1. Description

The e-SENS AS4 Profile is a profile of the ebMS3 and AS4 OASIS Standards. It has provisions for use in four-corner topologies, but can also be used in point-to-point exchanges. This specifications profile can be implemented using open source or closed source commercial software products compliant with these standards. It is designed to support both One Way and Two Ways (Request-Response) exchanges. The profiling is heavily based on the ENTSOG (the European Network of Transmission System Operators (TSO) for Gas) AS4 profile for TSOs and on e-CODEX specifications.

2. Base Specifications

This specification is based on the following OASIS Standards:


3. Profiling
The e-SENS PR-AS4 profiles the OASIS ebMS3 and AS4 standards, using input from two other sources:

- ENTSOG (the European Network of Transmission System Operators (TSO) for Gas) AS4 profile for TSOs is an interoperability profile for AS4 [ENTSOGAS4]. This profile incorporates state-of-the-art security guidelines and has been reviewed by experts from the European Union Agency for Network and Information Security (ENISA). This e-SENS profile is very close to the ENTSOG profile for general AS4 messaging aspects.
- The e-CODEX project has selected the use of ebMS3/AS4 according to the e-Delivery convergence agreements [ECODEXD5.11] and profiled some functionality. e-CODEX has used ebMS3/AS4 in production since July 2013 in its community. Their profiling covers specific aspects of relevance to e-SENS, in particular the support for four corner topologies, which this e-SENS profile adopts.

The following table summarizes the features provided by the ebMS3 and AS4 standards. The e-SENS profile is a profiled extended subset of the AS4 standard.

<table>
<thead>
<tr>
<th>Functionality</th>
<th>ebMS 3.0 AS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Messaging</td>
<td>Web Services</td>
</tr>
<tr>
<td>Internet Transport</td>
<td>HTTP 1.1</td>
</tr>
<tr>
<td>Transport Layer Integrity, Sender Authentication, Receiver Authentication and Message Confidentiality (Non-Persistent)</td>
<td>Transport Layer (SSL / TLS) Security</td>
</tr>
<tr>
<td>Message and Payload Packaging</td>
<td>SOAP 1.2 with attachments</td>
</tr>
<tr>
<td>Routing and Dispatching, SOA Integration</td>
<td>Mandatory &quot;Service&quot; and &quot;Action&quot; header elements</td>
</tr>
<tr>
<td>Exchange Patterns</td>
<td>One Way or Two Way (*)</td>
</tr>
<tr>
<td>Exchange Pattern Bindings</td>
<td>Push, Pull and Sync (*)</td>
</tr>
<tr>
<td>Payload Compression</td>
<td>Gzip (**)</td>
</tr>
<tr>
<td>Message Identification</td>
<td>ebMS 3.0 &quot;MessageId&quot;</td>
</tr>
<tr>
<td>Message Correlation</td>
<td>ebMS 3.0 &quot;RefToMessageId&quot; and &quot;ConversationId&quot;</td>
</tr>
<tr>
<td>Message Timestamp</td>
<td>ebMS 3.0 &quot;Timestamp&quot; and WS-Security &quot;Timestamp&quot;</td>
</tr>
<tr>
<td>Party Identification</td>
<td>ebMS 3.0 &quot;From&quot; and &quot;To&quot; party identifiers.</td>
</tr>
<tr>
<td>Non-Repudiation of Origin</td>
<td>WS-Security 1.1 using XML Signature</td>
</tr>
<tr>
<td>Message Confidentiality</td>
<td>WS-Security 1.1 using XML Encryption</td>
</tr>
<tr>
<td>Non-Repudiation of Receipt</td>
<td>Signed Receipt Signal Message</td>
</tr>
<tr>
<td>Reliable Message</td>
<td>AS4 reception awareness feature for lightweight, interoperable reliable messaging (**)</td>
</tr>
</tbody>
</table>

Table 1. ebMS3/AS4 Functional Overview. (*) in ebMS3, not in AS4 (**) AS4 extension tot ebMS3

This specification defines an e-SENS AS4 profile as the selection of a specific conformance profile of the AS4 standard [AS4] that is profiled further for increased consistency and ease of configuration, and an AS4 Usage Profile that defines how to use a compliant implementation for e-SENS document exchange. In this profile some features available in AS4 are not used (Pull and Sync exchange pattern bindings) whereas others (TLS, XML Signature and XML Encryption) are mandatory in the e-SENS profile. Furthermore, support for the Two Way MEP is mandatory.

