AS4 Profile of ebMS 3.0 Version 1.0

OASIS Standard

23 January 2013

Specification URIs

This version:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.odt (Authoritative)
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.html
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.pdf

Previous version:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/cs02/AS4-profile-v1.0-cs02.odt (Authoritative)
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/cs02/AS4-profile-v1.0-cs02.html
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/cs02/AS4-profile-v1.0-cs02.pdf

Latest version:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/AS4-profile-v1.0.odt (Authoritative)
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/AS4-profile-v1.0.html
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/AS4-profile-v1.0.pdf

Technical Committee:

OASIS ebXML Messaging Services TC

Chairs:

Makesh Rao (marao@cisco.com), Cisco Systems, Inc.
Sander Fieten (sander@fieten-it.com), Individual

Editors:

Jacques Durand (jdurand@us.fujitsu.com), Fujitsu America Inc.
Pim van der Eijk (pvde@sonnenglanz.net), Sonnenglanz Consulting

Additional artifacts:

This prose specification is one component of a Work Product which also includes:

- XML examples: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/examples/

Related work:

This specification is related to:

- *OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features.* 01 October 2007. OASIS Standard. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms_core-3.0-spec-os.html
- *OASIS ebXML Messaging Services Version 3.0: Part 2, Advanced Features*. Latest version. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-v3-part2.html

Abstract:

While ebMS 3.0 represents a leap forward in reducing the complexity of Web Services B2B messaging, the specification still contains numerous options and comprehensive alternatives for addressing a variety of scenarios for exchanging data over a Web Services platform. The AS4 profile of the ebMS 3.0 specification has been developed in order to bring continuity to the principles and simplicity that made AS2 successful, while adding better compliance to Web Services standards, and features such as message pulling capability and a built-in Receipt mechanism. Using ebMS 3.0 as a base, a subset of functionality is defined along with implementation guidelines adopted based on the “just-enough” design principles and AS2 functional requirements to trim down ebMS 3.0 into a more simplified and AS2-like specification for Web Services B2B messaging. In addition to addressing EDIINT requirements, a Minimal Client conformance profile is provided that addresses lower-end exchange requirements. This document defines the AS4 profile as a combination of conformance profiles that concern an implementation capability, and of a usage profile that concerns how to use this implementation. A couple of variants are defined for the AS4 conformance profile - the AS4 ebHandler profile, the AS4 Light Client profile and the AS4 Minimal Client profile - which reflect different endpoint capabilities.

Status:

This document was last revised or approved by the membership of OASIS on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document.

Technical Committee members should send comments on this Work Product to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee’s web page at http://www.oasis-open.org/committees/ebxml-msg/.

For information on whether any patents have been disclosed that may be essential to implementing this Work Product, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page at http://www.oasis-open.org/committees/ebxml-msg/ipr.php.

Citation format:

When referencing this Work Product the following citation format should be used:

[AS4-Profile]

AS4 Profile of ebMS 3.0 Version 1.0. 23 January 2013. OASIS Standard. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.html.

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

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 section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark for above guidance.

Table of Contents

1 Introduction

1.1 Rationale and Context

Historically, the platform for mission-critical business-to-business (B2B) transactions has steadily moved from proprietary value-added networks (VANs) to Internet-based protocols free from the data transfer fees imposed by the VAN operators. This trend has been accelerated by lower costs and product ownership, a maturing of technology, internationalization, widespread interoperability, and marketplace momentum. The exchange of EDI business documents over the Internet has substantially increased along with a growing presence of XML and other document types such as binary and text files.

The Internet messaging services standards that have emerged provide a variety of options for end users to consider when deciding which standard to adopt. These include pre-Internet protocols, the EDIINT series of AS1 [RFC3335] AS2 [RFC4130] and AS3 [RFC4823], simple XML over HTTP, government specific frameworks, ebMS 2.0 [ebMS2], and Web Services variants. As Internet messaging services standards have matured, new standards are emerging that leverage prior B2B messaging services knowledge for applicability to Web Services messaging.

The emergence of the OASIS ebMS 3.0 Standard [ebMS3CORE] represents a leap forward in Web Services B2B messaging services by meeting the challenge of composing many Web Services standards into a single comprehensive specification for defining the secure and reliable exchange of documents using Web Services. The ebMS 3.0 standard composes the fundamental Web Services standards SOAP 1.1 [SOAP11], SOAP 1.2 [SOAP12], SOAP with Attachments [SOAPATTACH], WS-Security ([WSS10], [WSS11]), WS-Addressing [WSADDRCORE], and reliable messaging (WS-Reliability 1.1 [WSR11] or WS-ReliableMessaging - currently at version 1.2 [WSRM12]) together with guidance for the packaging of messages and receipts along with definitions of messaging choreographies for orchestrating document exchanges.

Like AS2, ebMS 3.0 brings together many existing standards that govern the packaging, security, and transport of electronic data under the umbrella of a single specification document. While ebMS 3.0 represents a leap forward in reducing the complexity of Web Services B2B messaging, the specification still contains numerous options and comprehensive alternatives for addressing a variety of scenarios for exchanging data over a Web Services platform.

In order to fully take advantage of the AS2 success story, this profile of the ebMS 3.0 specification has been developed. Using ebMS 3.0 as a base, a subset of functionality has been defined along with implementation guidelines adopted based on the “just-enough” design principles and AS2 functional requirements to trim down ebMS 3.0 into a more simplified and AS2-like specification for Web Services B2B messaging. The main benefits of AS4 compared to AS2 are:

- Compatibility with Web services standards.
- Message pulling capability.
- A built-in Receipt mechanism

AS4 also provides a Minimal Client conformance profile that supports data exchanges that have lower-end requirements and do not require (the equivalent of) some of the more advanced capabilities of AS2 and ebMS 3.0, such as support for multiple payloads, message receipts and signing or encryption of messages and receipts.

Profiling ebMS V3 means:

- Defining a subset of ebMS V3 options to be supported by the AS4 handler.
- Deciding which types of message exchanges must be supported, and how these exchanges should be conducted (level of security, binding to HTTP, etc.).

- Deciding of AS4-specific message contents and practices (how to make use of the ebMS message header fields, in an AS4 context).
- Deciding of some operational best practices, for the end-user.

The overall goal of a profile for a standard is to ensure interoperability by:

- Establishing particular usage and practices of the standard within a community of users.
- Defining the subset of features in this standard that needs to be supported by an implementation.

Two kinds of profiles are usually to be considered when profiling an existing standard:

1. **Conformance Profiles**. These define the different ways a product can conform to a standard, based on specific ways to implement this standard. A conformance profile is usually associated with a specific conformance clause. Conformance profiles are of prime interest for product managers and developers: they define a precise subset of features to be supported.
2. **Usage Profiles** (also called Deployment Profiles). These define how a standard should be used by a community of users, in order to ensure best compatibility with business practices and interoperability. Usage profiles are of prime interest for IT end-users: they define how to configure the use of a standard (and related product) as well as how to bind this standard to business applications. A usage profile usually points at required or compatible conformance profile(s).

AS4 is defined as a combination of:

- Three primary AS4 conformance profiles (see section 2) that define three subsets of ebMS V3 features, at least one of which is to be supported by an AS4 implementation.
- An optional complementary conformance profile (see section 4) that specifies how to use AS4 endpoints with ebMS 3.0 intermediaries. This is based on a simplified subset of the multi-hop messaging feature defined in ebMS 3.0 Part 2, Advanced Features specification [ebMS3ADV].
- An AS4 Usage Profile (see section 4) that defines how to use an AS4-compliant implementation in order to achieve similar functions as specified in AS2.

The three primary AS4 conformance profiles (CP) are defined below:

(1) The **AS4 ebHandler CP**. This conformance profile supports both Sending and Receiving roles, and for each role both message pushing and message pulling.

(2) The **AS4 Light Client CP**. This conformance profile supports both Sending and Receiving roles, but only message pushing for Sending and message pulling for Receiving. In other words, it does not support incoming HTTP requests, and may have no fixed IP address.

(3) The **AS4 Minimal Client CP**. Like the Light Client CP, this conformance profile does not support the push transport channel binding for the Receiving role and therefore does not require HTTP server capabilities. As its name indicates, this CP omits all but a minimal set of features.

Compatible existing conformance profiles for ebMS V3 are:

- Gateway RM V3 or Gateway RX V3: a Message Service Handler (MSH) implementing any of these profiles will also be conforming to the AS4 ebHandler CP (the reverse is not true).

NOTE: Full compliance to AS4 actually requires and/or authorizes a message handler to implement a few additional features beyond the above Conformance Profiles, as described in the Conformance Section 6. These additional features are described in Section 3.

1.2 Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in [RFC2119].

1.3 Normative References

[ebBP-SIG] *OASIS ebXML Business Signals Schema*, 21 December 2006. OASIS Standard. http://docs.oasis-open.org/ebxml-bp/ebbp-signals-2.0

[ebMS3CORE] *OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features*, 1 October 2007, OASIS Standard. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms_core-3.0-spec.pdf

[ebMS3ADV] *OASIS ebXML Messaging Services Version 3.0: Part 2, Advanced Features.* Committee Specification 01, 19 May 2011. OASIS Committee Specification. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-v3-part2.odt

[ebMS3-CP] *OASIS ebXML Messaging Services, Version 3.0: Conformance Profiles.* OASIS Committee Specification, 24 April 2010. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/ebms3-confprofiles.pdf

[RFC1952] *GZIP file format specification version 4.3*. IETF RFC. May 1996. http://tools.ietf.org/html/rfc1952

[RFC2045] *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.* IETF RFC. November 1996. http://www.ietf.org/rfc/rfc2045.txt

[RFC2119] *Key words for use in RFCs to Indicate Requirement Levels*. IETF RFC. March 1997. http://www.ietf.org/rfc/rfc2119.txt

[RFC2616] *Hypertext Transfer Protocol – HTTP/1.1.* IETF RFC. June 1999. http://www.ietf.org/rfc/rfc2616.txt

[SOAP12] *SOAP Version 1.2 Part 1: Messaging Framework.* W3C Recommendation. 27 April 2007. http://www.w3.org/TR/soap12-part1/

[SOAPATTACH] *SOAP Messages with Attachments*, W3C Note. 11 December 2000. http://www.w3.org/TR/SOAP-attachments

[WSADDRCORE] *Web Services Addressing 1.0 – Core.* W3C Recommendation. 9 May 2006. http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/

[WSIAP10] *WS-I Attachments Profile Version 1.0*, WS-I Final Material. 20 April 2004. http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html

[WSIBP20] *Basic Profile Version 2.0*, WS-I Final Material. 9 November 2010. http://ws-i.org/Profiles/BasicProfile-2.0-2010-11-09.html

[WSIBSP11] *Basic Security Profile Version 1.1*, WS-I Final Material. 24 January 2010. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.1.html

[WSS11] *Web Services Security: SOAP Message Security 1.1.* OASIS Standard incorporating Approved Errata. 1 November 2006, http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-errata-os-SOAPMessageSecurity.pdf

[WSS11-UT] *Web Services Security UsernameToken Profile 1.1.* OASIS Standard. 1 February 2006. http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-UsernameTokenProfile.pdf.

[WSS11-X509] *Web Services Security X.509 Certificate Token Profile 1.1*. OASIS Standard incorporating Approved Errata. 1 November 2006. http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-errata-os-x509TokenProfile.pdf

[XML10] *Extensible Markup Language (XML) 1.0.* W3C Recommendation 26 November 2008. http://www.w3.org/TR/REC-xml/

[XMLDSIG] *XML-Signature Syntax and Processing (Second Edition).* W3C Recommendation. 10 June 2008. http://www.w3.org/TR/xmldsig-core/

[XMLENC] *XML Encryption Syntax and Processing.* W3C Recommendation. 10 December, 2002. http://www.w3.org/TR/xmlenc-core/

1.4 Non-normative References

[CII] *UN/CEFACT Cross Industry Invoice Version 2.0*. UN/CEFACT Standard. http://www.unece.org/uncefact/data/standard/CrossIndustryInvoice_2p0.xsd

[ebCorePartyId] *OASIS ebCore Party Id Type Technical Specification Version 1.0*. OASIS Committee Specification, 28 September 2010. http://docs.oasis-open.org/ebcore/PartyIdType/v1.0/PartyIdType-1.0.odt

[ebBP] *OASIS ebXML Business Process Specification Schema Technical Specification v2.0.4.* OASIS Standard, 21 December 2006. http://docs.oasis-open.org/ebxml-bp/2.0.4/ebxmlbp-v2.0.4-Spec-os-en.odt

[ebCPPA] *Collaboration-Protocol Profile and Agreement Specification Version 2.0. OASIS Standard*, September, 2002. http://www.oasis-open.org/committees/ebxml-cppa/documents/ebcpp-2.0.pdf

[ebMS2] *Message Service Specification Version 2.0*, OASIS Standard. 1 April 2002. http://www.oasis-open.org/committees/ebxml-msg/documents/ebMS_v2_0.pdf

[GLN] GS1 Global Location Number (GLN). http://www.gs1.org/barcodes/technical/idkeys/gln

[IIC-DP] *Deployment Profile Template For OASIS ebXML Message Service 2.0 Standard.* OASIS Public Review Draft, 4 December 2006**.** http://docs.oasis-open.org/ebxml-iic/ebXML_DPT-v1.1-ebMS2-template-pr-01.pdf

[RFC3335] *MIME-based Secure Peer-to-Peer Business Data Interchange over the Internet (AS1)*. IETF RFC, September 2002. http://tools.ietf.org/html/rfc3335

[RFC3798] *Message Disposition Notification*. IETF RFC, May 2004. http://tools.ietf.org/html/rfc3798

[RFC4130] *MIME-Based Secure Peer-to-Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2)*. IETF RFC, July 2005. http://tools.ietf.org/rfc/rfc4130

[RFC4823] *FTP Transport for Secure Peer-to-Peer Business Data Interchange over the Internet (AS3)*. IETF RFC, April 2007. http://tools.ietf.org/html/rfc4823

[SOAP11] *Simple Object Access Protocol (SOAP) 1.1*, W3C Note. 08 May 2000. http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[WSIBP12] *Basic Profile Version 1.2.* WS-I Final Material. 09 November 2010. http://ws-i.org/Profiles/BasicProfile-1.2-2010-11-09.html

[WSR11] *WS-Reliability 1.1.* OASIS Standard, 15 November 2004. http://docs.oasis-open.org/wsrm/ws-reliability/v1.1/wsrm-ws_reliability-1.1-spec-os.pdf

[WSRM12] *Web Services Reliable Messaging (WS-ReliableMessaging) Version 1.2*, OASIS Standard. 2 February 2009, http://docs.oasis-open.org/ws-rx/wsrm/200702/wsrm-1.2-spec-os.doc

[WSS10] *Web Services Security: SOAP Message Security 1.0*, 2004. http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

2 AS4 Conformance Profiles for ebMS V3 Core Specification

AS4 is more than a conformance profile, in the sense given in **[ebMS3-CP]**. It is a combination of a conformance profile and a usage profile, as explained in the introduction section. Consequently, only this section (section 2) is conforming to the format recommended in **[ebMS3-CP]** for describing conformance profiles. The usage profile part (section 5) is following a format based on tables similar to those found in **[IIC-DP]**.

