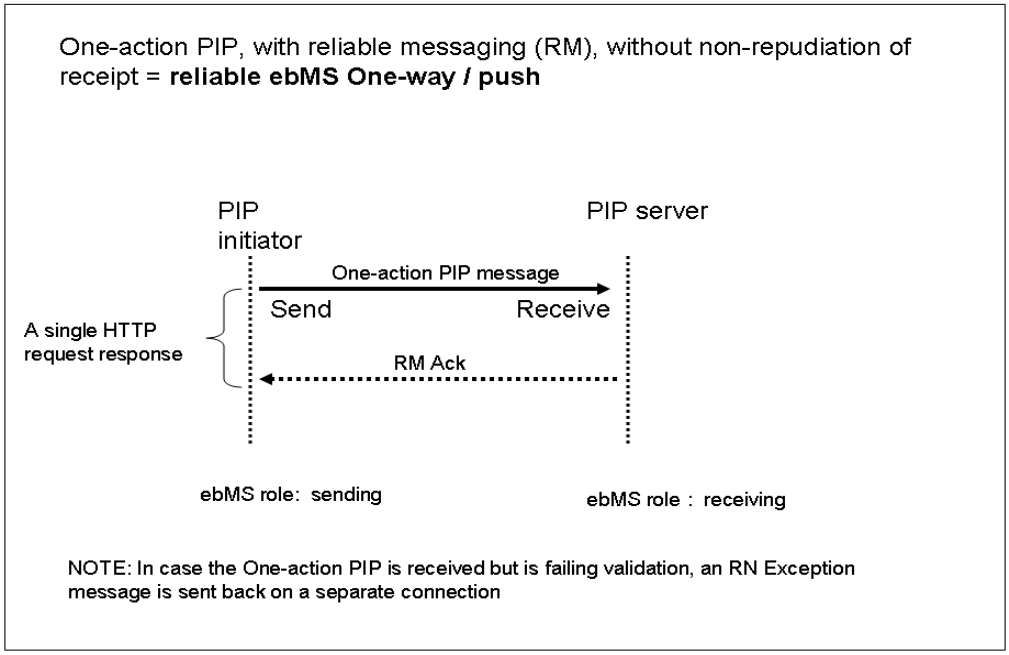
6.4. IT Scenario: One-Action PIP from Server to Server

The business requirement is for one business partner to send another a RosettaNet Business Message, and in return it receives a confirmation of receipt (RN Receipt Acknowledgement or other signal, depending on non-repudiation requirements) or an RN Exception for it. Both partners have “Server” capability, i.e. are able to receive incoming requests.

The choreography variants are:

6.4.1. One-action PIP without Non-Repudiation of Receipt

In this variant, Reliable Messaging is used and no receipt message (RosettaNet format or ebMS format) is expected. The Reliable Messaging acknowledgement is expected on the same HTTP connection (back-channel).



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant):

General PMode parameters:

- **PMode.MEP: http://www.oasis-open.org/committees/ebxml-msg/one-way**
- **PMode.MEPbinding: http://www.oasis-open.org/committees/ebxml-msg/push**

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: true.**
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer: true**

PMode[1].Reliability:

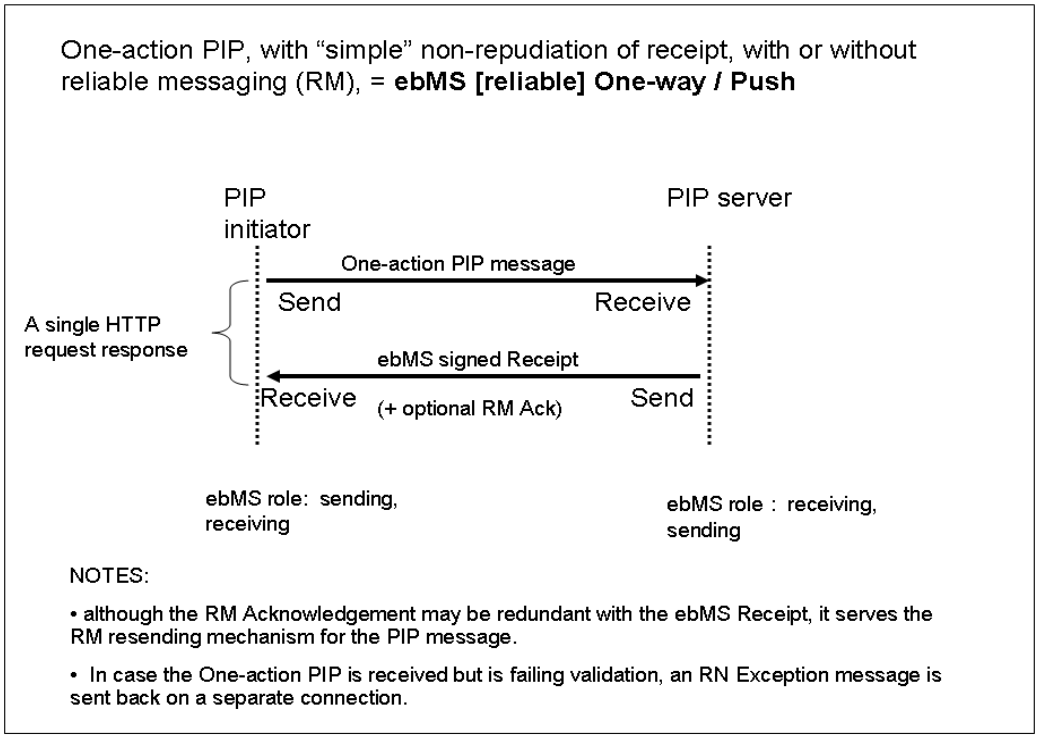
- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern: Response**

PMode[1].Security:

- **PMode[1].Security.SendReceipt: false**

6.4.2. One-action PIP, with Simple Non-Repudiation of Receipt

In this variant, a signed ebMS receipt message (but no RosettaNet Receipt Acknowledgement) is expected. Non-repudiation is required, with “simple” semantics (sent before the message payload is validated). Reliable Messaging is optional, mostly for providing automatic message resending capability.