3.1. Benefits of ebMS3/AS4

Message packaging provided by AS4 as an add-on feature relies on ebMS 3.0 support for the SOAP 1.1 and 1.2 standards [SOAP12]. AS4 combines the traditional functional support of payload compression in line with ebMS 3.0 message packaging norms. The compression must be applied in AS4 prior to the application of any message-level security such as digital signing or encryption. AS4 does not define a maximum message size, though implementations will have practical limits based on available memory, disk or database storage etc.

AS4 offers a secure document exchange protocol for use over the Internet that leverages envelope structure to transport arbitrary payloads. Support for Message Security and Confidentiality is provided by AS4 via ebMS 3.0 WS-Security 1.0 and 1.1 standards. This includes combinations of XML Digital Signatures and XML Encryption X.509 security tokens for signing and encrypting as primary means.
for authenticating messages, ensuring privacy, and guaranteeing safe data transmission. Additionally, AS4 supports the use of username/password tokens as access control to message pull channels.

The ebMS 3.0 and AS4 standards provide support for Non-Repudiation of Receipt (NRR) by using a Signed Receipt Signal Message. The receipt is returned using a special signal message and may also contain error handling information if there was some problem with the document exchange.

AS4 makes use of the message receipt as a signal to the original message sender that the recipient of the message has received the business payload. AS4 supports duplicate message detection and message retry/resending scenarios for when receipts for messages are not received by the sender.

Other technical highlights are:

- **Payload agnosticism:** the document type (e.g., purchase order, invoice, etc.) is not tied to any defined SOAP action or operation;
- **Support for single or multiple payloads contained either within the SOAP body or as SOAP attachment(s);**
- **Support for the ebMS 3.0 One-Way/Push message exchange pattern with support for either synchronous or asynchronous signal responses;**
- **Support for the ebMS 3.0 One-Way/Pull message exchange pattern which is beneficial for exchanging documents with non-addressable endpoints;**
- **Reception Awareness features and Duplicate Detection capabilities make use of the eb:Receipt as the sole type of acknowledgment.**

Note that this version of this profile does not use the **Pull** pattern. However, the **Pull** pattern is of potential interest for certain e-Interaction scenarios. Future versions of this profile may require additional support for **Pull**.

### 3.2. AS4 and Conformance Profiles

The e-SENS AS4 profile is based on the AS4 Profile of the ebMS 3.0 Version 1.0 OASIS Open Standard [AS4]. AS4 itself is based on other standards, in particular on OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features OASIS Standard [EBMS3], which in turn is based on various Web Services specifications.

The OASIS Technical Committee responsible for maintaining the AS4, ebMS 3.0 Core and other related specifications is tracking and resolving issues in the specifications [EBERRATA]. These resolutions will eventually be published as a consolidated Specification Errata but should already be taken into account by implementers, to avoid functional or interoperability issues.

The AS4 standard defines multiple conformance profiles, which define specific functional subsets of the version 3.0 ebXML Messaging, Core Specification. A conformance profile corresponds to a class of compliant applications.

This AS4 Profile is based on an extended subset of the **AS4 ebHandler Conformance Profile** and an Usage Profile. It supports transparent interconnection of existing electronic delivery communities via e-SENS e-Delivery Gateways using the ebMS3 “Push” transport channel bindings.

By using “Push”, messages that are submitted to a sending gateway (C2) are forwarded to the receiving gateway (C3) immediately, without the (unpredictable) delay of a “Pull” transport channel binding. Assuming the latency of the transmission of the message from the receiving gateway (C3) to the end entity (C4), the business processing of the message by that end entity and the reverse flow from C4, via C3 and C2 can be minimized similarly, this profile can support business processes that need “interactive” responses.

An example of such a business process is e-Confirmation [ECONF]. For e-Confirmation, a health care provider in MS B is able to get an insurance confirmation for a patient who is insured with a health insurance organization in a member state (MS A) of the EU/EES. The Health Care Provider requests for insurance verification which delivers a Provisional Replacement Certificate after having verified positively that the patient is insured. The Health Care Provider needs to have an interface to the e-Confirmation service. The request for insurance verification is submitted to an access point. The requesting access point routes the message to the access point of the providing health insurance organizations. When the message is delivered to the health insurance organization then the confirmation can be provided and routed back to the health care provider.