2.1 The AS4 ebHandler Conformance Profile

The AS4 ebHandler Conformance Profile addresses common functional requirements of e-Business/e-Government gateways. It is identified by the URI:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/cprofiles/200809/as4ebhandler

(Note: this URI is only an identifier, not a document address.)

2.1.1 Feature Set

The AS4 CP is defined as follows, using the table template and terminology provided in Appendix A ("Conformance") of the core ebXML Messaging Services V3.0 Conformance Profiles specification [ebMS3-CP]**.**

Conformance Profile: **AS4 ebHandler**	**Profile summary**: <"Sending+Receiving" / "AS4 ebHandler" / Level 1 / HTTP 1.1 + SOAP 1.2 + WSS 1.1 >
Functional Aspects	**Profile Feature Set**
ebMS MEP	The following ebMS simple Message Exchange Patterns (MEPs) MUST be supported both as Initiating and Responding partner: ● One-way / Push ● One-way / Pull NOTE: This does not prevent an implementation to support asynchronous Two-way MEPs. Regardless of which MEP is used, the sending of an `eb:Receipt` message MUST be supported: ● For the One-way / Push, both "response" and "callback" reply patterns MUST be supported. ● For the One-way / Pull, the "callback" pattern is the only viable option, and the User message sender MUST be ready to accept an `eb:Receipt` either piggybacked on (or bundled with) an `eb:PullRequest`, or piggybacked on another User Message, or sent separately. In all MEPs, the User message receiver MUST be able to send an `eb:Receipt` as a separate message (i.e. not piggybacked on an `eb:PullRequest` message or on another User message). An MSH conforming to this profile is therefore NOT required to bundle an `eb:Receipt` with any other ebMS header or message body.

	Use of the `ebbpsig:NonRepudiationInformation` element (as defined in [ebBP-SIG) is REQUIRED as content for the `eb:Receipt` message, i.e. when conforming to this profile a Receiving MSH must be able to create a Receipt with such a content, and a Sending MSH must be able to process it.
Reliability	Reception Awareness, defined as the ability for a Sending ebHandler to notify its application (message Producer) of lack of reception of an `eb:Receipt` related to a sent message, MUST be supported. This implies support for: ● Correlating `eb:Receipt` elements with previously sent User messages, based on the ebMS message ID ● Detection of a missing `eb:Receipt` for a sent message ● Ability to report an error to the message Producer in case no `eb:Receipt` has been received for a sent message. The semantics for sending back an `eb:Receipt` message is as follows: a well-formed ebMS user message has been received and the MSH is taking responsibility for its processing (additional application-level delivery semantics, and payload validation semantics are not relevant). Support for a WS reliable messaging specification is optional .
Security	The following security features MUST be supported: ● Support for username / password token, digital signatures and encryption. ● Support for content-only transforms. ● Support for security of attachments. ● Support for message authorization at P-Mode level (see 7.10 in [ebMS3-CORE]) Authorization of the Pull signal , for a particular MPC , must be supported at minimum. ● Transport-level secure protocols such as SSL or TLS Two authorization options MUST be supported by an MSH in the Receiving role, and at least one of them in the Sending role: ● **Authorization Option 1**: Use of the WSS security header targeted to the “ebms” actor, as specified in section 7.10 of the ebMS V3.0 Core Specification, 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. ● **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 MPC. NOTE on XMLDsig: XMLDsig allows arbitrary XSLT transformations when constructing the plaintext over which a signature or reference is created. Conforming applications that allow use of XSLT transformations when verifying either signatures or references are encouraged to maintain lists of “safe” transformations for a given partner, service, action and role combination. Static analysis of XSLT expressions with a human user audit is encouraged for trusting a given expression as “safe” .

	The use of transport-level secure protocols such as SSL or TLS is RECOMMENDED.
Error generation and reporting	The following error processing capabilities MUST be supported: ● Capability of the Receiving MSH to report errors from message processing, either as ebMS error messages or as SOAP Faults to the Sending MSH. The following modes of reporting to a Sending MSH are supported: ● Sending error as a separate request (**ErrorHandling.Report.ReceiverErrorsTo**=<URL of Sending MSH>) ● Sending error on the back channel of the underlying protocol (**ErrorHandling.Report.AsResponse**="true"). ● Capability to report to a third-party address (**ErrorHandling.Report.ReceiverErrorsTo**=<other address>). ● Capability of Sending MSH to report generated errors as notifications to the message producer (support for **Report.ProcessErrorNotifyProducer**="true")(e.g. delivery failure). ● Generated errors: All specified errors in [ebMS3CORE] must be generated when applicable, except for EBMS:0010: On a Receiving MSH, there is no requirement to generate error EBMS:0010 for discrepancies between message header and the P-Mode.reliability and P-Mode.security features. It is required to generate such errors, on a Receiving MSH, for other discrepancies.
Message Partition Channels	Message partition channels (MPC) MUST be supported in addition to the default channel, so that selective pulling by a partner MSH is possible. This means AS4 handlers MUST be able to use the `@mpc` attribute and to process it as expected.
Message packaging	The following features MUST be supported both on sending and receiving sides: ● Support for attachments. ● Support for Message Properties. ● Support for processing messages that contain both a signal message unit (`eb:SignalMessage`) and a user message unit (`eb:UserMessage`) – this may happen when a same ebMS message carries message units for different MEP instances. NOTE: per WS-I Basic Profile 2.0, at most one payload may be inserted as direct child element of the SOAP Body.
Interoperability Parameters	The following interoperability parameters values MUST be supported for this conformance profile: ● **Transport:** HTTP 1.1 ● **SOAP version:** 1.2 ● **Reliability Specification:** none. ● **Security Specification:** WSS 1.1.

2.1.2 WS-I Conformance Profiles

The Web-Services Interoperability consortium has defined guidelines for interoperability of SOAP messaging implementations. In order to ensure maximal interoperability across different SOAP stacks, eg.

MIME and HTTP implementations, compliance with the following WS-I profiles is REQUIRED whenever related features are used:

- Basic Security Profile (BSP) 1.1 [WSIBSP11].
- Attachment Profile (AP) 1.0 [WSIAP10] with regard to the use of MIME and SOAP with Attachments.

Notes:

- Compliance with AP1.0 would normally require compliance with BP1.1, which in turn requires the absence of a SOAP Envelope in the HTTP response of a One-Way MEP (R2714). However, recent BP versions such as BP1.2 [WSIBP12] and BP2.0 [WSIBP20] override this requirement. Consequently, the AS4 ebHandler conformance profile does not require conformance to these deprecated requirements inherited from BP1.1 (R2714, R1143) regarding the use of HTTP.
- WS-I compliance is here understood as requiring that the features exhibited by an AS4 ebHandler MUST comply with the above WS-I profiles. For example, since only SOAP 1.2 is required by the AS4 ebHandler, the requirements from BSP 1.1 that depend on SOAP 1.1 would not apply. Similarly, none of the requirements for DESCRIPTION (WSDL) or REGDATA (UDDI) apply here, as these are not used.

This conformance profile also requires conformance to the following WS-I profiles :

- Basic Profile 2.0 (BP2.0) [WSIBP20].

2.1.3 Processing Mode Parameters

This section contains a summary of P-Mode parameters relevant to AS4 features for this conformance profile. An AS4 handler MUST support and understand those that are mentioned as "required". For each parameter, either:

- Full support is required: An implementation MUST support the possible options for this parameter.
- Partial support is required: Support for a subset of values is required.
- No support is required: An implementation is not required to support the features controlled by this parameter, and therefore is not required to understand this parameter.

An AS4 handler is expected to support the P-Mode set below both as a Sender (of the user message) and as a Receiver.

2.1.3.1 General P-Mode parameters

- **PMode.ID**: support required.
- **PMode.Agreement:** support required.
- **PMode.MEP:** support required for**:** http://www.oasis-open.org/committees/ebxml-msg/one-way
- **PMode.MEPbinding:** support required for**: http://www.oasis-open.org/committees/ebxml-msg/push** and **http://www.oasis-open.org/committees/ebxml-msg/pull**.
- **PMode.Initiator.Party:** support required**.**
- **PMode.Initiator.Role:** support required**.**
- **PMode.Initiator.Authorization.username** and **PMode.Initiator.Authorization.password:** support required for: wsse:UsernameToken.
- **PMode.Responder.Party:** support required**.**

- **PMode.Responder.Role:** support required**.**
- **PMode.Responder.Authorization.username** and **PMode.Responder.Authorization.password:** support required for: wsse:UsernameToken.

2.1.3.2 PMode[1].Protocol

- **PMode[1].Protocol.Address:** support required for “http” protocol.
- **PMode[1].Protocol.SOAPVersion:** support required for SOAP 1.2.

2.1.3.3 PMode[1].BusinessInfo

- **PMode[1].BusinessInfo.Service:** support required**.**
- **PMode[1].BusinessInfo.Action:** support required**.**
- **PMode[1].BusinessInfo.Properties[]:** support required.
- **(PMode[1].BusinessInfo.PayloadProfile[]:** support not required**)**
- **(PMode[1].BusinessInfo.PayloadProfile.maxSize:** support not required**)**

2.1.3.4 PMode[1].ErrorHandling

- **(PMode[1].ErrorHandling.Report.SenderErrorsTo:** support not required**)**
- **PMode[1].ErrorHandling.Report.ReceiverErrorsTo:** support required (for address of the MSH sending the message in error or for third-party).
- **PMode[1].ErrorHandling.Report.AsResponse:** support required (true/false).
- **(PMode[1].ErrorHandling.Report.ProcessErrorNotifyConsumer** support not required**)**
- **PMode[1].ErrorHandling.Report.ProcessErrorNotifyProducer**: support required (true/false)
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer:** support required (true/false)

2.1.3.5 PMode[1].Reliability

Support not required.

2.1.3.6 PMode[1].Security

- **PMode[1].Security.WSSVersion:** support required for: 1.1
- **PMode[1].Security.X509.Sign:** support required.
- **PMode[1].Security.X509.Signature.Certificate:** support required.
- **PMode[1].Security.X509.Signature.HashFunction:** support required.
- **PMode[1].Security.X509.Signature.Algorithm:** support required.
- **PMode[1].Security. X509.Encryption.Encrypt:** support required.
- **PMode[1].Security.X509.Encryption.Certificate:** support required.
- **PMode[1].Security.X509.Encryption.Algorithm:** support required.
- **(PMode[1].Security.X509.Encryption.MinimumStrength:** support not required**)**
- **PMode[1].Security.UsernameToken.username:** support required.

- **PMode[1].Security.UsernameToken.password:** support required.
- **PMode[1].Security.UsernameToken.Digest:** support required (true/false)
- **(PMode[1].Security.UsernameToken.Nonce:** support not required**)**
- **PMode[1].Security.UsernameToken.Created:** support required.
- **PMode[1].Security.PModeAuthorize:** support required (true/false)
- **PMode[1].Security.SendReceipt:** support required (true/false)
- **Pmode[1].Security.SendReceipt.ReplyPattern:** support required (both “response” and “call-back”))

2.2 The AS4 Light Client Conformance Profile

The AS4 Light Client Conformance Profile addresses common functional requirements of e-Business/e-Government light gateways. It is identified by the URI:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/cprofiles/200809/as4lightclient

(Note: this URI is only an identifier, not a document address.)

As indicated by its name, this profile applies only to one side of an MEP (acting as a “client” to the other party). It is not required and often not even possible for two MSHs conforming to this profile to engage in a point-to-point exchange. Indeed, at least one MSH must be ready to receive an incoming HTTP request in any MEP as defined in ebMS, but this profile does not require this capability. As a result, when an MSH is conforming exclusively to this profile, it can only engage into point-to-point exchanges with MSHs that conform to “more” than this profile – e.g. MSHs that conform to the ebHandler profile– in order to be able to receive requests. Two light clients can also exchange messages using store-and-forward ebMS3 intermediaries, as described in section 4.

2.2.1 Feature Set

Conformance Profile: **AS4 Light Client**	**Profile summary**: <“Sending+Receiving” / “AS4 Light Client” / Level 1 / HTTP 1.1 + SOAP 1.2>
Functional Aspects	**Profile Feature Set**
ebMS MEP	The following Message Exchange Patterns (MEPs) MUST be supported as Initiating partner: ● One-way / Push ● One-way / Pull NOTE: This does not prevent an implementation to support Two-way MEPs. The following requirement details apply for each MEP: ● For the One-way / Push, the “response” reply pattern MUST be supported on the **PMode[1].Security.SendReceipt.ReplyPattern** parameter by the initiating client MSH. ● For the One-way / Pull, the “callback” pattern is the only viable option, and the receiving MSH (initiating light client) MUST be able to send an `eb:Receipt` separately from the `eb:PullRequest`. It MAY additionally be able to send an `eb:Receipt` piggybacked on an `eb:PullRequest`.

