AS4 Profile of ebMS 3.0 Version 1.0

Committee Specification Draft 03

23 February 2011

Specification URIs:

This Version:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/csd03/AS4-profile-csd03.pdf (Authoritative)

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/csd03/AS4-profile-csd03.html

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/csd03/AS4-profile-csd03.odt

Previous Version:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/AS4-profile-cs-01.pdf (Authoritative)

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/AS4-profile-cs-01.html

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/AS4-profile-cs-01.odt

Latest Version:

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/AS4-profile.pdf (Authoritative)

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/AS4-profile.html

http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/AS4-profile.odt

Technical Committee:

OASIS ebXML Messaging Services TC

Chair:

Ian Jones, British Telecommunications plc

Editor:

Jacques Durand, Fujitsu Computer Systems
Pim van der Eijk, Sonnenglanz Consulting

Related Work:

This specification is related to:

- OASIS ebXML Messaging Services Version 3.0: Part 1, Core Specification

Declared XML Namespace:

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

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. This document defines the AS4 profile as a combination of a conformance profile that concerns 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 and the AS4 Light Client profile - that reflect different endpoint capabilities.

Status:

This document was last revised or approved by the OASIS ebXML Messaging Services TC 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 specification 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 specification, 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 specification the following citation format should be used:

[AS4-Profile-v1.0]

AS4 Profile of ebMS 3.0 Version 1.0. 23 February 2011. OASIS Committee Specification Draft 03. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/csd03/AS4-profile-csd03.pdf.

Notices

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 names "OASIS", “ebXML”, “ebXML Messaging Services”, and “ebMS” are trademarks 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 http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

1 Introduction

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 [ebMS3] 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 fundamental Web Services standards SOAP 1.1 [SOAP11], SOAP 1.2 [SOAP12], SOAP with Attachments [SOAPATTACH], WS-Security 1.0 [WSS10], WS-Security 1.1 [WSS11], WS-Reliability 1.1 [WSR11] and 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 its previous version are:

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

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 use 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:

- Two AS4 Conformance Profiles (see section 2) that define the subset of ebMS V3 features to be supported by an AS4 implementation.

- An AS4 Usage Profile (section 4) that defines how to use an AS4-compliant implementation in order to achieve similar functions as specified in AS2.

The two 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 IP address.

Compatible existing conformance profiles for ebMS V3 are:

- Gateway RM V3 or Gateway RX V3: an MSH product 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 CPs. These features are described in Section 3.

1.1 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 IETF RFC 2119.

1.2 Normative References

[ebBP-SIG]	*OASIS ebXML Business Signals Schema*, 2006.http://docs.oasis-open.org/ebxml-bp/ebbp-signals-2.0
[ebMS3]	*OASIS ebXML Messaging Services, Version 3.0: Part 1, Core Features*, 2007. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms_core-3.0-spec.pdf
[ebMS3-CP]	*OASIS ebXML Messaging Services, Version 3.0: Conformance Profiles.* Committee Specification 01, April 2010. http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/200707/ebms3-confprofiles.pdf
[RFC1925]	IETF RFC. GZIP file format specification version 4.3. http://tools.ietf.org/html/rfc1952

[RFC2119] S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. IETF RFC 2119, March 1997. http://www.ietf.org/rfc/rfc2119.txt

[RFC2045] N Freed, et al, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, 1996*. http://www.ietf.org/rfc/rfc2119.txt

[SOAP12] M. Gudgin, et al, *SOAP Version 1.2 Part 1: Messaging Framework*, 2003. http://www.w3.org/TR/soap12-part1/

[SOAPATTACH] J. Barton, et al, *SOAP Messages with Attachments*, 2000 http://www.w3.org/TR/SOAP-attachments

[WSIAP10] *WS-I Attachment Profile V1.0*, Web-Services Interoperability Consortium, 2007. http://www.ws-i.org/deliverables/workinggroup.aspx?wg=basicprofile

[WSIBP20] *WS-I Basic Profile V2.0 (final material)*, Web-Services Interoperability Consortium, November 2010. http://ws-i.org/Profiles/BasicProfile-2.0-2010-11-09.html

[WSIBSP11] *Basic Security Profile Version 1.1*, Web-Services Interoperability Consortium, 2006. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.1.html

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

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

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

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

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

1.3 Non-normative References

[ebBP] *ebXML Business Process Specification Schema Technical Specification v2.0.4.* OASIS Standard, 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*, http://www.oasis-open.org/committees/ebxml-cppa/documents/ebCPP-2_0.pdf, OASIS Standard, September, 2002.