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant):

General PMode parameters:

- **PMode.MEP: http://www.oasis-open.org/committees/ebxml-msg/one-way**
- **PMode.MEPbinding: http://www.oasis-open.org/committees/ebxml-msg/push**

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: true.**
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer: true**. (in case RM is used)

PMode[1].Reliability: (if used)

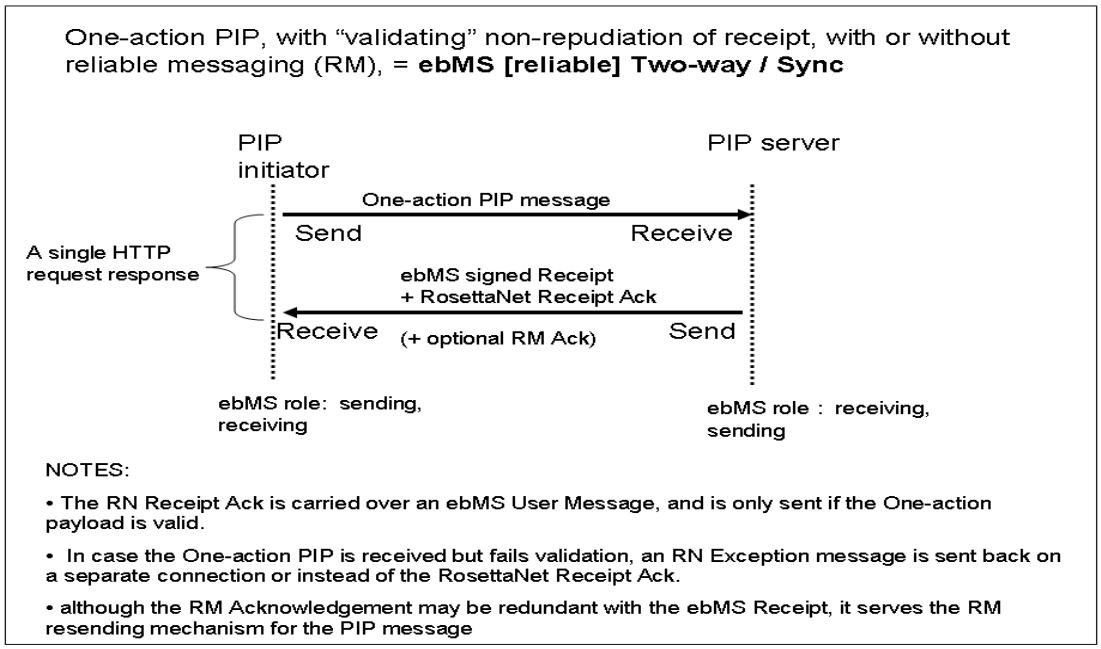
- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern: Response**

PMode[1].Security:

- **PMode[1].Security.SendReceipt:** true
- **PMode[1].Security.SendReceipt.ReplyPattern**: response

6.4.3. One-action PIP, with Validating Non-Repudiation of Receipt

In this variant, both a signed ebMS receipt message and a RosettaNet Receipt Acknowledgement are expected. Non-repudiation is required, with "validating" semantics (sent after the message payload is validated). The MSH on the PIP Server side is waiting for submission of the RN Receipt Acknowledgement to be submitted before sending back the ebMS Receipt header, both piggy-backed on the HTTP response. Reliable Messaging is optional, mostly for providing automatic message resending capability.



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant):

General PMode parameters:

- **PMode.MEP: http://www.oasis-open.org/committees/ebxml-msg/two-way**
- **PMode.MEPbinding: http://www.oasis-open.org/committees/ebxml-msg/sync**

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: true.**
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer: true**. (in case RM is used)

PMode[1].Reliability: (if used)

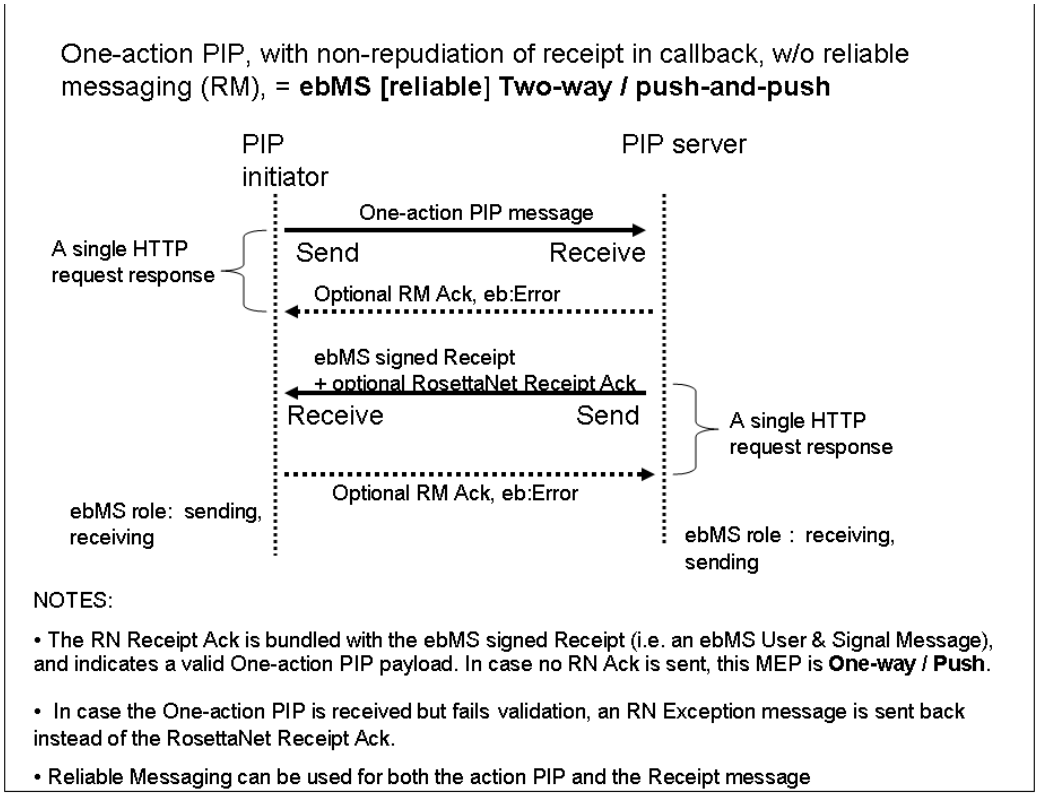
- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern: Response**

PMode[1].Security:

- **PMode[1].Security.SendReceipt: true**
- **Pmode[1].Security.SendReceipt.ReplyPattern: response**

6.4.4. One-action PIP, with Callback Receipt for Non-Repudiation

In this variant, the signed ebMS receipt message (and optionally a RosettaNet Receipt Acknowledgement in case of deep non-repudiation) is sent back as a "callback" on a separate connection. The MSH on the PIP Server side is waiting for submission of the RN Receipt Acknowledgement before sending back the ebMS Receipt header, both piggy-backed on the HTTP request. Reliable Messaging is optional, mostly for providing automatic message resending capability.