	In all MEPs, the User Message receiver MUST be able to send an eb:Receipt as a separate message (i.e. not piggybacked on an eb:PullRequest message or on another User message). An MSH conforming to this profile is therefore NOT REQUIRED to bundle an eb:Receipt with any other ebMS header or message body. However, when receiving an eb:Receipt, an MSH conforming to this profile MUST be able to process an eb:Receipt bundled with an other ebMS message header or body. Use of the ebbpsig:NonRepudiationInformation element (as defined in [ebBP-SIG) is REQUIRED as content for the eb:Receipt message, i.e. when conforming to this profile a Receiving MSH must be able to create a Receipt with such a content, and a Sending MSH must be able to process it.
Reliability	Reception Awareness, defined as the ability for a Sending light Client to notify its application (message Producer) of lack of reception of an eb:Receipt related to a sent message, MUST be supported. This implies support for: ● Correlating eb:Receipt elements with previously sent User messages, based on the ebMS message ID. ● Detection of a missing eb:Receipt for a sent message. ● Ability to report an error to the message Producer in case no eb:Receipt has been received for a sent message. The semantics for sending back an eb:Receipt message is as follows: a well-formed ebMS user message has been received and the MSH is taking responsibility for it's processing, (additional application-level delivery semantics, and payload validation semantics are not relevant). Support for a WS reliable messaging specification is optional.
Security	Both authorization options for message pulling (authorizing an eb:PullRequest for a particular MPC) described in the ebHandler conformance profile MUST be supported: 1. Support for username / password token: minimal support for wss:UsernameToken profile in the Pull signal - for authorizing a particular MPC. Support for adding a WSS security header targeted to the “ebms” actor, as specified in section 7.10 of ebMS V3, with the wsse:UsernameToken profile. The use of transport-level secure protocol such as SSL or TLS is recommended. 2. Support for a regular wsse security header (XMLDsig for authentication, use of X509), and no additional wsse security header targeted to “ebms”. The use of transport-level secure protocols such as SSL or TLS is RECOMMENDED.
Error generation and reporting	Error notification to the local message producer MUST be supported (e.g. reported failure to deliver pushed messages). The reporting of message processing errors for pulled messages to the remote party MUST be supported via Error messages (errors may be bundled with another pushed message or a Pull Request signal message.).
Message Partition Channels	Sending on the default message partition channel is sufficient (support for additional message partitions is NOT REQUIRED.)
Message packaging	Support for attachments is REQUIRED – i.e. an XML message payload may use the SOAP body or a MIME part. Support for Message Properties is REQUIRED.

	NOTE: per WS-I Basic Profile 2.0, at most one payload may be inserted as direct child element of the SOAP Body.
Interoperability Parameters	The following interoperability parameters values MUST be supported for this conformance profile: ● **Transport:** HTTP 1.1 ● **SOAP version:** 1.2 ● **Reliability Specification:** none. ● **Security Specification:** WSS 1.1.

2.2.2 WS-I Conformance Requirements

This conformance profile will require compliance with the following WS-I profile :

1. Basic Profile 2.0 (BP2.0) [WSIBP20].

Note: this must be interpreted as requiring that the features exhibited by an AS4 Light Client ebMS conformance profile MUST comply with the above WS-I profile.

2.2.3 Processing Mode Parameters

This section contains a summary of P-Mode parameters relevant to AS4 features for this conformance profile. An AS4 Light client MUST support and understand those that are mentioned as "required". For each parameter, either:

- Full support is required: An implementation is supposed to support the possible options for this parameter.
- Partial support is required: Support for a subset of values is required.
- No support is required: An implementation is not required to support the features controlled by this parameter, and therefore not required to understand this parameter.

An AS4 Light client is expected to support the P-Mode set below both as a Sender (of the user message, in case of a one-way / push) and as a Receiver (in case of a one-way / pull).

2.2.3.1 General P-Mode parameters

- **PMode.ID**: support required.
- **PMode.Agreement:** support required.
- **PMode.MEP:** support required for: http://www.oasis-open.org/committees/ebxml-msg/one-way
- **PMode.MEPbinding:** support required for: **http://www.oasis-open.org/committees/ebxml-msg/push** and **http://www.oasis-open.org/committees/ebxml-msg/pull**.
- **PMode.Initiator.Party:** support required**.**
- **PMode.Initiator.Role:** support required**.**
- **PMode.Initiator.Authorization.username** and **PMode.Initiator.Authorization.password:** support required for: wsse:UsernameToken. (as initiator of the one-way / pull)
- **PMode.Responder.Party:** support required**.**
- **PMode.Responder.Role:** support required**.**

- **PMode.Responder.Authorization.username** and **PMode.Responder.Authorization.password:** support not required.

2.2.3.2 PMode[1].Protocol

- **PMode[1].Protocol.Address:** support required for “http” protocol.
- **PMode[1].Protocol.SOAPVersion:** support required for SOAP 1.2.

2.2.3.3 PMode[1].BusinessInfo

- **PMode[1].BusinessInfo.Service:** support required**.**
- **PMode[1].BusinessInfo.Action:** support required**.**
- **PMode[1].BusinessInfo.Properties[]:** support required.
- **(PMode[1].BusinessInfo.PayloadProfile[]:** support not required**)**
- **(PMode[1].BusinessInfo.PayloadProfile.maxSize:** support not required**)**

2.2.3.4 PMode[1].ErrorHandling

- **(PMode[1].ErrorHandling.Report.SenderErrorsTo:** support not required**)**
- **PMode[1].ErrorHandling.Report.AsResponse:** support required (true/false) as initiator of the one-way / push, as well as for the `eb:PullRequest` signal (PMode[1][s]).
- **(PMode[1].ErrorHandling.Report.ProcessErrorNotifyConsumer** support not required**)**
- **PMode[1].ErrorHandling.Report.ProcessErrorNotifyProducer**: support required (true/false)
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer:** support required (true/false)

2.2.3.5 Pmode[1].Reliability

Support not required.

2.2.3.6 PMode[1].Security

- **PMode[1].Security.WSSVersion:** support required for: 1.1
- **PMode[1].Security.X509.Sign:** support required.
- **PMode[1].Security.X509.Signature.Certificate:** support required.
- **PMode[1].Security.X509.Signature.HashFunction:** support required.
- **PMode[1].Security.X509.Signature.Algorithm:** support required.
- **PMode[1].Security. X509.Encryption.Encrypt:** support not required.
- **PMode[1].Security.X509.Encryption.Certificate:** support not required.
- **PMode[1].Security.X509.Encryption.Algorithm:** support not required.
- **(PMode[1].Security.X509.Encryption.MinimumStrength:** support not required**)**
- **PMode[1].Security.UsernameToken.username:** support required.
- **PMode[1].Security.UsernameToken.password:** support required.
- **PMode[1].Security.UsernameToken.Digest:** support required (true/false)

- **(PMode[1].Security.UsernameToken.Nonce:** support not required**)**
- **PMode[1].Security.UsernameToken.Created:** support required.
- **PMode[1].Security.PModeAuthorize:** support required (true/false)
- **PMode[1].Security.SendReceipt:** support required (true/false)
- **Pmode[1].Security.SendReceipt.ReplyPattern:** support required for “response”if PMode.MEP-binding is "push", and for "callback" if PMode.MEPbinding is "pull".

2.3 The AS4 Minimal Client Conformance Profile

The AS4 Minimal Client addresses low-end functional data exchange requirements. It also supports business processes that do not require signing of messages and of message receipts. It is identified by the URI:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/cprofiles/200809/as4minimalclient

(NOTE: this URI is only an identifier, not a document address.)

As indicated by its name, this profile applies only to one side of an MEP (acting as a “client” to the other party). It is not required and often not even possible for two MSHs conforming to this profile to engage in a point-to-point exchange. Indeed, at least one MSH must be ready to receive an incoming (HTTP) request in any MEP as defined in ebMS, but this profile does not require this capability. As a result, when an MSH is conforming exclusively to this profile, it can only engage into point-to-point exchanges with MSHs that conform to “more” than this profile –e.g. MSHs that conform to the ebHandler profile– in order to be able to receive requests.

2.3.1 Feature Set

Conformance Profile: **AS4 Minimal Client**	**Profile summary**: <“Sending” / “AS4 Minimal Client” / Level 1 / HTTP 1.1 + SOAP 1.2>
Functional Aspects	**Profile Feature Set**
ebMS MEP	The following Message Exchange Patterns (MEPs) MUST be supported as Initiating partner: ● One-way / Push ● One-way / Pull NOTE: The requirement to support Pull is relaxed in the AS4 Minimal Sender Conformance Clause. No support for Receipts is required: the **PMode[1].Security.SendReceipt** parameter does NOT need to be supported for value “true”.
Reliability	Support for a WS reliable messaging specification is NOT REQUIRED. Support for Reception Awareness is NOT REQUIRED.
Security	The first authorization option for message pulling (authorizing an `eb:PullRequest` for a particular MPC) described in the ebHandler conformance profile SHOULD be supported: 1. Support for adding a WSS security header targeted to the “ebms” actor, as specified in section 7.10 of ebMS V3, with the `wsse:UsernameToken` pro-

	file [WSS11-UT]. NOTE: This requirement is relaxed in the AS4 Minimal Sender Conformance Clause. Support for the WSS Web Services Security X.509 Certificate Token Profile [WSS11-X509] is NOT REQUIRED. The use of transport-level secure protocols such as SSL or TLS is RECOMMENDED.
Error generation and reporting	Error notification to the local message producer MUST be supported (e.g. reported failure to deliver pushed messages).
Message Partition Channels	Sending on the default message partition channel is sufficient (support for additional message partitions is NOT REQUIRED.)
Message packaging	Support for attachments is NOT REQUIRED – i.e. an XML message payload will always use the SOAP body. NOTE: per WS-I Basic Profile 2.0, at most one payload may be inserted as direct child element of the SOAP Body. Support for Message Properties is NOT REQUIRED.
Interoperability Parameters	The following interoperability parameters values MUST be supported for this conformance profile: ● **Transport:** HTTP 1.1 ● **SOAP version:** 1.2 ● **Reliability Specification:** none. ● **Security Specification:** none.

2.3.2 WS-I Conformance Requirements

This conformance profile will require compliance with the following WS-I profile :

2. Basic Profile 2.0 (BP2.0) [WSIBP20].

Note: this must be interpreted as requiring that the features exhibited by an AS4 Minimal Client ebMS conformance profile MUST comply with the above WS-I profile.

2.3.3 Processing Mode Parameters

This section contains a summary of P-Mode parameters relevant to AS4 features for this conformance profile. An AS4 Minimal client MUST support and understand those that are mentioned as "required". For each parameter, either:

- Full support is required: An implementation is supposed to support the possible options for this parameter.
- Partial support is required: Support for a subset of values is required.
- No support is required: An implementation is not required to support the features controlled by this parameter, and therefore not required to understand this parameter.

An AS4 Minimal client is expected to support the P-Mode set below as a Sender of the user message.

2.3.3.1 General P-Mode parameters

- **PMode.ID**: support required.

- **PMode.Agreement:** support required.
- **PMode.MEP:** support required for: http://www.oasis-open.org/committees/ebxml-msg/one-way
- **PMode.MEPbinding:** support required for: **http://www.oasis-open.org/committees/ebxml-msg/push**.
- **PMode.Initiator.Party:** support required**.**
- **PMode.Initiator.Role:** support required**.**
- **PMode.Initiator.Authorization.username** and **PMode.Initiator.Authorization.password:** support not required.
- **PMode.Responder.Party:** support required**.**
- **PMode.Responder.Role:** support required**.**
- **PMode.Responder.Authorization.username** and **PMode.Responder.Authorization.password:** support not required.

2.3.3.2 PMode[1].Protocol

- **PMode[1].Protocol.Address:** support required for “http” protocol.
- **PMode[1].Protocol.SOAPVersion:** support required for SOAP 1.2.

2.3.3.3 PMode[1].BusinessInfo

- **PMode[1].BusinessInfo.Service:** support required**.**
- **PMode[1].BusinessInfo.Action:** support required**.**
- **PMode[1].BusinessInfo.Properties[]:** support not required.
- **(PMode[1].BusinessInfo.PayloadProfile[]:** support not required**)**
- **(PMode[1].BusinessInfo.PayloadProfile.maxSize:** support not required**)**

2.3.3.4 PMode[1].ErrorHandling

- **(PMode[1].ErrorHandling.Report.SenderErrorsTo:** support not required**)**
- **PMode[1].ErrorHandling.Report.AsResponse:** support required (true/false) as initiator of the one-way / push.
- **(PMode[1].ErrorHandling.Report.ProcessErrorNotifyConsumer** support not required**)**
- **PMode[1].ErrorHandling.Report.ProcessErrorNotifyProducer**: support required (true/false)
- **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer:** support not required

2.3.3.5 Pmode[1].Reliability

Support not required.

2.3.3.6 Pmode[1].Security

Support not required.

- **PMode[1].Security.SendReceipt:** support not required.

2.4 Conformance Profiles Compatibility

The AS4 profile is compatible with the following ebMS V3 conformance profiles, defined in [ebMS3-CP]:

1. Gateway RM V2/3
2. Gateway RM V3
3. Gateway RX V2/3
4. Gateway RX V3

AS4 may be deployed on any MSH that conforms to one of the above conformance profiles.

NOTE: AS4 may also be deployed on an MSH that supports B2B messaging protocols other than ebMS, such as AS2 [RFC4130]. Such an MSH could be used by organizations that use AS2 for some business partners, or for some types of documents, and AS4 for others.