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

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

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

[RFC4130] D. Moberg and R. Drummond. *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] T. Harding and R. Scott. *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] D. Box, et al, *Simple Object Access Protocol (SOAP) 1.1*, 2000. http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[WSIBP12] *WS-I Basic Profile V1.2 (final material)*, Web-Services Interoperability Consortium, November 2010. http://ws-i.org/Profiles/BasicProfile-1.2-2010-11-09.html

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

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

[WSS10] Anthony Nadalin, et al, eds., *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

NOTE: AS4 is more than a Conformance Profile, in the sense given in **[ebMS3-CP]**. It is a combination of a Conformance Profile and of an 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 4) is following a format based on tables similar to those found in **[IIC-DP]**.

2.1 The AS4 ebHandler Conformance Profile

The AS4 ebHandler is identified by the URI:

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

2.1.1 Features Set

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

Conformance Profile: **AS4 ebHandler**	**Profile summary**: <“Sending+Receiving” / “AS4 eb Handler” / Level 1 / HTTP1.1 + SOAP 1.2 + WSS1.1 >
Functional Aspects	**Profile Feature Set**
ebMS MEP	Both Sender and Receiver MUST support all ebMS simple MEPs: ● One-way / Push ● One-way / Pull 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) a 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 a 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]) MUST be supported as content for the eb:Receipt message, i.e. when conforming to this profile a Sending MSH must be able to create a Receipt with such a content, and a Receiving 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:Receipts 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 of sending back an eb:Receipt message is: a well-formed ebMS user message has been received and the MSH is taking responsibility for its processing, (no additional application-level delivery semantics, and no payload validation semantics). No support for a WS reliable messaging specification is required although that is an option.
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]) Authorization of the Pull signal - for a particular MPC - must be supported at minimum. 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 ebMS V3, with the wsse:UsernameToken profile. This header may either come in addition to the regular wsse security header (XMLDsig for authentication), or may be the sole wsse header, if a transport-level secure protocol such as SSL or TLS is used. ● **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” .
Error generation and	The following error processing capabilities MUST be supported:

reporting	● Capability of the Receiving MSH to report errors from message processing, either as ebMS error messages or as Faults to the Sending MSH. The following modes of reporting to Sending MSH are supported: ● Sending error as a separate request (ErrorHandling.Report.ReceiverErrorsTo=<URL of Sending MSH>) ● Sending error on the back channel of 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 [ebMS3] are to be generated when applicable, except for EBMS:0010: On Receiving MSH, no requirement to generate error EBMS:0010 for discrepancies between message header and the following P-Mode features: P-Mode.reliability and P-Mode.security, but requirement to generate such error 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 MessageProperties. ● Support for processing messages that contain both a signal message unit (eb:SignalMessage) and a user message unit (eb:UserMessage).
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, 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 (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.
- The above WS-I profiles must be complied with within the scope of features exhibited by the AS4 ebHandler conformance profile. For example, since only SOAP 1.2 is required by 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 is supposed to support the possible options for this parameter.
- 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.

2.1.3.1 General PMode parameters

- **(PMode.ID**: support not required)
- **(PMode.Agreement:** support not 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” scheme.
- **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 “callback”))

2.2 The AS4 Light Client Conformance Profile

The AS4 light Client is identified by the URI:

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

2.2.1 Feature Set

Conformance Profile: **AS4-LightClient**	**Profile summary**: <“Sending+Receiving” / “ lighthandler-rm” / Level 1 / HTTP1.1 + SOAP 1.1>
Functional Aspects	**Profile Feature Set**
ebMS MEP	The following MEPs MUST be supported: ● One-way / Push (as initiator). ● One-way / Pull (as initiator). Regardless of which MEP is used, the sending of an eb:Receipt message MUST be supported: ● For the One-way / Push, the “response” reply pattern 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 a PullRequest, or sent separately. The User message

	receiver MUST be able to send an eb:Receipt separately from the 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 a 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 a 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]) MUST be supported as content for the eb:Receipt message, i.e. when conforming to this profile a Sending MSH must be able to create a Receipt with such a content, and a Receiving 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:Receipts 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 of sending back an eb:Receipt message is: a well-formed ebMS user message has been received and the MSH is taking responsibility for its processing, (no additional application-level delivery semantics, and no payload validation semantics). Support for a WS reliable messaging specification is NOT REQUIRED although that is an option.
Security	Both authorization options for message pulling (authorizing a 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”.
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 (such an error may be bundled with another pushed message or a Pull signal.).