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant):

General PMode parameters:

- **PMode.MEP:**
  http://www.oasis-open.org/committees/ebxml-msg/two-way (if RosettaNet Receipt Acknowledgement is expected) or:
  http://www.oasis-open.org/committees/ebxml-msg/one-way
- **PMode.MEPbinding:**
  http://www.oasis-open.org/committees/ebxml-msg/push-and-push (if RosettaNet Receipt Acknowledgement is expected) or :
  http://www.oasis-open.org/committees/ebxml-msg/push

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: true.**
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer: true**. (in case RM is used)

PMode[1].Reliability: (if used)

- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern: Response**

PMode[1].Security:

- **PMode[1].Security.SendReceipt: true**
- **PMode[1].Security.SendReceipt.ReplyPattern**: **callback**

6.5. IT Scenario: One-Action PIP from Pure Client to Server

The business requirement is same as in 6.4 (Server to Server). But the PIP Initiator cannot act as a Server, i.e. cannot receive incoming requests. It is a "Pure Client", meaning it either does not have a static IP address, or has connectivity restrictions preventing others to connect to it.

The choreography variants are:

6.5.1. One-action PIP without Non-Repudiation of Receipt

In this variant, Reliable Messaging is used and no receipt message (RosettaNet format or ebMS format) is expected. The Reliable Messaging acknowledgement is expected on the same HTTP connection (back-channel).

The choreography is same as defined in section 6.4.1.

6.5.2. One-action PIP, with Simple Non-Repudiation of Receipt

In this variant, a signed ebMS receipt message (but no RosettaNet Receipt Acknowledgement) is expected. Non-repudiation is required, with "simple" semantics (sent before the message payload is validated). Reliable Messaging is optional, mostly for providing automatic message resending capability.

The choreography is same as defined in section 6.4.2.

6.5.3. One-action PIP, with Validating Non-Repudiation of Receipt

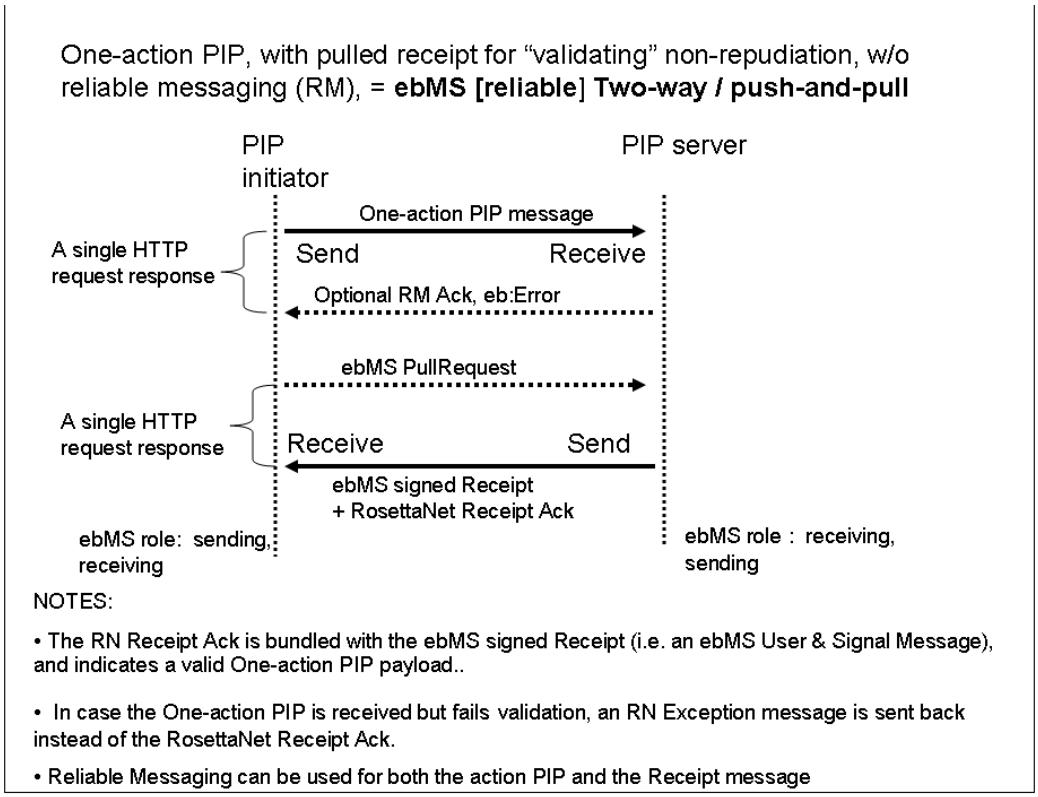
In this variant, both a signed ebMS receipt message and a RosettaNet Receipt Acknowledgement are expected. Non-repudiation is required, with "validating" semantics (sent after the message payload is validated). The MSH on the PIP Server side is waiting for submission of the RN Receipt Acknowledgement to be submitted before sending back the ebMS Receipt header, both piggy-backed on the HTTP response. Reliable Messaging is optional, mostly for providing automatic message resending capability.

The choreography is same as defined in section 6.4.3.

6.5.4. One-action PIP, with Pulled Receipt for Validating Non-Repudiation

This case typically applies when the validation time for the payload document prohibits a synchronous response as in 6.5.3.

The MSH on the Pure Client side is sending PullRequest signals until the PIP Server side sends back the RN Receipt Acknowledgement along with the ebMS Receipt header, both piggy-backed on the HTTP response. Reliable Messaging is optional, mostly for providing automatic message resending capability



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant):

General PMode parameters:

- **PMode.MEP:**
  http://www.oasis-open.org/committees/ebxml-msg/two-way
- **PMode.MEPbinding:**
  http://www.oasis-open.org/committees/ebxml-msg/push-and-pull

PMode[2].BusinessInfo:

- **PMode[2].BusinessInfo.MPC:** this parameter must specify a Message Partition Channel where Receipts will be assigned for pulling.