3 AS4 Additional Features

This section defines features that were not specified in the ebMS V3 Core Specification and therefore out of scope for the previous conformance profiles (ebHandler CP and Light Client CP). These features should be considered as additional capabilities that are either required by or made optional to AS4 implementations as indicated in the conformance clauses in section 6.

The profiling tables below can be used for adding user-defined profiling requirements to be adopted within a business community. Whenever the feature, or its profiling, is mandatory, the right-side column (Profile Requirement) will specify it.

3.1 Compression

The AS4 Compression feature provides configurable (de)compression of application payloads. AS4 messages containing compressed application payloads are built in conformance with the SOAP with Attachments (SwA) [SOAPATTACH] specification[1]. Each compressed payload is carried in a separate MIME body part. Compression of the SOAP envelope and/or of a payload contained within the SOAP Body of an ebMS Message is not supported by the feature described here. However, if compression of the SOAP envelope is required then the content-coding feature of HTTP/1.1 [RFC2616] MAY be used.

To compress the payload(s) of a message payload, the GZIP [RFC1952] compression algorithm MUST be used. Compression MUST be applied before payloads are attached to the SOAP Message.

The `eb:PartInfo` element in the message header that relates to the compressed message part, MUST have an `eb:Property` element with `@name ="CompressionType"`:

```
<eb:Property name="CompressionType">application/gzip</eb:Property>
```

The content type of the compressed attachment MUST be "`application/gzip`".

These are indicators to the receiving MSH that the sending MSH has compressed a payload part. The receiving AS4 MSH MUST decompress any payload part(s) compressed by the sending MSH before delivering the message.

When compression, signature and encryption are required, any attached payload(s) MUST be compressed prior to being signed and/or encrypted.

Packaging requirements:

- An `eb:PartInfo/eb:PartProperties/eb:Property/@name="MimeType"` value is REQUIRED to identify the MIME type of the payload before compression was applied.
- For XML payloads, an `eb:PartInfo/eb:PartProperties/eb:Property/@name="CharacterSet"` value is RECOMMENDED to identify the character set of the payload before compression was applied. The values of this property MUST conform to the values defined in section 4.3.3 of [XML10].

Example:

```
<eb:PartInfo href="cid:attachment1234@example.com" >
      <eb:PartProperties>
            <eb:Property name="MimeType">application/xml</eb:Property>
            <eb:Property name="CharacterSet">utf-8</eb:Property>
```

[1] Although a SOAP 1.2 version of SwA has not been formally submitted to W3C, it appears that most SOAP products have anticipated that usage, and after investigation, it appears that they have done so in a consistent, interoperable way. This specification is acknowledging these *de facto* upgrades of SwA, (see Appendix C of **[ebMS3CORE]**).

```
        <eb:Property name="CompressionType">application/gzip</eb:Property>
       </eb:PartProperties>
<eb:PartInfo>
```

An additional P-Mode parameter is defined, which MUST be supported as part of the compression feature:

- **PMode[1].PayloadService.CompressionType:** (either absent, empty or equal to "application/gzip")

Value="application/gzip": the AS4 sending MSH SHOULD compress the attached payload(s) over this MEP segment. GZIP compression of payloads in data formats that provide native, built-in compression typically often does not result in good compression ratios and is therefore NOT REQUIRED.

Absent or empty (default): no compression is used over this MEP segment.

In case of error during decompression, the following error MUST be used: Code = EBMS:0303, Short Description = DecompressionFailure, Severity = Failure, Category = Communication.

3.2 Reception Awareness features and Duplicate Detection

These capabilities make use of the eb:Receipt as the sole type of acknowledgment. Duplicate detection only relies on the eb:MessageInfo/eb:MessageId.

Features	Profile requirements
Reception awareness error handling (REQUIRED support)	Ability for the MSH expecting an eb:Receipt to generate an error in case no eb:Receipt has been received for a sent message. It is RECOMMENDED that this error be a new error: Code = EBMS:0301, Short Description = MissingReceipt, Severity = Failure, Category = Communication. Ability for the MSH expecting an eb:Receipt to report a MissingReceipt error to the message Producer.
Message Retry (OPTIONAL support)	Ability for a User message sender that has not received an expected eb:Receipt to resend the User message. If doing so, the eb:MessageInfo/eb:MessageId element of the resent message and of the original User message MUST be same. When resending a message for which non-repudiation of receipt is required, the sender MUST ensure that the hash values for the digests to be included in the Receipt (i.e. the content of ebbpsig:MessagePartNRInformation elements), do not vary from the original message to the retry(ies), so that non-repudiation of receipt can be asserted based on the original message and the receipt of any of its retries.
Duplicate Detection (REQUIRED support)	Ability for the MSH receiving a User message to detect and/or eliminate duplicates based on eb:MessageInfo/eb:MessageId. If duplicates are just detected (not eliminated) then at the very least it is REQUIRED that the Receiving MSH notifies its application (message Consumer) of the duplicates. For examples, these could be logged. Related quantitative parameters (time window for the detection, or maximum message log size) are left to the implementation.

The following additional P-Mode parameters are defined   as part of the reception awareness feature:

- **PMode[1].ReceptionAwareness:** (true / false) Note: when set to true, the **PMode[1].Security.SendReceipt** must also be set to true.

- **PMode[1].ReceptionAwareness.Retry:** (true / false)

- **PMode[1].ReceptionAwareness.Retry.Parameters:**. (contains a composite string specifying: (a) maximum number of retries or some timeout, (b) frequency of retries or some retry rule). The string contains a sequence of parameters of the form: name=value, separated by either comas or ';'. Example: "maxretries=10,period=3000", in case the retry period is 3000 ms.

- **PMode[1].ReceptionAwareness.DuplicateDetection:** (true / false)

- **PMode[1].ReceptionAwareness.DetectDuplicates.Parameters:** (contains an implementation specific composite string. As an example this string may specify either (a) maximum size of message log over which duplicate detection is supported, (b) maximum time window over which duplicate detection is supported). The string contains a sequence of parameters of the form: name=value, separated by either comas or ';'. Example: "maxsize=10Mb,checkwindow=7D", in case the duplicate check window is guaranteed of 7 days minimum.

3.3 Alternative Pull Authorization

In addition to the two authorization options described in the AS4 Conformance Profile (section 2.1.1), 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 AS4 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. When using this option for authorizing pulling, there is no need to insert any WS-Security header in the Pull request at all.

3.4 Semantics of Receipt in AS4

The notion of Receipt in ebMS V3 is not associated with any particular semantics, such as delivery assurance. However, when combined with security (signing), it is intended to support Non Repudiation of Receipt (NRR).

In AS4, the `eb:Receipt` message serves both as a business receipt (its content is profiled in Section 2), and as a reception indicator, being a key element of the reception awareness feature. No particular delivery semantics can be assumed however: the sending of an `eb:Receipt` only means the following, from a message processing viewpoint:

- The related ebMS user message has been received and is well-formed.

- The message has been successfully processed by the Receiving MSH (i.e. not just "received"). Successful processing of a message means that none of the MSH operations needed over this message has generated an error.

- Because the latest steps of a message processing in the Receiving MSH (leading to actual "delivery" to the message Consumer) may vary greatly in their implementation from one implementation to the other, it is left to implementers to clarify to users at what exact step of the MSH processing flow the `eb:Receipt` is sent.

The meaning of NOT getting an expected Receipt, for the sender of a related user message, is one of the following:

1. The user message was lost and never received by the Receiving MSH.
2. The user message was received, but the `eb:Receipt` was never generated, e.g. due to a faulty configuration (P-Mode).
3. The user message was received, the `eb:Receipt` was sent back but was lost on the way.

See section 5.1.8 for AS4 usage rules about Receipts.

Note: The use of the phrase 'business receipt' in AS4 is to distinguish the nature of the AS4/ebMS3 receipt as being sufficient for Non-Repudiation of Receipt (NRR). In this sense it is very similar to the Message Disposition Notification (MDN, [RFC3798]) response that is used by AS2 as a business receipt for non-repudiation. This receipt in AS4/ebMS3 contains the same information as the MDN, and thus distinguishes itself from the web services reliable messaging (sequence) acknowledgment.

3.5 Sub-channels for Message Pulling

Optionally, the sub-channel feature defined in section 2 of ebMS V3 Part 2 ([ebMS3ADV]) for intermediaries in a multi-hop context, MAY be supported by an AS4 MSH. On the Sending side of an AS4 exchange, this feature will apply to a sending AS4 MSH in the same way it applies to the edge intermediary in ebMS V3 Part 2.

In short, this feature allows for a Producer application to submit messages intended for many receiving parties (i.e. different Client AS4 MSHs) over the same MPC, possibly covered by a single Pmode. This MPC is configured for message pulling and will be authorized for different pulling endpoints (AS4 Clients). This MPC is associated with a set of sub-channels to which different authorization credentials apply. Each client will be authorized to pull on its own sub-channel, Sub-channels are identified by an MPC Id extension as illustrated below:

If the MPC identifier is an URI of the form:

```
http://sender.example.com/mpc123
```

A sub-channel of this MPC may have an identifier of the form:

```
http://sender.example.com/mpc123/subc42
```

The `@mpc` attribute value in the message is not altered so the message is still considered as sent over this MPC (mpc123). The sub-channel identifier is only apparent in the pull Request messages generated by the Receiver MSH,

The following additional P-Mode parameter is defined and MUST be used when sub-channels are used:

- **Pmode[1].BusinessInfo.subMPCext**:: this parameter specifies the subchannel extension to be used. For example if `PMode[1].BusinessInfo.MPC =` “http://sender.example.com/mpc123” and `subMPCext =` “`subc42`” then the subchannel to pull from is: http://sender.example.com/mpc123/subc42.

On the Receiving MSH side, support for this feature means the ability to understand the above PMode parameter in order to issue `eb:PullRequest` signals with the proper subchannel MPC value, while being able to process received pulled messages that contain the MPC value corresponding to the core channel.

3.6 Additional Features Errors

The following error codes are extending the set of ebMS V3 error codes to support the AS4 additional features. They are to be generated and/or processed by an AS4 MSH depending on which feature is supported (i.e. depending on the conformance profile):

Error Code	Short Description	Recommended Severity	Category Value	Description or Semantics
EBMS:0301	MissingReceipt	failure	Communication	A Receipt has not been received for a message that was previously sent by the MSH generating this error.
EBMS:0302	InvalidReceipt	failure	Communication	A Receipt has been received for a message that was previously sent by the MSH generating this error, but the content does not match the message content (e.g. some part has not been acknowledged, or the digest associated does not match the signature digest, for NRR).
EBMS:0303	Decompression-Failure	failure	Communication	An error occurred during the decompression.

4 Complementary Requirements for the AS4 Multi-Hop Profile

The ebMS 3.0 Part 2, Advanced Features specification [ebMS3ADV] defines several advanced messaging features. One of these is a multi-hop feature that provides functionality to exchange ebMS messages through clouds of intermediaries, or *I-Clouds*. These intermediaries serve various purposes, including message routing and store-and-forward (or store-and-collect) connections. Intermediaries allow messages to flow through a *multi-hop* path and serve to interconnect (private or public) networks and clouds. This section specifies an optional profile for AS4 endpoints in order to converse with ebMS 3.0 intermediaries. This profile is complementary to the primary profiles defined in section 2. This complementary profile:

- Simplifies the fine-grained endpoint configuration options of [ebMS3ADV] to a single processing mode parameter (section 4.3).
- Extends the capability of AS4 endpoints to exchange messages in a peer-to-peer fashion to exchanges across intermediaries (section 4.4).

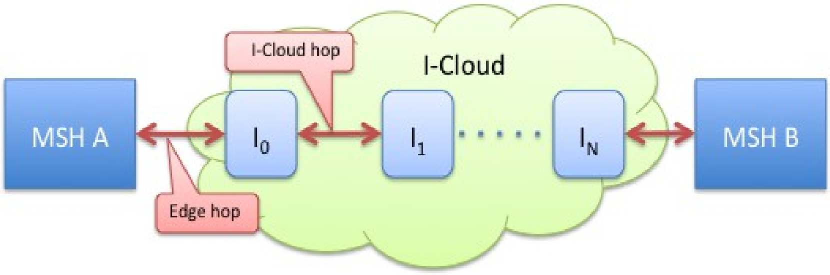
Section 4.1 is non-normative and provides the rationale and context for using AS4 and intermediaries. Section 4.2 defines some general constraints and assumptions. Section 4.3 presents the single additional processing mode parameter required for multi-hop. Section 4.4 provides a minimal interoperability subset for AS4 endpoints in an *I-Cloud*.

4.1 Rationale and Context

A key motivation for AS4 is to provide a simplified profile of ebMS 3.0 that allows Small and Medium-Size Enterprises (SMEs) to exchange messages using Web Services. Two situations can be distinguished:

- Situations where one partner in an exchange is an SME and the other is a larger organization. AS4 allows SME trading partners of a large organization to operate “client-only” endpoints and pull messages from a B2B gateway server operated by the large organization. That B2B gateway operates as a server and is addressable and available for pulling. These exchanges can be said to be *asymmetric*.
- Situations where all partners are SMEs, organized in collaborative SME B2B networks. In these situations there is no single larger partner that the other partners are organized around. These exchanges can be said to be *symmetric*.

When two endpoints exchange messages directly, they cannot both be client-only endpoints. Intermediaries can serve SME networks by offering store-and-collect capabilities, just like Internet Service Providers (ISPs) offer mailbox services for email, Value-Added Network (VAN) services offer document exchange services, and Cloud-based File Storage services offer secure temporary storage and exchange of large files.



In the diagram, messages can be sent any time to MSH A or MSH B as long as the I-Cloud is able to forward messages to AS4 edge intermediaries I_0 and I_N, from which they can be pulled at a convenient time.