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. the message payload will use the SOAP body. Support for MessageProperties 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:** WSS 1.1.

2.2.2 WS-I Conformance Requirements

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

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

Note: the above WS-I profile MUST be complied with within the scope of features exhibited by the AS4 Light Client ebMS conformance 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.
- 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.

2.2.3.1 General PMode parameters

- **(PMode.ID**: support not required)
- **(PMode.Agreement:** support not 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 not required.

2.2.3.2 PMode[1].Protocol

- **PMode[1].Protocol.Address:** support required for “http” scheme.
- **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).
- **(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”))

2.3 Conformance Profiles Compatibility

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

- Gateway RM V2/3
- Gateway RM V3
- Gateway RX V2/3
- 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 ebMS V3 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 below.

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

Application payloads that are built in conformance with the SOAP Messages with Attachments [SOAPATTACH] specification may be compressed. Support for compression MUST then be provided by AS4 implementations. Compression of the SOAP envelope and/or payload containers within the SOAP Body of an ebMS Message is not supported.

To compress the payload(s) of a message built in conformance with the SOAP Messages with Attachments [SOAPATTACH] specification, the GZIP [RFC1925] 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 =“Compressed”:

```
<eb:Property name="Compressed"/>
```

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

These are indicators to the receiver that compression has been used on this part.

When compression, signature and encryption are required of the MSH, the message MUST be compressed prior to being signed and/or encrypted.

Packaging requirements:

- A eb:PartInfo/eb:PartProperties/eb:Property/@name="MimeType" value is RECOMMENDED to identify the mimetype of the payload before compression was applied.
- A eb:PartInfo/eb:PartProperties/eb:Property/@name="CharacterSet" value is RECOMMENDED to identify the character set of the payload before compression was applied.