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: true.**
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer: true**. (in case RM is used)

PMode[1].Reliability: (if used)

- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern: Response**

PMode[1].Security:

- **PMode[1].Security.SendReceipt: true**
- **PMode[1].Security.SendReceipt.ReplyPattern: pulled**

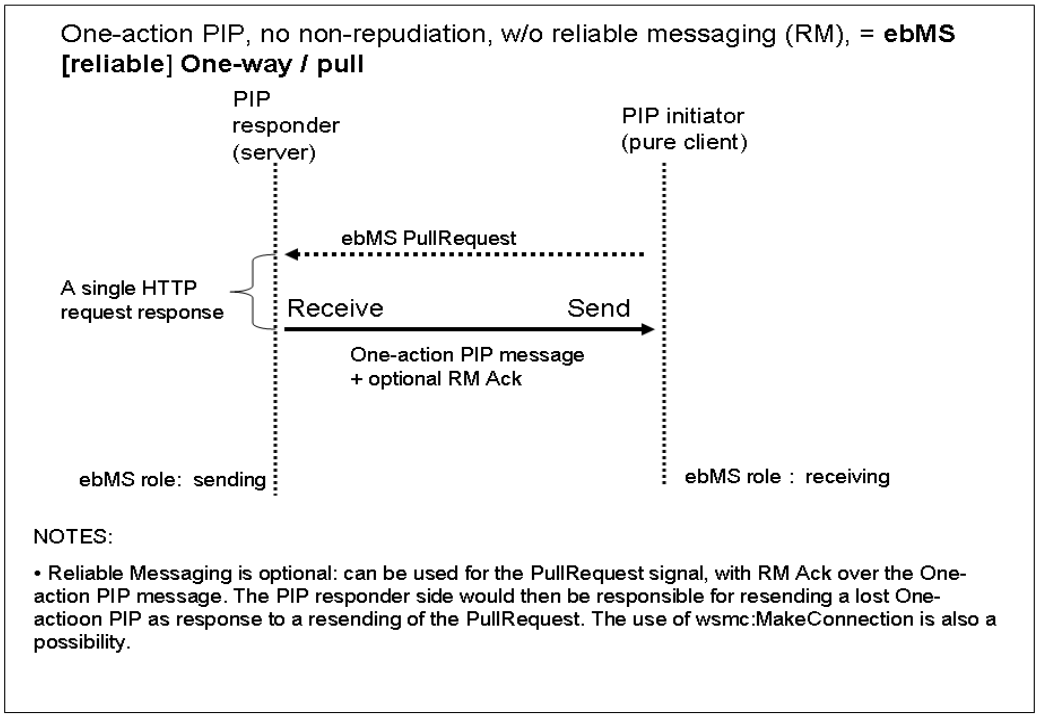
6.6. IT Scenario: One-Action PIP from Server to Pure Client

The business requirement is same as in 6.4 (Server to Server) and 6.5 (Client to Server). But the sender of the One-action PIP cannot initiate the transfer because its partner is a Pure Client. In that case, the action PIP receiver (Pure Client) is the PIP Initiator.

The choreography variants are:

6.6.1. One-action PIP without Non-Repudiation of Receipt

In this variant, no receipt message (RosettaNet format or ebMS format) is expected. Reliable Messaging is used for the PullRequest signal, with acknowledgement expected on the same HTTP connection (back-channel). Optionally an RM Ack is sent for the Action PIP, on a separate connection (e.g. bundled with the next PullRequest).



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant).

General PMode parameters:

- **PMode.MEP:**
  http://www.oasis-open.org/committees/ebxml-msg/one-way
- **PMode.MEPbinding:**
  http://www.oasis-open.org/committees/ebxml-msg/pull

PMode[1].BusinessInfo:

- **PMode[1].BusinessInfo.MPC:** this parameter must specify a Message Partition Channel where the Action PIP message will be assigned for pulling.

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: false.** (errors regarding the Action PIP must be sent as callback)

PMode[1].Reliability: (if used. This applies to the PullRequest signal as well)

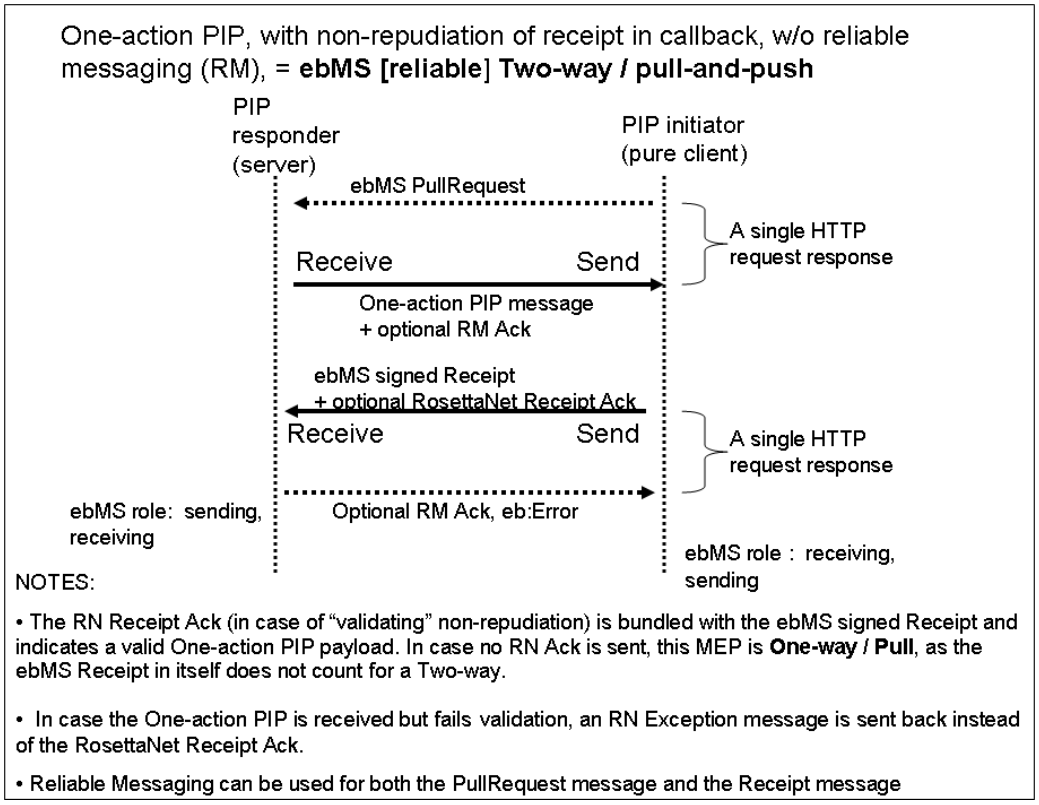
- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern:** Response (this concerns the PullRequest message, N/A for the pulled Action PIP message.)
- **PMode[1].Reliability.AtLeastOnce. Contract.AckResponse: False** (in case no RM Ack is expected for the Action PIP sent over the HTTP Response).

PMode[1].Security:

- **PMode[1].Security.SendReceipt: false**

6.6.2. One-action PIP, with Non-Repudiation of Receipt

In this variant, non-repudiation is required. A signed ebMS receipt message is sent in callback mode, with an optional RosettaNet Receipt Acknowledgement (in case of “validating” non-repudiation). Reliable Messaging is optional, mostly for providing automatic message resending capability.



The PMode parameters that control this variant are (in addition to other PMode parameters common to all One-action PIP variants and not specific to this variant).