### 3.3. e-SENS AS4 ebHandler Feature Set

The e-SENS AS4 feature set is, with some exceptions, a subset of the feature set of the AS4 ebHandler Conformance Profile. This section selects specific options in situations where the AS4 ebHandler provides more than one option. This can be used as a checklist of features to be provided in AS4 products. The structure of this chapter mirrors the structure of the ebMS3 Core Specification [EBMS3].

The ebMS 3.0 protocol can support synchronous as well as asynchronous communication and provides full convergence with Web Services. It reuses the SOAP 1.2, WS-Security 1.1, and SOAP-with-attachments specifications. It complies with the WS-I Basic Profile (BP) and Basic Security Profile (BSP) and provides additional features of particular relevance to small and medium-size enterprise, in particular message pulling.

Compared to the AS4 ebHandler Conformance Profile, this profile updates or adds some functionality:

- There is an added requirement to support Two Way MEPs.
- Transport Layer Security, if handled in the AS4 handler, is profiled and is mandatory.
- The WS-Security version is the 1.1.1 version.
- Algorithms specified for securing messages at the Message Layer are updated to current guidelines and use of signing and encryption is mandatory.

It also relaxes some requirements:
- Support for Pull mode in AS4 is not required in this version.
- All payloads are exchanged in separate MIME parts.
- Receipts and errors are reported synchronously only.
- WS-Security support is limited to the X.509 Token Profile. The use of UserName Tokens is not supported.

3.4. Message Exchange Patterns

The following paragraphs summarize some key concepts and terminology defined in the ebMS 3 core specification [EBMS3]:

1. **Messaging Service Handler (MSH), Producer, Consumer** an MSH is an entity that is able to generate or process messages that conform to the ebMS specification, and to act as sender or receiver. A Producer is an entity (e.g. application) that interacts with a Sending MSH (i.e. an MSH in the Sending role) to initiate the sending of a user message. A Consumer is an entity that interacts with a Receiving MSH (i.e. an MSH in the Receiving role) to consume data from a received user message.

2. **Message, User Message, Signal Message** a Message is a logical unit which consists of User Messages or Signal Messages or both. A User Message is a message that contains a User Message unit (an eb:Messaging/eb:UserMessageXML structure). A Signal Message is an ebMS message that contains a Signal Message unit (an eb:Messaging/eb:SignalMessage XML structure). In other words, there exist two types of messages in the ebMS standard: the first type allows transmitting data interpreted by a Consumer and the second type allows transmitting data interpreted by an MSH as a signal (e.g. a pull signal).

3. **Message Exchange Pattern (MEP)**, One-Way/Push, One-Way/Pull, Two-Way/Sync MEP a MEP is an agreement between sending and receiving MSHs. Some aspects of MEPs supported in the messaging layer include:
   - Specifying the correlation between messages sent and received in the message header.
   - Message binding to the underlying transfer-protocol.

4. **Processing Mode (PMode)** - A PMode is the contextual information that governs the processing of a particular message (thus is basically a set of configuration parameters). The PMode associated with a message determines, among other things, which security and/or which reliability protocol and parameters, as well as which MEP is being used when sending a message. The technical representation of the PMode configuration is implementation-dependent. For example, the Open Source product Domibus [DOMIBUS] which is the CEF sample implementation of the eDelivery Access Point, has an XML representation of PModes. E-CODEX has developed tooling to support the creation, management and distribution of PModes using the Domibus PMode XML file format.

The **Messaging Model** of the AS4 profile constrains the channel bindings of message exchanges between two AS4 MSHs. The following diagram shows the AS4 Messaging Model, various actors and operations in message exchange:

![Messaging Model](image)

*Figure 1. Entities of the AS4 Messaging Model and their Interactions [EBMS3].*

Business applications or middleware, acting as Producer, Submit message content and metadata to the Sending MSH, which packages this content and sends it to the Receiving MSH of the business partner, which in turn Delivers the message to another business application that Consumes the message content and metadata. Subject to configuration, Sending and Receiving MSH may Notify Producer or Consum
of particular events. Note that there is a difference between Sender and Initiator: For Push exchanges, the Sending MSH initiates the transmission of the message. For Pull exchanges (not supported in the e-SENS profile), the transmission is initiated by the Receiving MSH. Also note that a business application can include MSH functionality, leaving the MSH as an abstract concept.