4.2 General Constraints

This profile defines the following general constraints:

- Whether or not two AS4 endpoint exchange user messages in a peer-to-peer fashion or across an I-Cloud is determined by a single processing mode parameter.
- Sender and Receiver MSH can diverge in some “init” and “resp” parameters (terminology from section 2.7.2 of [ebMS3ADV]), as some parameters in an exchange relate to the edge intermediaries, not to the ultimate destination MSH.
- Whether or not an AS4 endpoint returns related response signals (receipts, errors) in a peer-to-peer fashion or across an I-Cloud is not based on configuration, but is determined by how the associated user message was delivered:
    - Receipts and errors for user messages received directly are sent back directly.
    - Receipts and errors for user messages received through an I-Cloud are sent back through the I-Cloud.
- Edge intermediaries connect to AS4 endpoints as servers: they do not pull messages from endpoints.
- Pull signals from AS4 endpoints target AS4 edge intermediaries and are not forwarded across an I-Cloud.
- An AS4 edge intermediary that is capable of delivering a particular user message to an AS4 endpoint SHOULD be configured to provide initial reverse routing of any related signals (receipts, errors).
- There is no requirement to support WS-ReliableMessaging sequence lifecycle messages.

4.3 Processing Mode Parameter

In this profile, AS4 processors either operate in peer-to-peer exchange mode or exchange messages across intermediaries based on the value of a single processing mode parameter, defined in section 6.4.2 of [ebMS3ADV]: **Pmode[1].Protocol.AddActorOrRoleAttribute**.

- If this value is set to *true* for a P-Mode, the ebMS header in AS4 user messages MUST have a SOAP 1.2 `role` attribute and its value MUST be set to the fixed value http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh.
- For AS4, the default value of this parameter is *false*, meaning that the SOAP 1.2 `role` attribute is not present. In SOAP 1.2, this is equivalent to the attribute being present with the value http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver.

4.4 AS4 Endpoint Requirements

The ebMS 3.0 multi-hop feature specifies requirements on endpoints to be able to exchange messages in an I-Cloud. This section further constrains these requirements and provides a minimal interoperability subset for AS4 endpoints. The structure of this section follows the structure of section 2.6 of [ebMS3ADV], which considers initiating messages and responding messages.

The section distinguishes three types of initiating messages:

- User Messages. No special processing is required of an AS4 processor, other than being able to insert the `role` attribute with the appropriate value, subject to the selected processing mode, as specified in section 4.3.
- ebMS Signal Messages. This AS4 profile constrains this further as follows:
    - No `ebint:RoutingInput` reference parameter and no `role` attribute are added to `eb:PullRequest` messages.
    - AS4 endpoints MUST NOT send initiating error messages.
- Non-ebMS Messages: this situation is not relevant in the case of AS4 as it does not require support for Web Services protocols like WS-ReliableMessaging [WSRM12]. For this reason there is no need to support initiating non-ebMS messages.

Section 2.6 of [ebMS3ADV] distinguishes the following type of responding messages:

- ebMS response User Messages. This is handled in the same way as ebMS request User Messages.
- ebMS Signal Messages. These messages are making use of WS-Addressing headers [WSADDRCORE] under certain conditions. This profile restricts or relaxes further the use of and/or support for these “wsa” headers.
    - AS4 endpoints are NOT REQUIRED to support `wsa:ReplyTo` header or `wsa:FaultTo` when generating responses.
    - If the user message that the signal relates to DOES NOT contain a `role` attribute with a value of http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh, processing of signals is as specified in the ebMS 3.0 Core Specification and in the other chapters of this specification.
    - If the user message that the signal relates to DOES contain a `role` attribute with a value of http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh, a response signal MUST contain
        - a `wsa:To` header element with value http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/icloud
        - a `wsa:Action` header element with value http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay.receipt or http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay.error
        - and a WS-Addressing reference parameter with content as specified in the subsection “Inferred RoutingInput for the reverse path” of section 2.6.2 of [ebMS3ADV]. The value of the MPC attribute is to be set based on the value of the MPC attribute in the user message. If that value is not set, the default value   http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultMPC is assumed (as defined in section 3.4.1 in [ebMS3CORE]):
            - ■ The MPC value for an AS4 receipt signal is formed by concatenating the string “`.receipt`” to the (default) MPC value of the received message.
            - ■ The MPC value for an AS4 error signal is formed by concatenating the string “`.error`” to the (default) MPC value of the message in error.

- Non-ebMS Messages: this situation is not relevant in the case of AS4, because AS4 does not require support for Web Services protocols that return signal messages, such as reliable messaging acknowledgments.

5 AS4 Usage Profile of ebMS 3.0 Core Specification

While the previous sections were describing messaging handler requirements for AS4 compliance (i.e. mostly intended for product developers), this section is about configuration and usage options.

This section is split in two major subsections:

- **AS4 Usage Rules**: this section provides the rules for using messaging features in an AS4-compliant way.
- **AS4 Usage Agreements**: this section provides notes to the users on the main options left open by the AS4 profiles, that have to be agreed on in order to interoperate.

Both sections are about features that are under responsibility of the user when using an AS4-compliant product.

5.1 AS4 Usage Rules

5.1.1 Core Components / Modules to be Used

This table summarizes which functional modules in the ebMS V3 specification are required to be implemented by the AS4 profile, and whether or not these modules are actually profiled for AS4.

ebMS V3 Component Name and Reference	**Profiling status**
Messaging Model (section 2)	Usage: **Required** Profiled: **Yes**
Message Pulling and Partitioning (section 3)	Usage: **Required** Profiled: **No** Notes**:** The profiling of QoS associated with Pulling is defined in another module. The MPC and pulling feature itself are not profiled.
Processing Modes (section 4)	Usage: **Required** Profiled: **Yes**
Message Packaging (section 5)	Usage: **Required** Profiled: **Yes** Notes: Default business process defines acceptable defaults for Role, Service and Action. Bundling options for message headers (piggybacking) are restricted.

ebMS V3 Component Name and Reference	**Profiling status**
Error Handling (section 6)	Usage: **Required** Profiled: **Yes** Notes: Addition of some new Error Codes regarding Reception Awareness
Security Module (section 7)	Usage: **Required** Profiled: **Yes** Notes: Guidance regarding which part(s) of the message may be encrypted and included in the signature. Further guidance on how to secure the `eb:PullRequest` Signal and the preventing of replay attacks..
Reliable Messaging Module (section 8)	Usage: **Not Required** Profiled: **No** Notes: This profile does not require the use of the Reliable Messaging Module using either WS-ReliableMessaging or WS-Reliability. It relies instead on `eb:Receipts` for supporting a light reliability feature called “Reception Awareness”.

713 5.1.2 Bundling rules

Scope of the Profile Feature	Defines bundling (or “piggybacking”) rules of ebMS MEPs, including Receipts.
Specification Feature	Message Packaging
Specification Reference	ebMS v3.0 Core Specification, Section 5.2.4.
Profiling Rule (a)	This profile supports the One-Way/Push MEP. Both synchronous and asynchronous transport channels for the response (`eb:Receipt`) are allowed by this profile.
Profiling Rule (b)	This profile supports the One-Way/Pull MEP. When sending a Receipt for this MEP, a Receiving MSH conforming to this profile MAY bundle the Receipt with any other ebMS message header (including an `eb:PullRequest` signal) or message body.

714 5.1.3 Security Element

Specification Feature	Use of WSS features
Specification Reference	ebMS v3.0 Core Specification, Section 7.1
Profiling Rule (a)	When using digital signatures or encryption, an AS4 MSH implementation is REQUIRED to use the Web Services Security X.509 Certificate Token Profile [WSS11-X509].

Alignment	35 17 *Web Services Security: SOAP Message Security 1.1*, 2005. [WSS11] 35 17 *Web Services Security X.509 Certificate Token Profile 1.1*, 2006 [WSS11-X509].

5.1.4 Signing Messages

Specification Feature	Digital Signatures for SOAP message headers and body
Specification Reference	ebMS v3.0 Core Specification, Section 7.2
Profiling Rule (a)	AS4 MSH implementations are REQUIRED to use Detached Signatures as defined by the XML Signature Specification [XMLDSIG] when signing AS4 user or signal messages. Enveloped Signatures as defined by [XMLDSIG] are not supported by or authorized in this profile.
Profiling Rule (b)	AS4 MSH implementations are REQUIRED to include the entire `eb:Messaging` SOAP header block and the (possibly empty) SOAP Body in the signature. The `eb:Messaging` header SHOULD be referenced using the "id" attribute.

5.1.5 Signing SOAP with Attachments Messages

Specification Feature	Signing attachments
Specification Reference	ebMS v3.0 Core Specification, Section 7.3
Profiling Rule (a)	AS4 MSH implementations are REQUIRED 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.
Profiling Rule (b)	AS4 MSH implementations are REQUIRED to include the entire `eb:Messaging` header block and all MIME body parts of included payloads in the signature.

5.1.6 Encrypting Messages

Specification Feature	Encrypting messages
Specification Reference	ebMS v3.0 Core Specification, Section 7.4
Profiling Rule (a)	If an AS4 user message is to be encrypted, AS4 MSH implementations MUST encrypt ALL payload parts. However, AS4 MSH implementations SHALL NOT encrypt the `eb:Messaging` header. If confidentiality of data in the `eb:Messaging` header is required, implementations SHOULD use transport level security.
Profiling Rule (b)	If an AS4 user message is to be encrypted and the user-specified payload data is to be packaged in the SOAP Body, AS4 MSH implementations are REQUIRED to encrypt the SOAP Body.

5.1.7 Encrypting SOAP with Attachments Messages

Specification Feature	Encryption of message attachments.
Specification Reference	ebMS v3.0 Core Specification, Section 7.5
Profiling Rule (a)	If an AS4 user message is to be encrypted and the user-specified payload data is to be packaged in conformance with the [SOAPATTACH] specification, AS4 MSH implementations are REQUIRED to encrypt the MIME Body parts of included payloads.

5.1.8 Generating Receipts

Specification Feature	`eb:Receipt` signal messages
Specification Reference	ebMS v3.0 Core Specification, Section 7.12..2 (Persistent Signed Receipt) ebMS v3.0 Core Specification, Section 5.2.3.3, `eb:Messaging/eb:SignalMessage/eb:Receipt`
Profiling Rule (a): Receipts for reception awareness	When a Receipt is to be used solely for reception awareness, the sender of the Receipt MUST contain a copy of the `eb:UserMessage` structure of the received AS4 message. The `eb:RefToMessageId` in the `eb:MessageInfo` group in the `eb:SignalMessage` contains the message identifier of the received message.
Profiling Rule (b): Receipts for Non Repudiation of Receipt (NRR)	When a Receipt is to be used for Non Repudiation of Receipt, the content of the `eb:Receipt` element MUST be a valid `ebbpsig:NonRepudiationInformation` element. When a Receipt is to be used for Non Repudiation of Receipt (NRR), the sender of the Receipt: - MUST use `ds:Reference` elements containing digests of the original message parts for which NRR is required. Message parts MUST NOT be identified using `ebbpsig:MessagePartIdentifier` elements. - MUST sign the AS4 receipt Signal Message. When signed receipts are requested in AS4 that make use of default conventions, the Sending message handler (i.e. the MSH sending messages for which signed receipts are expected) MUST identify message parts (referenced in `eb:PartInfo` elements in the received User Message) 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 . The Receiving message handler (i.e. the MSH generating the receipt signal) can reuse the `ds:Reference` elements from the `SignedInfo` reference list in the received message. Note that 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.
Profiling Rule (c)	An AS4 message that has been digitally signed MUST be acknowledged with a message containing an `eb:Receipt` signal that itself is digitally signed.

	The `eb:Receipt` MUST contain the information necessary to provide non-repudiation of receipt of the original message, as described in profiling rule (b). NOTE: the digest(s) to be inserted in the `ebbpsig: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.

5.1.9 MIME Header and Filename information

Specification Feature	Optional presence of a "filename" value in "Content-disposition" header on MIME body parts.
Specification Reference	MIME specification (IETF) [RFC2045]
Profiling Rule (a)	The "Content-disposition" header on MIME body parts, when used, MUST carry file name information. Implementations MUST support the setting (when sending) and reading (when receiving) of "Content-disposition" header,
Profiling Rule (b)	When end users wish to supply file names and have that information confidential, they SHOULD use TLS/SSL based encryption.

5.2 AS4 Usage Agreements

This section defines the operational aspect of the profile configuration aspects that users have to agree on, mode of operation, etc to interoperate. This section is not normative and is provided here only as guidance for users.

All the user agreement options related to a specific type of message exchange instance (e.g. related to a specific type of business transaction) are controlled by the Processing Mode (P-Mode) parameters defined in the ebMS Core V3 specification. This section only lists the parameters that are particularly relevant to AS4.

5.2.1 Controlling Content and Sending of Receipts

Scope of the Profile Feature	Choice among options in sending Receipts.
Specification Feature	`eb:Receipt` signal messages
Specification Reference	ebMS v3.0 Core Specification, Section 2.2
Usage Profiling (a)	Must `eb:Receipt` signals be used for non-repudiation of receipt (NRR), or just act as reception awareness feature? For non-repudiation, the `eb:Receipt` element must contain a well-formed `ebbpsig:NonRepudiationInformation` element. This is indicated by the new P-Mode parameter: **PMode[1].Security.SendReceipt.NonRepudiation :** value = 'true' (to be used for non-repudiation of receipt), value = 'false' (to be used simply for reception awareness).

Usage Profiling (b)	Receipts for One-Way/Push MEP: Both synchronous and asynchronous transport channels for the response `eb:Receipt` are allowed by this profile. (Values “Response” and “Callback”) This option is controlled by the P-Mode parameter: • **PMode[1].Security.SendReceipt.ReplyPattern:** value = ‘Response’ (sending receipts on the HTTP response or back-channel). • **PMode[1].Security.SendReceipt.ReplyPattern:** value = ‘Callback’ (sending receipts using a separate connection.)
Usage Profiling (c)	Receipts for the One-Way/Pull MEP: • **Pmode[1].Security.SendReceipt.ReplyPattern:** value = ‘Callback’ (sending receipts using a separate connection, and not bundled with an `eb:PullRequest`.)