General PMode parameters:

- **PMode.MEP: http://www.oasis-open.org/committees/ebxml-msg/two-way**. (in case an RN Ack Receipt is expected. Otherwise this MEP would be a One-way/ Pull).
- **PMode.MEPbinding: http://www.oasis-open.org/committees/ebxml-msg/pull-and-push**

PMode[1].BusinessInfo:

- **PMode[1].BusinessInfo.MPC**: this parameter must specify a Message Partition Channel where the Action PIP message will be assigned for pulling.

PMode[1].ErrorHandling:

- **PMode[1].ErrorHandling.Report.AsResponse: false.**
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer: true.** (in case RM is used)

PMode[1].Reliability: (if used. These parameters apply as well to the PullRequest message)

- **PMode[1].Reliability.AtLeastOnce.Contract: true**
- **PMode[1].Reliability.AtLeastOnce.ReplyPattern: Response** (this concerns the PullRequest message, N/A for the pulled Action PIP message.)

- **PMode[1].Reliability.AtLeastOnce. Contract.AckResponse: False** (in case no RM Ack is expected for the Action PIP sent over the HTTP Response).

PMode[1].Security:

- **PMode[1].Security.SendReceipt: true**
- **PMode[1].Security.SendReceipt.ReplyPattern**: **callback**

7. Quality of Service Policies

7.1. General Security Policies

7.1.1. Signature and Encryption Rules

When using digital signatures or encryption, an MSH implementation conforming to this profile is **REQUIRED** to use the Web Services Security X.509 Certificate Token Profile [WSS11-X509].

This profile **REQUIRES** to use Detached Signatures as defined by the XML Signature Specification [XMLDSIG] when signing ebMS user messages or signal messages. Enveloped Signatures as defined by [XMLDSIG] are not supported by or authorized in this profile.

This profile **REQUIRES** to include the entire eb:Messaging SOAP header block and the SOAP Body in the signature.

This profile **REQUIRES** to use the Attachment-Content-Only transform when building application payloads using SOAP with Attachments [SOAPATTACH]. The Attachment-Complete transform is not supported by this profile.

This profile **REQUIRES** to include the entire eb:Messaging header block and all MIME body parts of included payloads in the signature.

An MSH conforming to this profile **SHALL NOT** encrypt the eb:PartyInfo section of the eb:Messaging header. Other child elements of the eb:Messaging header **MAY** be encrypted or left unencrypted as defined by trading partner agreements or collaboration profiles.

If an user message is to be encrypted and the user-specified payload data is to be packaged in the SOAP Body, MSH implementations are **REQUIRED** to encrypt the SOAP Body.

If an user message is to be encrypted and the user-specified payload data is to be packaged in conformance with the [SOAPATTACH] specification, MSH implementations are **REQUIRED** to encrypt the MIME Body parts of included payloads.

When both signature and encryption are required of the MSH, the message **MUST** be signed prior to being encrypted, as required in ebMS 3 [ebMS3], section 7.6.

7.1.2. Pull Authorization

Message pulling requires authorization in addition to general authentication security, because pulling is targeted to a Message Partition Channel (MPC). Two different MSHs pulling from the same MSH should only be authorized to pull from their dedicated MPC.

A Sending MSH conforming to this profile **MUST** be able to selectively authorize a Receiving MSH that sends a PullRequest in two ways (Options 1 and 2 below), and MAY authorize pulling as Option 3 below:

- **Authorization Option 1**: Use of the WSS security header targeted to the “ebms” actor, as specified in section 7.10 of ebMS V3, with the wsse:UsernameToken profile. This header may either come in addition to the regular wsse security header (XMLDsig for authentication), or may be the sole wsse header, if a transport-level secure protocol such as SSL or TLS is used. An example of message is given in Appendix …

- **Authorization Option 2**: Use of a regular wsse security header (XMLDsig for authentication, use of X509), and no additional wsse security header targeted to "ebms", In that case, the MSH must be able to use the credential present in this security header for Pull authorization, i.e. to associate these with a specific Message Partition Channel (MPC).
- **Authorization Option 3**: In addition to the two previous authorization options an implementation MAY optionally decide to support a third authorization technique, based on transient security (SSL or TLS). SSL/TLS can provide certificate-based client authentication. Once the identity of the Pulling client is established, the Security module may pass this identity to the ebms module, which can then associate it with the right authorization entry, e.g. the set of MPCs this client is allowed to pull from.

This third authorization option – compatible with although not specified in ebMS Core V3 - relies on the ability of the ebms module to obtain the client credentials. This capability represents an (optional) new feature.

7.2. Handling of Receipts

When a Receipt is to be used solely as a reception indicator the sender of the Receipt MAY decide to not insert the ebbpsig:NonRepudiationInformation child element. No other element than ebbpsig:NonRepudiationInformation is allowed as child of eb:Receipt. If this element is not used, then eb:Receipt MUST be empty.

Non Repudiation of Receipt (NRR) requires eb:Receipt signals to be signed, and to contain digests of the original message parts for which NRR is required.

When signed receipts make use of default conventions, the Sending message handler (i.e. sending messages for which signed receipts are expected) MUST identify message parts using Content-Id values in the MIME headers, and MUST sign the SOAP body and all attachments using the **http://docs.oasis-open.org/wss/oasis-wss-SwAProfile-1.1#Attachment-Content-Signature-Transform** within the SignedInfo References list.

As a reminder, the Sending message handler MUST not encrypt any signed content before signing (Section 7.6 in ebMS V3). If using compression in an attachment, the Sending message handler MUST sign the data after compression (see section 3.1). Variations from default conventions can be agreed to bilaterally, but conforming implementations are only required to provide receipts using the default conventions described in this section.

In this profile, when sending a receipt for a message that has been digitally signed the eb:Receipt signal MUST itself be digitally signed, and non-repudiation feature MUST be used: the eb:Receipt element MUST contain the information necessary to provide non-repudiation of receipt of the original message.

NOTE: The digest(s) to be inserted in the ebbp:MessagePartNRInformation element(s) or the Receipt, related to the original message parts for which a receipt is required, may be obtained from the signature information of the original message (ds:SignedInfo element), as only those parts that have been signed are subject to NRR. This means a Receiving message handler may not have to compute digests outside its security module.

7.3. General Reliability Policies

It is RECOMMENDED to use the WS-RM contract AtMostOnce whenever AtLeastOnce is used, so that duplicates generated by the resending mechanism can be eliminated.

8. Deployment Configurations and MSH Requirements

8.1. Connecting to Web Services

When PIP back-end processing is done by Web services deployed behind the firewall, it is RECOMMENDED to deploy the MSH as a Gateway (e.g. in the DMZ). This gateway will convert back and forth ebMS messages into Web service invocations (or responses). Because of the Web-service compliant nature of ebMS V3 messages it is often sufficient to remove the eb:Messaging header before forwarding the message to the appropriate document-style Web service.