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>
      <eb:Property name="Compressed"/>
     </eb:PartProperties>
<eb:PartInfo>
```

An additional PMode parameter is defined, that MUST be supported:

- **PMode[1].PayloadService.Compression:** {true / false}

True: some attached payload(s) may be compressed over this MEP segment.

False (default): no compression is used over this MEP segment.

NOTE: the requirement for Compression feature applies to both conformance profiles (AS4 ebHandler and AS4 light Client).

3.2 Reception Awareness features and Duplicate Detection

These capabilities are making use of the eb:Receipt as the sole type of acknowledgement. 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 resend 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 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 for implementers to decide.
Others	

NOTE: these requirements apply to both conformance profiles (AS4 ebHandler and AS4 light Client)

The following additional PMode parameters are defined and MUST be supported:

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

- **PMode[1].ReceptionAwareness.Replay.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 a composite string specifying 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. With this option, there may be no need for any WS-Security headers 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:

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

(b) The Receiving MSH is taking responsibility for processing this user message. However, no guarantee can be made that this user message will be ultimately delivered to its Consumer application (this responsibility lays however now on the Receiver side).

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 (PMode).
3. The user message was received, the eb:Receipt was sent back but was lost on the way.

See section 4.1.8 for AS4 usage rules about Receipts.

4 AS4 Usage Profile of ebMS 3.0

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 is stating the rules for using messaging features in an AS4-compliant way.
- **AS4 Usage Agreements**: this section is reminding the users of what are the main options left open by the AS4 profiles, that they have to agree on in order to interoperate.

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

4.1 AS4 Usage Rules

4.1.1 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** Notes**:** This Profile only supports the One-Way/Push MEP (Sync and Async) and the One-Way/Pull MEP
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 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".

4.1.2 Bundling rules

Scope of the Profile Feature	Defines bundling (or "piggybacking") rules of ebMS MEPs, including Receipts.
Specification Feature	
Specification Reference	ebMS v3.0, Section 2.2
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. When sending a Receipt for this MEP, a Receiving MSH conforming to this profile SHOULD NOT bundled the Receipt with any other ebMS message header or body.
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 SHOULD NOT bundled the Receipt with any other ebMS message header (including a PullRequest signal) or message body,
Test References	

4.1.3 Security Element

Specification Feature	Use of WSS features
Specification Reference	ebMS v3.0, 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	• *Web Services Security: SOAP Message Security 1.1*, 2005. [WSS11] • *Web Services Security X.509 Certificate Token Profile 1.1*, 2006 [WSS11-X509].
Test References	
Notes	

4.1.4 Signing Messages

Specification Feature	Digital Signatures for SOAP message headers and body
Specification Reference	ebMS v3.0, 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.
Alignment	
Test References	

4.1.5 Signing SOAP with Attachments Messages

Specification Feature	Signing attachments
Specification Reference	ebMS v3.0, 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.
Alignment	
Test References	

4.1.6 Encrypting Messages

Specification Feature	Encrypting messages
Specification Reference	ebMS v3.0, Section 7.4
Profiling Rule (a)	AS4 MSH implementations are SHALL NOT encrypt the eb:PartyInfo section of the eb:Messaging header. Other child elements of the eb:Messaging header MAY be encrypted or left unencrypted as defined by trading partner agreements or collaboration profiles.
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.
Alignment	
Test References	

4.1.7 Encrypting SOAP with Attachments Messages

Specification Feature	Encryption of message attachments.
Specification Reference	ebMS v3.0, 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.
Alignment	
Test References	
Notes	

4.1.8 Generating Receipts

Specification Feature	eb:Receipt signal messages
Specification Reference	ebMS v3.0, Section 7.12..2 (Persistent Signed Receipt) ebMS v3.0, Section 5.2.3.3, eb:Messaging/eb:SignalMessage/eb:Receipt
Profiling Rule (a): Receipts for reception	When a Receipt is to be used solely as a reception indicator (for reception awareness), the sender of the Receipt MAY decide to not insert the

awareness	ebbpsig:NonRepudiationInformation child element. No other element than ebbpsig:NonRepudiationInformation is allowed as child of eb:Receipt. If this element is not used, then eb:Receipt MUST be empty.
Profiling Rule (b): Receipts for Non Repudiation of Receipt (NRR)	Non Repudiation of Receipt (NRR) requires eb:Receipt signals to be signed, and to contain digests of the original message parts for which NRR is required. When signed receipts are requested in AS4 that make use of default conventions, the Sending message handler (i.e. sending messages for which signed receipts are expected) MUST identify message parts using Content-Id values in the MIME headers, and MUST sign the SOAP body and all attachments using the http://docs.oasis-open.org/wss/oasis-wss-SwAProfile-1.1#Attachment-Content-Signature-Transform within the SignedInfo References list. As a reminder, the Sending message handler MUST not encrypt any signed content before signing (Section 7.6 in ebMS V3). If using compression in an attachment, the Sending message handler MUST sign the data after compression (see section 3.1). Variations from default conventions can be agreed to bilaterally, but conforming implementations are only required to provide receipts using the default conventions described in this section.
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 nonrepudiation of receipt of the original message, as described in profiling rule (b). NOTE: the digest(s) to be inserted in the ebbp:MessagePartNRInformation element(s) or the Receipt, related to the original message parts for which a receipt is required, may be obtained from the signature information of the original message (ds:SignedInfo element), as only those parts that have been signed are subject to NRR. This means a Receiving message handler may not have to compute digests outside its security module.