5.2.2 Error Handling Options

Specification Feature	Error Handling options
Specification Reference	ebMS v3.0 Core Specification, chapter 6
Usage Profiling (a): Receiver-side error	All Receiver-side error reporting options are left for users to agree on, including the choice to not report at all: • **PMode[1].ErrorHandling.Report.ReceiverErrorsTo:** recommendation is to report such Receiver-side errors to the Sender. Otherwise: report URI that is different from sender URI? • **PMode[1].ErrorHandling.Report.AsResponse:** recommendation for one-way messages (except when pulling is in use) is value=”true”: report errors on the back-channel of erroneous messages. Errors for pulled messages can only be reported on a separate connection. • **PMode[1].ErrorHandling.Report.ProcessErrorNotifyConsumer:** (true / false) for controlling escalating the error to the application layer.
Usage Profiling (b): Reception Awareness errors	What is the behavior of a Sender that failed to receive a Receipt (even after message retries)? • No error reporting (in case no reception awareness required). • Error reporting from the Sender MSH to its message Producer (application-level notification). Error type: EBMS:0301: MissingReceipt (see Section 3.2 in Additional Features.) P-Mode parameter: • **PMode[1].ErrorHandling.Report.MissingReceiptNotifyProducer**: (new) true if (b), false if (a) • **PMode[1].ErrorHandling.Report.SenderErrorsTo**: (in case an error should be sent about such failures – e.g. to a third party if not to the original Receiver of the non-acknowledged user message.)
Usage Profiling (c):	How are errors about Receipt messages reported? P-Mode parameters:

Error about Receipts	• **PMode[1].ErrorHandling.Report.SenderErrorsTo:** reporting URI that is different from Receiver URI? • **PMode[1].ErrorHandling.Report.AsResponse:** (true / false) NOTE: In case of Receipts already sent over the HTTP back-channel, can only be "false" meaning such errors will be sent over separate connection. • **PMode[1].ErrorHandling.Report.ProcessErrorNotifyProducer:** (true / false) for controlling escalating the error to the application layer.

5.2.3 Securing the PullRequest

Specification Feature	Pulling authorization options
Specification Reference	ebMS v3.0 Core Specification, Section 7.11.x AS4 Conformance Profile authorization options (section 2.1.1)
Usage Profiling (a)	An AS4 Sending MSH MAY authenticate a Receiving MSH that sends an `eb:PullRequest` in two ways: 1. (Option 1 in 2.1.1) 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 [WSS11-UT]. 2. (Option 2 in 2.1.1) by using [WSS11-X509] coupled with the Message Partition Channel that a Pull signal is accessing for pulling messages. P-Mode parameters: • **PMode.Initiator.Authorization:** must be set to true (the initiator of a Pull request must be authorized). • **PMode.Initiator.Authorization.username:** (for option (a)) • **PMode.Initiator.Authorization.password:** (for option (a)) • **PMode[1].Security.PModeAuthorize:** must be set to true in the PMode leg describing the transfer of a pulled message. • **PMode[1].Security.X509.sign**: (for option (b)) • **PMode[1].Security.X509.SignatureCertificate**: (for option (b)) NOTE: in (b), the P-Mode parameters about X509 are controlling both the authentication of `eb:PullRequest` signals and authentication of other User Messages.
Usage Profiling (b)	`eb:PullRequest` signals: are they sent using the HTTPS transport protocol with optional Client-side Authentication? P-Mode parameter: • **PMode[1].Protocol.Address**: The URL scheme will indicate whether HTTPS is used or not.

5.2.4 Reception Awareness Parameters

Specification Feature	Message Retry and Duplicate Detection options
Specification Reference	AS4 Profile (this specification), AS4 Additional Features (section 3)

Usage Profiling (a): Sender options	In case Reception Awareness is used: what is the behavior of a Sender that did not receive a Receipt? (a) No message retry. (b) Resend the message. Retry parameters: to agree on: (1) retry count, (2) retry frequency P-Mode parameters (additional to those defined in ebMS Core V3): - **PMode[1].ReceptionAwareness:** (true / false) - **PMode[1].ReceptionAwareness.Retry:** (true / false) - **PMode[1].ReceptionAwareness.Retry.Parameters:** (contains a composite string specifying: (a) maximum number of retries or some timeout, (b) frequency of retries or some retry rule.
Usage Profiling (b): Receiver options	Is duplicate detection enabled? (a) No. Duplicates are not detected. (b) The receiver detects and eliminates duplicates based on `eb:MessageInfo/eb:MessageId`. P-Mode parameters (additional to those defined in ebMS Core V3): - **PMode[1].ReceptionAwareness.DuplicateDetection:** (true / false) - **PMode[1].ReceptionAwareness.DuplicateDetection.Parameters**

5.2.5 Default Values of Some P-Mode Parameters

Specification Feature	Default values and authorized values for main P-Mode parameters.
Specification Reference	ebMS v3.0 Core Specification, Appendix D.3
Usage Profiling (a)	**PMode.MEP** parameter will be constrained to the following value: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay
Usage Profiling (b)	**PMode.MEPbinding** parameter will be constrained to the following values: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/push http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/pull
Usage Profiling (c)	**PMode.Initiator.Role** parameter will have the following default value: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/initiator
Usage Profiling (d)	**PMode.Responder.Role** parameter will have the following default value: http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/responder
Usage Profiling (e)	**PMode[1].BusinessInfo.Service** parameter will have the following default

	value: http://docs.oasis-open.org/ebxml-msg/as4/200902/service *NOTE: this default is to be considered a P-Mode content default: absence of the P-Mode itself will cause the default value defined in the ebMS V3 Core specification (section 4.3) to apply. This value is usually enforced by the MSH implementation itself.*
Usage Profiling (f)	**PMode[1].BusinessInfo.Action** parameter will have the following default value: http://docs.oasis-open.org/ebxml-msg/as4/200902/action *NOTE: this default is to be considered a P-Mode content default: absence of the P-Mode itself will cause the default value defined in the ebMS V3 Core specification (section 4.3) to apply. This value is usually enforced by the MSH implementation itself*
Usage Profiling (g)	**PMode[1].Reliability** parameters are not supported by this profile

734 5.2.6 HTTP Confidentiality and Security

Specification Feature	HTTP Security Management and Options This table is intended as a guide for users, to specify their own agreements on HTTP confidentiality and security.
Specification Reference	ebMS v3.0 Core Specification, Section 7, Appendix D.3.6.
Usage Profiling (a)	Is HTTP transport-layer encryption required? What protocol version(s)?
Usage Profiling (b)	What encryption algorithm(s) and minimum key lengths are required?
Usage Profiling (c)	What Certificate Authorities are acceptable for server certificate authentication?
Usage Profiling (d)	Are direct-trust (self-signed) server certificates allowed?
Usage Profiling (e)	Is client-side certificate-based authentication allowed or required?
Usage Profiling (f)	What client Certificate Authorities are acceptable?
Usage Profiling (g)	What certificate verification policies and procedures must be followed?

735 5.2.7 Deployment and Processing requirements for CPAs

Usage Profile Feature	CPA Access
Usage Profiling (a)	Is a specific registry for storing CPAs required? If so, provide details.
Usage Profiling (b)	Is there a set of predefined CPA templates that can be used to create given

	Parties' CPAs?
Usage Profiling (c)	Is there a particular format for file names of CPAs, in case that file name is different from CPAId value?

5.2.8 Message Payload and Flow Profile

Usage Profile Feature	Message Quantitative Aspects
Usage Profiling (a)	What are typical and maximum message payload sizes that must be handled? (maximum, average)
Usage Profiling (b)	What are typical communication bandwidth and processing capabilities of an MSH for these Services?
Usage Profiling (c)	Expected Volume of Message flow (throughput): maximum (peak), average?
Usage Profiling (d)	How many Payload Containers must be present?
Usage Profiling (e)	What is the structure and content of each container? [List MIME Content-Types and other process-specific requirements.] Are there restrictions on the MIME types allowed for attachments?
Usage Profiling (f)	How is each container distinguished from the others? [By a fixed ordering of containers, a fixed Manifest ordering, or specific Content-ID values.]. Any expected relative order of attachments of various types?
Usage Profiling (g)	Is there an agreement that message part filenames must be present in MIME Content-Disposition parameter ?

5.2.9 Additional Deployment or Operational Requirements

Usage Profile Feature	Operational or Deployment Conditions
Usage Profiling (a)	Operational or deployment aspects that are object to further requirements or recommendations.

6 Conformance Clauses

This chapter defines five AS4 conformance clauses.

6.1 AS4 ebHandler Conformance Clause

In order to conform to the AS4 ebHandler Profile, an implementation must comply with all normative statements and requirements in Section 2.1.

In particular, it must:

- Observe all requirements stated as such in the Feature Set table of Section 2.1.1.
- Comply with WS-I requirements listed in Section 2.1.2.
- Support the P-Mode parameters as required in Section 2.1.3.

In addition, the implementation MUST implement all of the additional features as indicated in Section 3, including the alternative Pull authorization mode, except for the sub-channel feature which remains optional in its implementation.

Finally, the implementation MUST support the Usage Rules defined in Section 5.1 .

The Usage Agreements in Section 5.2 are not prescriptive, and implementations are free to support any subset of the features described that are not already mandated in sections 2.1, 3 or 5.1 .

6.2 AS4 Light Client Conformance Clause

In order to conform to the AS4 Light Client Profile, an implementation MUST comply with all normative statements and requirements in Section 2.2.

In particular, it must:

- Observe all requirements stated as such in the Feature Set table of Section 2.2.1.
- Comply with WS-I requirements listed in Section 2.2.2.
- Support the P-Mode parameters as required in Section 2.2.3.

In addition, the implementation must implement all of the additional features as indicated in Section 3, including the alternative Pull authorization mode, with the following exception: the sub-channel feature (as a Receiving MSH) which remains optional.

Finally, the implementation must support the Usage Rules defined in Section 5.1 .

The Usage Agreements in Section 5.2 are not prescriptive, and implementations are free to support any subset of the features described that are not already mandated in sections 2.2, 3 or 5.1 .

6.3 AS4 Minimal Client Conformance Clause

In order to conform to the AS4 Minimal Client Profile, an implementation MUST comply with all normative statements and requirements in Section 2.3.

In particular, it must:

- Observe all requirements stated as such in the Feature Set table of Section 2.3.1.
- Comply with WS-I requirements listed in Section 2.3.2.

- Support the P-Mode parameters as required in Section 2.3.3.

The implementation does not have to implement any of the additional features as indicated in Section 3.

Finally, the implementation must support only the Usage Rules defined in Section 5.1 that pertain to its required features (2.3.1).

6.4 AS4 Minimal Sender Conformance Clause

In order to conform to the AS4 Minimal Sender Profile, an implementation MUST conform to the AS4 Minimal Client Conformance Clause as defined in section 6.3 with the following exceptions:

- Support for One Way / Pull messages is NOT REQUIRED.
- Support for WS-Security is NOT REQUIRED (as it is only used in the Minimal Client CP to secure Pull requests, which are not used by a Minimal Sender).

The implementation does not have to implement any of the additional features as indicated in Section 3.

This conformance clause is intended for very light devices producing AS4 user messages but with no requirement to receive and process AS4 user messages - for example, monitoring devices.

6.5 AS2/AS4 ebHandler Conformance Clause

In order to conform to the AS2/AS4 ebHandler Profile, an implementation MUST, in addition to supporting AS4 message exchanges that comply with all normative statements and requirements specified in section 6.1 , also conform to the EDIINT Applicability Statement 2 (AS2, [RFC4130]).

6.6 AS4 Multi-Hop Endpoint Conformance Clause

In AS4, support for the multi-hop feature of ebMS 3.0 Part 2 is optional. In order to conform to the AS4 Multi-Hop Endpoint Conformance Clause, an implementation MUST conform to:

- All normative statements and requirements specified in section 4.
- At least one of the other conformance clauses (AS4 ebHandler Conformance Clause, AS4 Light Client Conformance Clause, AS4 Minimal Client Conformance Clause, AS4 Minimal Sender Conformance Clause or the AS2/AS4 ebHandler Conformance Clause).

Appendix A Sample Messages

This appendix contains examples of:

- an AS4 user message;
- an AS4 user message with a compressed attachment;
- AS4 receipts providing Non-Repudiation of Receipt (NRR);
- an AS4 Pull Request message signal.