8.2. Required V3 Conformance Profiles

The following ebMS V3 Conformance Profiles are RECOMMENDED (see **[ebMS3-CP]**):

(1) Case of a pure-client message endpoint: **light-handler (LH-RM)** Conformance Profile
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/cprofiles/200707/lighthandler-rm

(2) Case of a server-enabled message endpoint: **Gateway RX V3** Conformance Profile.

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/cprofiles/200707/gateway-rxv3

9. Appendix A: CPA Profiling and Sample

9.1. CPA Profiling Forms

The profiling and definition of CPA data (both instance and profile template) can be facilitated using a set of forms, such as those provided by the ebXML Implementation, Interoperability and Conformance (IIC) OASIS Committee. It is recommended for business partners to use the "Deployment Profile Template for CPPA V2.0" published by the OASIS IIC, in order to finalize their collaboration agreements. A subset of these forms is presented here.

Each element (or entry) in each one of these forms maps to a CPA element. Either the name of the entry is explicit enough to refer to the corresponding CPA element, or the name of the corresponding CPA element is mentioned in clear, usually prefixed with the qualifier "tp:" (e.g. tp:channelID).

- When entries in these forms must map to some PIP definition elements, it is indicated in the form entry.
- When entries in these forms are left to the user to instantiate as s/he wants to, the entry value is left empty (or just referring to the actual name of the CPA element, e.g. tp:TransportID)

A sample CPA document is listed in the next sub-section.

NOTE: These forms and their content are based on CPPA V2.1, which is very close to V2.0 (includes an errata from V2.0 and has additional extensibility points - some element names may be different. Please refer to the Errata for V2.0.) Once CPPA V3 is complete, it is expected that it will become the primary target for this profiling. Profiling CPPA V3 can largely reuse the profiling done for CPPA 2.1.

9.2. Profiling the CPA Artifact Names and References

This form is used to identify the CPA profile, and also any CPA instance that is derived from a profile. It recommends some naming conventions for the CPA artifacts.

CPA Profile Info

CPA Profile Info	**Name**	[Provide a name for the Collaboration Protocol Agreement profile. The name should identify when applicable: (a) the version of CPA, (b) the community sharing this profile (here, RN), (c) type of artifact (here a profile), (d) name of profile, (e) party ID if this profile is attached to a party.] Recommended: "CPA2.1-RN-Profile-"<profileID>"-"<partner1>" Examples: CPA2.1-RN-Profile-PIP3A4-222222 CPA2.1-RN-Profile-TP31-222222
	File name	[Provide a file name for the Collaboration Protocol Agreement profile file.] "CPA2.1-RN-Profile-"<profileID>"-"<partner1>"-file" (followed by appropriate suffix – e.g. .xml for the XML definition.)

		Examples: CPA2.1-RN-Profile-PIP3A4-222222-file.pdf CPA2.1-RN-Profile-TP31-222222-file.xml
CPA Instance Info	**Name**	[Define the name format for the CPA instances resulting from using this profile. The name should identify when applicable: (a) the version of CPA, (b) the community sharing this profile, (c) name of profile, (d) ID of instance, (e) party IDs.] Recommended: "CPA2.1-RN-"<profileID>"-"<instID>"-"<partner1-partner2> Example: CPA2.1-RN -P15-001-222222-333333 CPA2.1-RN -TP2-004-222222-333333
	File name	[Define the file name format for a Collaboration Protocol Agreement instance.] Recommended: "CPA2.1-RN-"< profileID >"-"<instID>"-"<partner1-partner2>"-file" (followed by appropriate suffix – e.g. .xml for the XML definition.) Example: CPA2.1-RN-P15-001-222222-333333-file.pdf CPA2.1-RN-TP2-004-222222-333333-file.xml
	CPA Id	[Define the format of the CPA Id. Must align with CPAId in message header.] Recommended: same as CPA name, i.e.: "CPA2.1-RN-"< profileID >"-"<instID>"-"<partner1-partner2>
	Lifetime of CPA	**Start**: [The starting date and time of the agreement.] **End**: [The end date and time of the agreement. The start and end date/times define the duration that the agreement is in effect.]
	Context of application	**ConversationLimit**: [NONE or numeric value. The agreement is terminated (no longer valid) when the conversation limit is reached.] **Concurrent Conversation Limit**: [NONE or numeric value. The maximum number of conversations that can be in process at the same time. Provide this value when there are constraints that limit the number of business transactions that one or more of the parties can process simultaneously.]

9.3. Profiling the Party Info

This form is used to identify the parties involve. A CPA profile will typically contain one of these fully instantiated. At least another one of these will need to be filled by another business partner in order to produce a complete CPA instance.

Profiling (alignment with data or QoS in RosettaNet PIPs, or with ebMS header data that is itselfprofiled) is required for some entries of this table. The rest of this table is provided as a support for users.

Party Info		
CPA Reference	[CPA Profile name] [CPA Instance name, if used for instantiating a particular CPA]	
Party element	**PartyId**	[The formal unique identifier for the organization. Must align with eb:PartyId in message header (section 4)] All Party ID elements present in CPA must appear in the message header**.**
	Type	[Must align with eb:PartyId/@type in message header (section 4)]
	Reference	[A URL or URI that points to a location (e.g. web page or directory) where more information can be found on the party.]
Collaboration Roles elements	[List the collaboration role names that this party is expected to fulfill. The role names need to be unique within this list. Each role will be detailed in a CollaborationRole form.]	
	CollaborationRole 1	Process Name [maps to eb:Service I header] Role Name [maps to eb:Role in header]
	CollaborationRole 2	Process Name [maps to eb:Service I header] Role Name [maps to eb:Role in header]
	(others?)	(there may be additional roles)
Certificates elements	[List the certificates info and ID.]	
	Certificate 1	
	Certificate 2	
	(others?)	(there may be additional certificates)
DeliveryChannels Elements	[Describes a *Party's Message*-receiving and *Message*-sending characteristics. It consists of one document-exchange definition and one transport definition. The details of each DeliveryChannel element will be specified in a different form.]	

	DeliveryChannel 1	[give only the tp:channelId]
	DeliveryChannel 2	[give only the tp:channelId]
	(others?)	
Transports Elements	**Transport ID**	[tp:TransportId]
Documents Exchanges	**Exchange ID**	[tp:docExchangeId]

9.4. Profiling the Collaboration Roles

This form is used to identify the roles in which a party may be acting under this CPA or CPA profile. One form will be filled for each role.

Profiling (alignment with data or QoS in RosettaNet PIPs) is required for some entries of this table. The rest of this table is provided as a support for users.