Alignment	
Test References	

439 4.1.9 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.
Alignment	
Test References	

4.2 AS4 Usage Agreements

This section defines the operational aspect of the profile: configuration aspects that users have to agree on, mode of operation, etc. 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 (PMode) parameters defined in the ebMS Core V3 specification. This section only lists the parameters that are particularly relevant to AS4.

4.2.1 Controlling Content and Sending of Receipts

Scope of the Profile Feature	Choose among options in sending Receipts.
Specification Feature	
Specification Reference	ebMS v3.0, Section 2.2
Usage Profiling (a)	Must eb:Receipts 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 ebbp:NonRepudiationInformation element. This is indicated by the new PMode 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. and Callback) This option is controlled by the PMode 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 PullRequest.)

Test References	
Notes	

4.2.2 Error Handling Options

Specification Feature	Error Handling options
Specification Reference	
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: reporting 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)? (a) No error reporting (in case no reception awareness required). (b) 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.) PMode 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): Error about Receipts	How are errors about Receipt messages reported? Pmode parameters: • **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 lay-

	er.
Alignment	
Test References	
Notes	

4.2.3 Securing the PullRequest

Specification Feature	Pulling authorization options
Specification Reference	ebMS v3.0, 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 a PullRequest in two ways: (a) (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. (b) (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. PMode 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 PMode parameters about X509 are controlling both the authentication of PullRequest signals and authentication of other User Messages.
Usage Profiling (b)	PullRequest signals: are they sent using the HTTPS transport protocol with optional Client-side Authentication? PMode parameter: • **PMode[1].Protocol.Address**: The URL scheme will indicate whether HTTPS is used or not.
Alignment	

Test References	
Notes	

4.2.4 Reception Awareness Parameters

Specification Feature	Message Replay and Duplicate Detection options
Specification Reference	N/A AS4 Profile: 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? (c) No message replay. (d) Resend the message. Replay parameters: to agree on: (1) retry number, (2) retry frequency. PMode parameters (additional to those defined in ebMS Core V3): • **PMode[1].ReceptionAwareness:** (true / false) • **PMode[1].ReceptionAwareness.Replay:** (true / false) • **PMode[1].ReceptionAwareness.Replay.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) In addition to (a), a receiver detects and eliminates duplicates based on eb:MessageInfo/eb:MessageId. PMode parameters (additional to those defined in ebMS Core V3): • **PMode[1].ReceptionAwareness.DuplicateDetection:** (true / false) • **PMode[1].ReceptionAwareness.DuplicateDetection.Parameters**
Others	
Notes	

4.2.5 Default Values of Some PMode Parameters

Specification Feature	Default values and authorized values for main PMode parameters.
Specification Reference	ebMS 3.0, 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 PMode content default: absence of the PMode itself will cause the default value defined in the ebMS V3 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 PMode content default: absence of the PMode itself will cause the default value defined in the ebMS V3 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
Alignment	
Test References	
Notes	

452 4.2.6 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 3, 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?
Alignment	
Test References	
Notes	

4.2.7 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?
Others	

4.2.8 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 ?
Others	

4.2.9 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.
Others	

5 Conformance Clauses

5.1 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 PMode parameters as required in Section 2.1.3.

In addition, the implementation must implement the additional features as indicated in Section 3.

Finally, the implementation must support the Usage Rules defined in Section 4.1.

The Usage Agreements in Section 4.2 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 4.1.

5.2 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 PMode parameters as required in Section 2.2.3.

In addition, the implementation must implement the additional features as indicated in Section 3.

Finally, the implementation must support the Usage Rules defined in Section 4.1.

The Usage Agreements in Section 4.2 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 4.1.

5.3 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 for the AS4 Light Client Conformance Clause stated in Section 5.2, with the exception that support for WS-Security is limited to support for the WS-Security UsernameToken profile [WSS11-UT], to be used for authorization of message pull signals (see section 7.10 in Core Spec). Support for the WS-Security X.509 Certificate Token Profile 1.1 [WSS11-X509] is not REQUIRED. Clients and servers SHOULD use transport level security for message security for any message exchange.

5.4 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 5.1, also be a conformant implementation of the EDIINT Applicability Statement 2 (AS2, [RFC4130]).

Appendix A Sample Messages

This appendix contains examples of the AS4 functionality Non-Repudiation of Receipt (NRR) and an example of an AS4 Pull message signal.

Appendix A.1 Non-Repudiation of Receipt

When the NonRepudiationInformation element is used in a Receipt, it contains a sequence of 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 4.1.4 and 4.1.5) 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.2 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>PreCqm0ESZqmITjf1qzrLFuOEYg=</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 Acknowledgments

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

- Timothy Bennett, Drummond Group Inc. <timothy@drummondgroup.com>
- Jacques Durand, Fujitsu Limited <jdurand@us.fujitsu.com>
- Richard Emery, Axway Software <remery@us.axway.com>
- Ian Jones, British Telecommunications plc <ian.c.jones@bt.com>
- Dale Moberg, Axway Software <dmoberg@axway.com>
- Makesh Rao, Cisco Systems, Inc. <marao@cisco.com>
- Pim van der Eijk, Sonnenglanz Consulting <pvde@sonnenglanz.net>
- John Voss, Cisco Systems, Inc. <jovoss@cisco.com>

Appendix C 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	CSD 3 draft for PR.