Appendix A.1 User Message

The following example contains the SOAP envelope of an AS4 message from a Seller to a Buyer to exchange an electronic invoice document. Both parties are identified using the GS1 global location numbers [GLN] encoded using the ebCore Party Id type notation [ebCorePartyId].The XML business document is an XML document (only the root element is displayed) based on the version 2.0 UN/CEFACT Cross-Industry Invoice schema [CII]. The business document is contained in the SOAP body. The values of `eb:Service` and `eb:Action` adopt the AS4 default values. The message is secured using a WS-Security header, details of which are omitted. This AS4 SOAP envelope is included in a SOAP-with-attachment container, which is also not shown here.

```
<S12:Envelope
    xmlns:S12="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
    xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
    xmlns:eb="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/" >
    <S12:Header>
        <eb:Messaging S12:mustUnderstand="true" id="_9ecb9d3c-cef8-4006-ac18-f425c5c7ae3d">
            <eb:UserMessage>
                <eb:MessageInfo>
                    <eb:Timestamp>2011-04-03T14:49:28.886Z</eb:Timestamp>
                    <eb:MessageId>2011-921@5209999001264.example.com</eb:MessageId>
                </eb:MessageInfo>
                <eb:PartyInfo>
                    <eb:From>
                        <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088"
                            >5209999001264</eb:PartyId>
                        <eb:Role>Seller</eb:Role>
                    </eb:From>
                    <eb:To>
                        <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088"
                            >5209999001295</eb:PartyId>
                        <eb:Role>Buyer</eb:Role>
                    </eb:To>
                </eb:PartyInfo>
                <eb:CollaborationInfo>
                    <eb:Service>http://docs.oasis-open.org/ebxml-msg/as4/200902/service</eb:Service>
                    <eb:Action>http://docs.oasis-open.org/ebxml-msg/as4/200902/action</eb:Action>
                    <eb:ConversationId>2011-921</eb:ConversationId>
                </eb:CollaborationInfo>
                <eb:PayloadInfo>
                    <eb:PartInfo href="#_f8aa8b55-b31c-4364-94d0-3615ca65aa40"/>
                </eb:PayloadInfo>
            </eb:UserMessage>
        </eb:Messaging>
        <wsse:Security S12:mustUnderStand="true">
            <!-- Content omitted -->
        </wsse:Security>
    </S12:Header>
    <S12:Body wsu:Id="_f8aa8b55-b31c-4364-94d0-3615ca65aa40">
        <CrossIndustryInvoice xmlns="urn:un:unece:uncefact:data:standard:CrossIndustryInvoice:2">
            <!-- content omitted -->
        </CrossIndustryInvoice>
    </S12:Body>
</S12:Envelope>
```

Appendix A.2 User Message with Compressed Payload

The following example illustrates a typical user message as above but with a compressed attachment. The SOAP-with-attachments container is shown for the compressed attachment as included in the `eb:PayloadInfo` element of the user message.

```
Content-Type: Multipart/Related; boundary=MIME_boundary; type=application/soap+xml;
        start="<as4msg@example.com>"

--MIME_boundary
Content-Type: application/soap+xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <as4msg@example.com>

<?xml version="1.0" encoding="UTF-8"?>
<S12:Envelope
    xmlns:S12="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
    xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
    xmlns:eb="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/" >
    <S12:Header>
        <eb:Messaging S12:mustUnderstand="true" id="_9ecb9d3c-cef8-4006-ac18-f425c5c7ae3d">
            <eb:UserMessage>
                <eb:MessageInfo>
                    <eb:Timestamp>2011-04-03T14:49:28.886Z</eb:Timestamp>
                    <eb:MessageId>2011-921@5209999001264.example.com</eb:MessageId>
                </eb:MessageInfo>
                <eb:PartyInfo>
                    <eb:From>
                        <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088"
                            >5209999001264</eb:PartyId>
                        <eb:Role>Seller</eb:Role>
                    </eb:From>
                    <eb:To>
                        <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088"
                            >5209999001295</eb:PartyId>
                        <eb:Role>Buyer</eb:Role>
                    </eb:To>
                </eb:PartyInfo>
                <eb:CollaborationInfo>
                    <eb:Service>http://docs.oasis-open.org/ebxml-msg/as4/200902/service</eb:Service>
                    <eb:Action>http://docs.oasis-open.org/ebxml-msg/as4/200902/action</eb:Action>
                    <eb:ConversationId>2011-921</eb:ConversationId>
                </eb:CollaborationInfo>
                <eb:PayloadInfo>
                    <eb:PartInfo href="cid:attachment1234@example.com">
                        <eb:PartProperties>
                            <eb:Property name="MimeType">application/xml<eb:Property>
                            <eb:Property name="CharacterSet">utf-8</eb:Property>
                            <eb:Property name="CompressionType">application/gzip</eb:Property>
                        </eb:PartProperties>
                    </eb:PartInfo>
                </eb:PayloadInfo>
            </eb:UserMessage>
        </eb:Messaging>
        <wsse:Security S12:mustUnderStand="true">
            <!-- Content omitted -->
        </wsse:Security>
    </S12:Header>
    <S12:Body wsu:Id="_f8aa8b55-b31c-4364-94d0-3615ca65aa40" />
</S12:Envelope>

--MIME_boundary
Content-Type: application/gzip
Content-ID: <attachment1234@example.com>
Content-Description: Compressed XML payload
Content-Disposition: attachment; filename=invoice.xml
Content-Transfer-Encoding: binary

#‹##°B.O#ÿ <!-- remaining binary content omitted -->

--MIME_boundary
```

Appendix A.3 Non-Repudiation of Receipt

When the `ebbpsig:NonRepudiationInformation` element is used in an `eb:Receipt`, it contains a sequence of `ebbpsig:MessagePartNRInformation` items for each message part for which evidence of non repudiation of receipt is being provided. In the normal default usage, these message parts are those that have been signed in the original message. Each message part is described with information defined by an XML Digital Signature Reference information item. The following example illustrates the ebMS V3 Signal Message header.

```
<eb3:Messaging S12:mustUnderstand="true" id="ValueOfMessagingHeader">
  <eb3:SignalMessage>
    <eb3:MessageInfo>
       <eb3:Timestamp>2009-11-06T08:00:09Z</eb3:Timestamp>
       <eb3:MessageId>orderreceipt@seller.com</eb3:MessageId>
        <eb3:RefToMessageId>orders123@buyer.com</eb3:RefToMessageId>
    </eb3:MessageInfo>
    <eb3:Receipt>
      <ebbp:NonRepudiationInformation>
        <ebbp:MessagePartNRInformation>
          <dsig:Reference URI="#5cb44655-5720-4cf4-a772-19cd480b0ad4">
            <dsig:Transforms>
               <dsig:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            </dsig:Transforms>
            <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
               <dsig:DigestValue>o9QDCwWSiGVQACEsJH5nqkVE2s0=</dsig:DigestValue>
          </dsig:Reference>
     </ebbp:MessagePartNRInformation>
        <ebbp:MessagePartNRInformation>
          <dsig:Reference URI="cid:a1d7fdf5-d67e-403a-ad92-3b9deff25d43@buyer.com">
            <dsig:Transforms>
               <dsig:Transform
                  Algorithm="http://docs.oasis-open.org/wss/oasis-wss-SwAProfile-1.1#Attachment-
Content-Signature-Transform" />
               </dsig:Transforms>
               <dsig:DigestMethod
                  Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
               <dsig:DigestValue>iWNSv2W6SxbOYZliPzZDcXAxrwI=</dsig:DigestValue>
          </dsig:Reference>
     </ebbp:MessagePartNRInformation>
        </ebbp:NonRepudiationInformation>
    </eb3:Receipt>
  </eb3:SignalMessage>
</eb3:Messaging>
```

For a signed receipt, a Web Services Security header signing over the signal header (and other elements as specified in sections 5.1.4 and 5.1.5) is required. An example WS-Security header is as follows:

```
<wsse:Security S12:mustUnderstand="true">
   <wsu:Timestamp wsu:Id="_1">
               <wsu:Created>2009-11-06T08:00:10Z</wsu:Created>
               <wsu:Expires>2009-11-06T08:50:00Z</wsu:Expires>
        </wsu:Timestamp>
        <wsse:BinarySecurityToken
    EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-
1.0#Base64Binary"
    ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-
1.0#X509v3"
    wsu:Id="_2">MIIFADCCBGmgAwIBAgIEOmitted</wsse:BinarySecurityToken>
        <ds:Signature Id="_3">
               <ds:SignedInfo>
                       <ds:CanonicalizationMethod
             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
                       <ds:SignatureMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
                       <ds:Reference URI="#ValueOfMessagingHeader">
                              <ds:Transforms>
                                      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#">
                                      <InclusiveNamespaces PrefixList="xsd"
                    xmlns="http://www.w3.org/2001/10/xml-exc-c14n#" />
                                      </ds:Transform>
                              </ds:Transforms>
                              <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
/>
```

```
                              <ds:DigestValue>ZXnOmitted=</ds:DigestValue>
                      </ds:Reference>
          <!-- Omitted other reference elements for other signed parts -->
               </ds:SignedInfo>
               <ds:SignatureValue>rxaP4of8JCpUkOmitted=</ds:SignatureValue>
               <ds:KeyInfo>
                      <wsse:SecurityTokenReference>
            <wsse:Reference URI="#_2"
              ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
profile-1.0#X509v3" />
                      </wsse:SecurityTokenReference>
               </ds:KeyInfo>
       </ds:Signature>
</wsse:Security>
```

Appendix A.4 Pull Request Signal Message

The following example shows an AS4 Pull Request Signal on a particular message partition channel. The message contains two WS-Security headers:

1. The first WS-Security header is targeted to the “ebms” role, and is used for authorization of access to the pull channel. This header is added to the message before the second WS-Security header.
2. A second WS-Security header is used to protect the signal message itself. This header is added to the message after the authorization header, and signs this authorization header, the ebMS Messaging header and the (empty) SOAP Body element.

```
<S12:Envelope xmlns:S12="http://www.w3.org/2003/05/soap-envelope"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:eb3="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/"
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd"
    xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd">
    <S12:Header>
        <eb3:Messaging S12:mustUnderstand="true" id='_ebmessaging' >
            <eb3:SignalMessage>
                <eb3:MessageInfo>
                    <eb3:Timestamp>2011-02-19T11:30:11.320Z</eb3:Timestamp>
                    <eb3:MessageId>msg123@smallco.example.com</eb3:MessageId>
                </eb3:MessageInfo>
                <eb3:PullRequest mpc="http://as4.bigco.example.com/queues/q_456" />
            </eb3:SignalMessage>
        </eb3:Messaging>
        <wsse:Security S12:role="ebms" S12:mustUnderstand="true" wsu:Id="_pullauthorization">
            <wsse:UsernameToken>
                <wsse:Username>smallcoAS4</wsse:Username>
                <wsse:Password
                    Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-
profile-1.0#PasswordDigest"
                    >B5twk47KwSrjeg==</wsse:Password>
                <wsu:Created>2011-02-19T11:30:11.327Z</wsu:Created>
            </wsse:UsernameToken>
        </wsse:Security>
        <wsse:Security S12:mustUnderStand="true">
            <wsse:BinarySecurityToken wsu:Id="_smallco_cert">
                <!-- details omitted -->
            </wsse:BinarySecurityToken>
            <ds:Signature>
                <ds:SignedInfo>
                    <ds:CanonicalizationMethod
                        Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                    <ds:SignatureMethod
                        Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                    <ds:Reference URI="#_ebmessaging">
                        <ds:Transforms>
```

```
                            <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                        </ds:Transforms>
                        <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmlds#sha1"/>
                        <ds:DigestValue>KshAH7QFFAw2sV5LQBOUOSSrCaI=</ds:DigestValue>
                    </ds:Reference>
                    <ds:Reference URI="#_pullauthorization">
                        <ds:Transforms>
                            <ds:Transform
                                Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                        </ds:Transforms>
                        <ds:DigestMethod
                                Algorithm="http://www.w3.org/2000/09/xmlds#sha1"/>
                        <ds:DigestValue>PreCqm0ESZqmITjf1qzrLFuOEYg=</ds:DigestValue>
                    </ds:Reference>
                    <ds:Reference URI="#_soapbody">
                        <ds:Transforms>
                            <ds:Transform
                                Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                        </ds:Transforms>
                        <ds:DigestMethod
                                Algorithm="http://www.w3.org/2000/09/xmlds#sha1"/>
                        <ds:DigestValue>FkwnI8mmXh71J5qcwO404ZnlXpg=</ds:DigestValue>
                    </ds:Reference>
                </ds:SignedInfo>
                <ds:SignatureValue>
                    <!-- details omitted -->
                </ds:SignatureValue>
                <ds:KeyInfo>
                    <wsse:SecurityTokenReference>
                        <wsse:Reference URI="#_smallco_cert"
                            ValueType="http://docs.oasisopen.org/wss/2004/01/oasis-200401-wss-
x509-token-profile-1.0#X509v3"
                        />
                    </wsse:SecurityTokenReference>
                </ds:KeyInfo>
            </ds:Signature>
        </wsse:Security>
    </S12:Header>
    <S12:Body wsu:Id="_soapbody" />
</S12:Envelope>
```

Appendix B Generating an AS4 Receipt

The following XSLT 1.0 stylesheet generates an AS4 Receipt message from an AS4 message, as specified in section 4.4 . The stylesheet supports signed messages for which the **Pmode[1].Security.SendReceipt.NonRepudiation** is set to true. It could be used in an AS4 MSH after a WS-Security module has verified the `wsse:Security` header in the user message, allowing the reuse of `ds:Reference` elements in the user message in the AS4 `eb:Receipt`.

Note that this section is non-normative: AS4 implementations are not required to use this (or any other) XSLT stylesheet to generate receipts for user messages.