ColaborationRole Info

CPA Reference	[CPA Profile name]	
	[CPA Instance name, if used for instantiating a particular CPA]	
Role Identification	**Name**	[maps to eb:Role] **Ref-1** in Tables 7,10 (see section 9.8)
	Type	[xlink:type], e.g. “simple”
	Href	[xlink:href] Example: xlink:href="http://www.rosettanet.org/processes/3A4.xml#Buyer">
Application Certificate	**ID:**	
	Comments:	
Process Specification	**name**	[The name of the business process specification that this role applies to] maps to ProcessSpecification nameID attribute in ebBP guideline (e.g. **urn:rosettanet:specification:interchange:PIP3A4:xml:ebbp:v11_00),** i.e. to eb:Service (see Section 4 Message Description) xlink:href : contains a reference to the ebBP definition (e.g. =http://www.rosettanet.org/processes/3A4.xml)

	Version	[Version of the business process specification]	
	Type		
	Uuid / nameId	tradingpartner uuid → attribute uuid of ebBP definition when present (attr in process specification top element) (Example= "urn:icann:rosettanet.org:bpid:3A4$2.0")	
Service Binding item (One for every Action or Signal message)	**Associated Service name**	[tp:ServiceBinding/tp:Service] Maps to eb:Service (see Section 4, Message Description) Example: <tp:Service>**urn:rosettanet:specification:interchange:PIP3A4:xml:ebbp:d11_00**</tp:Service>	
	Action direction	[send/ receive]	
	Action Binding	[tp:id] example: companyA_ABID1 (to be used for further references. Unique) [tp:action] example: "Purchase Order Request Action" maps to eb:Action (see Section 4, Message Description)(e.g. ="PurchaseOrderRequestAction") [tp:packageId] Example: tp:packageId="CompanyA_RequestPackage". Refers to MIME structure of payload.	
	Business Transaction Characteristics	tp:isNonRepudiationRequired	maps to "Non-Repudiation of Origin and Content", column 8 in PIP definition tables below. (="true" in below example) **Ref-8** in Tables 7,10 (see section 9.8)
		tp:isNonRepudiationReceiptRequired	maps to "Non-Repudiation Required" column 3 in PIP tables below. (="true" in below example) **Ref-3** in Tables 7,10 (see section 9.8)
		tp:isConfidential	(using SSL or digital envelope) **Ref-9** in Tables 7,10 (see section 9.8)
		tp:isAuthenticated	NOTE: should map to DTD related docs
		tp:isTamperProof	(NOTE: authenticated gives integrity)
		tp:isAuthorizationRequired	maps to "Is Authorization Required" column 7 in PIP tables below. (="true" in below example)

		Ref-7 in Tables 7,10 (see section 9.8)
	tp:timeToAcknowledgeReceipt	maps to “Time to Acknowledge“ column 4 in PIP tables below. (="PT2H" in below example) NOTE: it should be equivalent to (retryInterval * Retries) in ebMS.
		Ref-4 in Tables 7,10 (see section 9.8)
	tp:timeToPerform	maps to “Time to Perform“ column 5 in PIP tables below (not really captured in CPA, about same as time to Ack)
		Ref-5 in Tables 7,10 (see section 9.8)
	tp: isIntelligibleCheckRequired	(no map to PIP attributes)
	tp: timeToAcknowledgeAcceptance	(no map to PIP attributes)
	Tp: ***retryCount***	Must NOT be used. Instead, the Retries element of the Reliable Messaging CPA element will map to “Retry Count“ column in PIP tables below:
		Ref-6 in Tables 7,10 (see section 9.8)

9.5. Profiling the Delivery Channels

Delivery Channels - A delivery channel describes a *Party's Message*-receiving and *Message*-sending characteristics. It consists of one document-exchange definition and one transport definition.

No profiling is required for this data. This table is provided as a support for users.

Delivery Channel Info		
CPA Reference	[CPA Profile name]	
	[CPA Instance name, if used for	instantiating a particular CPA]
Identity and Components	**channelId**	
	transportId	
	docExchangeId	
Messaging Characteristics		
	ackRequested	Reliable Messaging parameter for Guaranteed Delivery (At Least Once)

ackSignatureRequested	NOTE: this is a way to support a form of non-repudiation of Receipt, that is generally not sufficient for RosettaNet.
duplicateElimination	Reliable Messaging parameter for No Duplicate Delivery (At Most Once)
Actor	

9.6. Profiling the Document Exchanges

Document Exchange - The Document-exchange layer specifies processing of the business documents by the Message-exchange function. Properties specified include encryption, digital signature, and reliable-messaging characteristics. The options selected for the Document-exchange layer are complementary to those selected for the transport layer. For example, if Message security is desired and the selected transport protocol does not provide *Message* encryption, then *Message* encryption must be specified in the Document-exchange layer.

Profiling (alignment with data or QoS in RosettaNet PIPs) is required for some entries of this table The rest of this table is provided as a support for users.

Document Exchange Info

CPA Reference		[CPA Template name] [CPA Instance name, if used for instantiating a particular CPA]
Doc Exchange ID		[tp:docExchangeId]
Sender Binding	**Reliable Messaging**	[tp:ReliableMessaging] - tp:Retries: [maps to "Retry Count" column 6 in above tables.] - tp:RetryInterval: [Example: <tp:RetryInterval>PT2H</tp:RetryInterval>] - tp:MessageOrderSemantics: [Example: "Guaranteed"]
	Persist Duration	[tp:PersistDuration]
	Non Repudiation of Origin	[tp:SenderNonRepudiation] - tp:NonRepudiationProtocol - tp:HashFunction - tp:SignatureAlgorithm - tp:SigningCertificateRef
	Digital Envelope	[tp:SenderDigitalEnvelope] - tp:DigitalEnvelopeProtocol - tp:EncryptionAlgorithm - tp:EncryptionSecurityDetailsRef
	Nemespaces	[tp:NamespaceSupported]
Receiver Binding		
	Reliable Messaging	[tp:ReliableMessaging] - tp:Retries - tp:RetryInterval - tp:MessageOrderSemantics
	Persist Duration	[tp:PersistDuration]

Non Repudiation of Receipt	[tp:ReceiverNonRepudiation] - tp:NonRepudiationProtocol - tp:HashFunction - tp:SignatureAlgorithm - tp:SigningSecurityDetailsRef
Digital Envelope	[tp:ReceiverDigitalEnvelope] - tp:DigitalEnvelopeProtocol - tp:EncryptionAlgorithm - tp:EncryptionCertificateRef
Nemespaces	[tp:NamespaceSupported]

9.7. Profiling the Transport Protocol

The transport layer identifies the transport protocol to be used in sending messages through the network and defines the endpoint addresses, along with various other properties of the transport protocol. Choices of properties in the transport layer are complementary to those in the document-exchange layer (see "Document-Exchange Layer" directly above.)

No profiling is required for this data. This table is provided as a support for users.

Transport Info		
CPA Reference	[CPA Template name] [CPA Instance name, if used for instantiating a particular CPA]	
Transport Sender	**Protocol**	[tp: TransportProtocol]
	Client security	[tp:TransportSecurityProtocol] [tp:ClientCertificateRef]
Transport Receiver	**Protocol**	[tp: TransportProtocol]
	End Point	[tp:Endpoint/@uri, tp:Endpoint/@type]
	Server security	[tp:TransportSecurityProtocol] [tp:ServerCertificateRef] [tp:ClientSecurityDetailsRef]

9.8. Examples of Tables Used in PIP Definitions

These tables are extracted from the PIP7C7 definition. Their purpose here is to illustrate the terms and properties that map to the concepts in above CPA forms. The last row in these tables has been added to identify columns that are referred to (Ref-n) in the above CPA forms.

Table 7: Business Activity Performance Controls

Role Name	Activity Name	Acknowledgment of Receipt		Time to Perform	Retry Count	Is Authorization Required?	Non-Repudiation of Origin and Content?
		Non-Repudiation Required?	Time to Acknowledge				
Foundry or Test Services	Notify of Semiconductor Test Data	Y	2 hrs	N/A	3	Y	Y
Ref-1	Ref-2	Ref-3	Ref-4	Ref-5	Ref-6	Ref-7	Ref-8

Table 10: Message Exchange Controls

#	Name	Time to Acknowledge	Time to Respond to Action	Included in Time to Perform	Is Authorization Required?	Is Non-Repudiation Required?	Is Secure Transport Required?
1.	Semiconductor Test Data Notification Action	2 hrs	N/A	N/A	Y	Y	Y
1.1.	Receipt Acknowledgment	N/A	N/A	N/A	N	N	Y
		Ref-4	Ref-10		Ref-7	Ref-3, Ref-8	Ref-9

10. Appendix B: Glossary

AMD	Abstract Message Service
ATPA	Abstract Trading Partner Agreement
MEP	Message Exchange Pattern
RNIF	RosettaNet™ Implementation Framework
PIP	(RosettaNet terminology): Partner Interface Process
TP	Trading Profile
ebMS	ebXML Messaging Services specification (an ebXML standard)
BPSS	ebXML Business Process Specification Schema (an ebXML standard)
ebBP	ebXML Business Process specification (applies to new version of BPSS, renamed)
CPP	ebXML Collaboration Protocole Profile (described in CPPA specification, an ebXML standard)
CPA	ebXML Collaboration Protocole Agreement (described in CPPA specification, an ebXML standard)
SBDH	Standard Business Document Header (also known as “Generic Header”)
TPP	Trading Partner Profile

11. References

Source	Description
[AMD]	Title: MMS Abstract Message Definition, Draft 00.07.00, January 7, 2005 RosettaNet Retrived from : http://members.rosettanet.org/dnn_rose/DMX/tabid/2979/DMXModule/624/Command/Core_ViewDetails/Default.aspx?EntryId=346
[ebBP-SIG]	Title: ebXML Business Signals Schema, 2006. OASIS Retrived from: http://docs.oasis-open.org/ebxml-bp/ebbp-signals-2.0
[ebMS2]	Title: ebXML Message Service Specification Version 2.0, April 1, 2002. OASIS Retrived from: http://www.oasis-open.org/committees/ebxml-msg/documents/ebMS_v2_0.pdf
[ebMS3]	Title: ebXML Message Service Specification Version 3.0 Part 1, Core Features, September 30, 2007 OASIS Retrived from: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms_core-3.0-spec.pdf
[ebMS3-CP]	Title: ebXML Messaging Services 3.0 Conformance Profiles, July 25, 2007 OASIS Retrived from: http://www.oasis-open.org/committees/download.php/29854/ebms-3%5B1%5D.0-confprof-cd-03.pdf
[ebCPPA2]	Title: Collaboration-Protocol Profile and Agreement Specification Version 2.0, May 20, 2002. OASIS Retrived from: http://www.oasis-open.org/committees/download.php/202/ebCPP-2_0.pdf
[DPT-ebMS2]	Title: Deployment Profile Template 1.1 for ebMS 2.0, OASIS IIC Committee draft, July 2005. OASIS

	Retrived from: http://www.oasis-open.org/committees/document.php?document_id=21667&wg_abbrev=ebxml-iic
[DPT-CPPA2]	Title: Deployment Profile Template 0.2 for CPPA 2.0, OASIS IIC working draft, August 2005. OASIS Retrived from: http://www.oasis-open.org/committees/document.php?document_id=21667&wg_abbrev=ebxml-iic
[**BPSS-PIP**]	Title: ebXML BPSS Guideline, v1.11, August 2004. RosettaNet Retrived from : http://members.rosettanet.org/dnn_rose/DMX/tabid/2979/DMXModule/624/Command/Core_ViewDetails/Default.aspx?EntryId=5726
[RFC2119]	Author: Scott Bradner Title: Key words for use in RFCs to Indicate Requirement Levels, March 1997. The Internet Engineering Task Force Retrived from: http://www.ietf.org/rfc/rfc2119.txt
[RFC2045]	Author: N. Freed Title: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, 1996. The Internet Engineering Task Force Retrived from: http://www.ietf.org/rfc/rfc2045.txt
[RN-NameSpaces]	Title: RosettaNet Namespace Specification and Management, v1.0, December, 2003. RosettaNet Retrived from : http://members.rosettanet.org/dnn_rose/DMX/tabid/2979/DMXModule/624/Command/Core_ViewDetails/Default.aspx?EntryId=5726
[SOAPATTACH]	Author: John J. Barton, et al. Title: SOAP Messages with Attachments, 2000 W3C Retrived from: http://www.w3.org/TR/SOAP-attachments
[WS-I]	Author: C.Ferris, et al. Title: WS-I Basic Profile (1.2 and 2.0) WS-Interoperability

	Retrived from: hhttp://www.ws-i.org/deliverables/workinggroup.aspx?wg=basicprofile
[WSS11]	Author: Anthony Nadalin, et al. Title: Web Services Security: SOAP Message Security 1.1, June 2005. OASIS Retrived from: http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-pr-SOAPMessageSecurity-01.pdf
[WSS11-X509]	Editor: Anthony Nadalin Title: Web Services Security X.509 Certificate Token Profile 1.1, 2006. OASIS Retrieved from: http://docs.oasis-open.org/wss/v1.1/wss-v1.1-errata-os-x509TokenProfile.pdf (obtained from: http://docs.oasis-open.org/wss/v1.1/)