The stylesheet handles both the peer-to-peer, direct exchange (based on AS4 profiling of [ebMS3CORE]) and indirect exchange through an I-Cloud (based on AS4 profiling of [ebMS3ADV]). The generation of `ebint:RoutingInput` structures supports default MPC values in the user messages.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xd="http://www.oxygenxml.com/ns/doc/xsl" exclude-result-prefixes="xd xsi" version="1.0"
    xmlns:S12="http://www.w3.org/2003/05/soap-envelope"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:ebint="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/multihop/200902/"
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
    xmlns:eb3="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:ebbp="http://docs.oasis-open.org/ebxml-bp/ebbp-signals-2.0"
    xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
    <xd:doc scope="stylesheet">
        <xd:desc>
            <xd:p><xd:b>Created on:</xd:b> Feb 5, 2012</xd:p>
            <xd:p><xd:b>Author:</xd:b> pvde</xd:p>
            <xd:p>This XSLT stylesheet is a non-normative part of the OASIS AS4 specification. It
                shows how AS4 messages can be derived from AS4 user messages.</xd:p>
        </xd:desc>
        <xd:param name="messageid">
            <xd:p>The messageid to use on the AS4 receipt signal message</xd:p>
        </xd:param>
        <xd:param name="timestamp">
            <xd:p>The timestamp to set on the AS4 receipt signal message</xd:p>
        </xd:param>
    </xd:doc>

    <xsl:output method="xml" indent="yes"/>

    <xsl:param name="messageid">someuniqueid@receiver.example.com</xsl:param>
    <xsl:param name="timestamp">2012-02-05T19:43:11.735Z</xsl:param>

    <xsl:template match="S12:Envelope">
        <S12:Envelope>
            <xsl:apply-templates/>
        </S12:Envelope>
    </xsl:template>

    <xsl:template match="S12:Header">
        <S12:Header>
            <xsl:apply-templates select="eb3:Messaging"/>
        </S12:Header>
    </xsl:template>

    <xd:doc>
        <xd:desc>When generating a receipt for a signed message, the receipt will be signed as well.
            We generate an identifier for the empty SOAP Body of the AS4 receipt for the WS-Security
            module.</xd:desc>
    </xd:doc>
    <xsl:template match="S12:Envelope[S12:Header//ds:Signature]/S12:Body">
        <S12:Body wsu:Id="{generate-id()}"/>
    </xsl:template>

    <xd:doc>
        <xd:desc>The empty body of receipt signal receipt for an unsigned message does not need an
```

```
          identifier</xd:desc>
    </xd:doc>
    <xsl:template match="S12:Envelope[not(S12:Header//ds:Signature)]/S12:Body">
        <S12:Body/>
    </xsl:template>

    <xd:doc>
        <xd:desc>There are two templates for <xd:i>eb3:Messaging</xd:i> element. This first template
            is for an AS4 user message that may have been exchanged over a multi-hop network. The
            receipt for such a message has WS-Addressing headers and a routing parameter based on the
            user message content.</xd:desc>
    </xd:doc>
    <xsl:template
        match="eb3:Messaging[
        @S12:role='http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh']">
        <xsl:variable name="mpc">
            <xsl:choose>
                <xsl:when test="descendant::eb3:UserMessage[1]/@mpc">
                    <xsl:value-of select="descendant::eb3:UserMessage[1]/@mpc"/>
                </xsl:when>
                <xsl:otherwise>http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC</xsl:otherwise>
            </xsl:choose>
        </xsl:variable>
        <wsa:To wsu:Id="{concat('_wsato_',generate-id())}"
            S12:role="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh"
            S12:mustUnderstand="true"
            >http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/icloud</wsa:To>
        <wsa:Action wsu:Id="{concat('_wsaaction_',generate-id())}"
            >http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay.receipt</wsa:Action>
        <ebint:RoutingInput wsa:IsReferenceParameter="true"
            id="{concat('_ebroutinginput_',generate-id())}" S12:mustUnderstand="true"
            S12:role="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh">
            <ebint:UserMessage mpc="{concat($mpc,'.receipt')}">
                <eb3:PartyInfo>
                    <eb3:From>
                        <xsl:copy-of select="descendant::eb3:UserMessage[1]//eb3:To/eb3:PartyId"/>
                        <xsl:copy-of select="descendant::eb3:UserMessage[1]//eb3:To/eb3:Role"/>
                    </eb3:From>
                    <eb3:To>
                        <xsl:copy-of select="descendant::eb3:UserMessage[1]//eb3:From/eb3:PartyId"/>
                        <xsl:copy-of select="descendant::eb3:UserMessage[1]//eb3:From/eb3:Role"/>
                    </eb3:To>
                </eb3:PartyInfo>
                <eb3:CollaborationInfo>
                    <xsl:copy-of select="descendant::eb3:UserMessage[1]//eb3:Service"/>
                    <eb3:Action>
                        <xsl:value-of
                            select="concat(descendant::eb3:UserMessage[1]//eb3:Action,'.receipt')"
                        />
                    </eb3:Action>
                    <xsl:copy-of select="descendant::eb3:UserMessage[1]//eb3:ConversationId"/>
                </eb3:CollaborationInfo>
            </ebint:UserMessage>
        </ebint:RoutingInput>
        <eb3:Messaging S12:mustUnderstand="true" id="{concat('_ebmessaging_',generate-id())}">
            <xsl:apply-templates select="descendant-or-self::eb3:UserMessage"/>
        </eb3:Messaging>
    </xsl:template>

    <xd:doc>
        <xd:desc>This second template for the <xd:i>eb3:Messaging</xd:i> element covers AS4
            point-to-point messages.</xd:desc>
    </xd:doc>
    <xsl:template
        match="eb3:Messaging[not(
        @S12:role='http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/part2/200811/nextmsh')]">
        <eb3:Messaging S12:mustUnderstand="true" id="{concat('_ebmessaging_',generate-id())}">
            <xsl:apply-templates select="descendant-or-self::eb3:UserMessage"/>
        </eb3:Messaging>
    </xsl:template>

    <xd:doc>
        <xd:desc>
            <xd:p>The AS4 receipt is generated based on <xd:i>eb3:UserMessage</xd:i> and
                    <xd:i>ds:Signature</xd:i>content.</xd:p>
            <xd:ul>
                <xd:li>A receipt for a signed AS4 message references the message parts using
                        <xd:i>ds:Reference</xd:i>s in the WS-Security header of that message</xd:li>
                <xd:li>A receipt for an unsigned AS4 message references the message using the
                        <xd:i>eb3:UserMessage</xd:i>s of the AS4 message.
                </xd:li>
            </xd:ul>
        </xd:desc>
    </xd:doc>
    <xsl:template match="eb3:UserMessage">
        <eb3:SignalMessage>
            <eb3:MessageInfo>
                <eb3:Timestamp>
                    <xsl:value-of select="$timestamp"/>
                </eb3:Timestamp>
                <eb3:MessageId>
                    <xsl:value-of select="$messageid"/>
                </eb3:MessageId>
                <eb3:RefToMessageId>
                    <xsl:value-of select="descendant::eb3:MessageId"/>
                </eb3:RefToMessageId>
            </eb3:MessageInfo>
            <eb3:Receipt>
                <xsl:choose>
                    <xsl:when test="/S12:Envelope/S12:Header/wsse:Security/ds:Signature">
                        <ebbp:NonRepudiationInformation>
                            <xsl:apply-templates select="//ds:Reference"/>
                        </ebbp:NonRepudiationInformation>
                    </xsl:when>
                    <xsl:otherwise>
                        <xsl:copy-of select="//eb3:UserMessage"/>
                    </xsl:otherwise>
                </xsl:choose>
            </eb3:Receipt>
        </eb3:SignalMessage>
    </xsl:template>

    <xsl:template match="ds:Reference">
        <ebbp:MessagePartNRInformation>
            <xsl:copy-of select="current()"/>
        </ebbp:MessagePartNRInformation>
    </xsl:template>

</xsl:stylesheet>
```

Appendix C Acknowledgments

The following individuals were members of the committee during the development of this specification or of a previous version of it:

- Roger Bass, Traxian <roger@traxian.com>
- Kenneth Bengtson, Alpha1Lab <kenneth@alfa1lab.com>
- Timothy Bennett, Drummond Group Inc. <timothy@drummondgroup.com>
- Weisin Chong, Cisco Systems
- Jacques Durand, Fujitsu America Inc. <jdurand@us.fujitsu.com>
- Richard Emery, Axway Software <remery@us.axway.com>
- Ian Jones, British Telecommunications plc <ian.c.jones@bt.com>
- Sander Fieten, Individual <sander@fieten-it.com>
- Kazunori Iwasa, Fujitsu Limited <kiwasa@jp.fujitsu.com>
- Theo Kramer, Flame Computing Enterprises <theo@flame.co.za>
- Dale Moberg, Axway Software <dmoberg@axway.com>
- Farrukh Najmi, Wellfleet Software Corporation <farrukh@wellfleetsoftware.com>
- Makesh Rao, Cisco Systems, Inc. <marao@cisco.com>
- Sven Rasmussen, Denmark Ministry of Science, Technology & Innovation <svrr@itst.dk>
- Akihisa Sako, Axway Software <asako@us.axway.com>
- Pim van der Eijk, Sonnenglanz Consulting <pvde@sonnenglanz.net>
- Ernst Jan van Nigtevecht <ejvn@sonnenglanz.net>
- John Voss, Cisco Systems, Inc. <jovoss@cisco.com>

Appendix D Revision History

Rev	Date	By Whom	What
	25 Jul 2008	J. Durand / T. Bennett	Initial draft
Rev 02	28 Oct 2008	J. Durand	candidate CD draft
Rev 03	15 Feb 2009	J. Durand	Various edits, updates on Receipts, Message samples.
CD 2	10/03/09	J. Durand	CD 2 draft for PR
CS 01	04/24/10	J. Durand	Document voted Committee Specification 01
Rev 06	02/22/11	J. Durand / P. van der Eijk	CSD 3 draft for PR: Many minor editorial updates and clarifications; updated references; new sections 2.2.3 and A.2.
CSD 03	02/23/11	P. van der Eijk	Document approved as CSD 03 on 2011-02-23 http://www.oasis-open.org/apps/org/workgroup/ebxml-msg/download.php/41302/MessagingTC022311.htm
WD 8	03/28/11	J. Durand / T. Kramer	Follow-up on Theo comments; normalized PMode name as "P-Mode", when in plain text. 2.1.3.1 and 2.2.3.1: made support "required" for PMode.ID and PMode.agreement (meaning an implementation must be able to use this Pmode value - if present - to fill-in the related message header element.)
WD 9	04/04/11	P. van der Eijk	Updated revision history and frontpage; suppressed line numbering in footers. Renamed some references to ebMS3 to “ebMS3 Core”. New optional profiling of the ebMS3, Part 2 multi-hop feature; New sample user message in appendix A. New Appendix B, Generating an AS4 Receipt . In Acknowledgments, names are ordered alphabetically by last name.
WD 10	04/11/11	P. van der Eijk	Improved language in section 4 (comment made by Theo), A.1 and B. In sample user message, added an id attribute to eb:Messaging (as it would need one to be signed). Appendix A.3, fixed a hash value. (The values are illustrative only but should be different).
WD 11	04/12/11	P. van der Eijk	Improved sample message (added missing `S12:mustUnderstand` attribute). Removed requirement to pass receipts

Rev	Date	By Whom	What
			to applications.
WD 12	04/20/11	P. van der Eijk	Fixed bad reference in 6.6 Fixed two affiliations
WD 13 / WD 14	04/22/11	P. van der Eijk	Fixed citations and front matter.
WD 15	05/09/11	P. van der Eijk	Update for message format of receipts for unsigned messages, supporting “reception awareness”. Section 3.2 added clarification that reception awareness requires sending of receipts.
WD 16	05/16/11	P. van der Eijk / Jacques Durand	Discussion of receipts for messages without `PayloadInfo`. Fixed some section reference numbers and missing references. Many minor textual improvements. Part 2 profiling as “complementary” to a “primary” profiling of Part 1.
WD 17	05/18/11	P. van der Eijk	Simplified Encryption, ebMS header is never encrypted (section 5.1.6) Added note on “id” attribute in section 5.1.4 .
CS 02	08/13/11	TC Admin	Document approved as Committee Specification after Public Review.
WD 18	11/10/11	P. van der Eijk	Fixed a bad reference to SOAP 1.1 in section 2.2.1 . In 3.1 compression, the original MIME type of a compressed payload is required instead of just being recommended; added clarification on use of CharacterSet in compression. Minor editorial changes and some comments on areas that may need clarification.
WD 19	17-11-11	F2F Meeting	2.1.1: Changed the wording to reflect that a conforming MSH must support both MEPs as Initiator and Responder. 2.1.1 and 2.2.1: Added a note that MSHs may also support Two-Way MEPs 3.1: Changed name of `PartInfo` attribute used for indicating a compressed payload. 5.1.8: Changed requirements on the eb:Receipt element for messages without attachments.
WD 20	12/05/12	J. Durand	Compression property cannot be empty string for XML schema compliance. MIME type value in property for a compressed attachment required instead of recommended. Added note that a compressed payload must be in a separate MIME part and not in the SOAP Body

Rev	Date	By Whom	What
			In AS4 5.1.8.(a) and 5.1.8.(b), clarified use of receipts with simple SOAP messages, where the SOAP envelope is not in a part with a content identifier, and has no MIME content ID, so here there can be no part identifier. In AS4 5.1.8.(a) and 5.1.8.(b), note that it is impossible to generate a valid ebBP reception awareness. Proposal on support for MPC sub-channels. Clarified support for advanced features in conformance clauses.
WD 21	12/23/12	J. Durand	Added some Receipt-related errors. Compression property renamed to `CompressionType`.
WD 22	01/22/12	J. Durand	Minor fixes in 2.1.1 and 2.2.1. New compression-related errors. Clarification in the introductory paragraph of 2.2 that a light clients cannot connect to another light client to pull or push messages.
WD 23	01/31/12	T. Kramer	Appendix 2: Added Example on User Message with Compressed Attachment. Bumped original Appendix 2 and 3 to 3 and 4. 3.1: Reworded
WD 24	02/05/12	P. van der Eijk	Updated compression example in A2. Updated reference to Part 2 to CS. Renamed section 3.6 to Additional Features Errors as it covers not just receipts but also compression. Updated XSLT stylesheet in appendix B for reception awareness; it now uses the `UserMessage`. Formatting (use of element and attribute styles). In section 3.4, incorporated Jacques' proposed rewording on semantics of receipt. SOAP-with-attachments not required for Minimal Client.
WD 25	02/09/12	J. Durand / P. van der Eijk	The minimal Client Conformance Profile now has its own subsection in chapter 2. In chapter 6, there are two Conformance Clauses for the Minimal Client: the Minimal Client conformance clause and the Minimal Sender conformance clause. The Minimal Sender does not support Pull, and therefore never requires WS-Security UserName Tokens, unlike the Minimal Client. Added SSL/TLS recommendation. Added some sentences on Minimal Profile to Abstract and Introduction. Added a sentence in 2.1 to introduce the ebHandler profile.

Rev	Date	By Whom	What
			Removed empty/unused rows in section 5. In the references section, the reference to XML 1.0 now points to the latest version. Updated Appendix C to current roster. Some formatting (style consistency, prefix consistency).
WD 25	03/20/12	Makesh Rao	Accepted all changes and corrected the citation.