The AS4 ebHandler Conformance Profile is the AS4 conformance profile that provides support for Sending and Receiving roles using Push channel bindings. Support is required for the following Message Exchange Patterns:

- One Way / Push
- Two Way / Push-and-Push

Generally in the ebMS MEP context pushing means that the sender initiates the message exchange (for HTTP this implies that the sender is an HTTP client, and the receiver a server). Pulling in the ebMS MEP context means that the receiver initiates the message exchange (so the receiver would be an HTTP client and the sender an HTTP server).

The One-Way/Push MEP for example specifies a situation when a sending MSH which has agreed to use the One-Way/Push MEP sends a message to a receiving MSH which has agreed to use One-Way/Push MEP as well. In this case the message that would be sent is most likely a message carrying the user data. (It can also be a signal message e.g. error message.) After the reception the receiving MSH would send a non-user message (i.e. a signal message) to the sending MSH to confirm the reception. Different user messages do not have any reference to each other.

![Figure 2. One-Way/Push MEP [EBMS3]](image)

While the AS4 ebHandler does not require support for the Two-Way MEP, support for this MEP is required for e.g. eInteraction SAT. A message handler that supports Two Way MEPs allows the Producer submitting a message unit to set the optional RefToMessageId element in the MessageInfo section.

For PMode.MEP, support is therefore required for the following values:

- http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay
- http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/twoWay

For PMode.MEPbinding, support is required for:

- 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/pushAndPush

Note that these URI values are identifiers only, which are defined in the ebMS3 standard. They do not resolve to content on the OASIS site.

Time-critical processes require the Push channel binding, because it allows the Sender to control the timing of transmission of the message. Interactive, request-response communication between end entities A and B can be provided as a combination of two Push mess ages, one from the sending gateway (C2), on behalf of A, to the receiving gateway (C3), which in turn acts as receiver for B, followed by a separate (asynchronous) response message from the C3 to C2.
The **Two-Way/Sync MEP** specifies a situation when a **MSH** which has agreed to use the **Two Way/Sync MEP** would send a user message to another **MSH** which has also agreed to use the **Two Way/Sync MEP**. After the reception of the user message the receiving **MSH** would send a user message to the sending **MSH** using the backchannel of the request channel. The second user message refers to the ID-field specified in the request user message. The **Sync** transport channel binding is not part of AS4 and not currently part of this profile.

The **Two-Way/Push-and-Push** MEP must be supported. It is very similar to a sequence of two **One-Way/Push** exchanges, in which the **Sender** and **Receiver** roles are reversed so that the **Responding MSH** that processed the request **User Message** in the first exchange becomes the **Sending MSH** in a second, separate exchange in which the business response **User Message** is transmitted. A response **User Message** must have a **RefToMessageId** element with the value set to the value of the **MessageId** in the corresponding request message. This second exchange is separately initiated and, unlike the **Sync** channel binding, does not depend on any connection timeout intervals of the underlying **HTTP** transport.

In many four-corner deployments, the interface between corners 3 and 4 is based on a polling mechanism. If a service provider would want to use ebMS3/AS4 as an interface to its customers, a polling interface could be provided using the ebMS3 **Pull** feature. However, that interface is out of scope for this specification and could use other technology.

In a four-corner topology, the scope of the interconnect transport protocol is limited to the interaction between the inner two corners (C2 and C3). The **Message Producer** is actually some middleware or business module that receives messages from an original sender (C1) using a separate transport infrastructure and re-submits the message for forwarding. That middleware or business module may or may not be integrated into a single software component, depending on the technologies and implementations used. Similarly, the **Message Consumer** is some module that forwards the message (by re-submitting it to some transport component) to the final recipient (C4). This model is not to be confused with the SOAP processing model as defined in SOAP 1.2, second edition [SOAP12] used in the ebMS3 Part 2 “Multihop” module [EBMS3P2], which defines ebMS3 SOAP intermediaries that forward SOAP messages. In that model there is no submission or delivery to the intermediary MSH and end-to-end security and reliability can be provided.

### 3.5. Message Packaging

The **AS4 Message Structure** provides a standard message header that addresses B2B requirements and offers a flexible packaging mechanism based on SOAP and MIME enveloping. Dashed line style is used for optional message components.

![Figure 3. AS4 Message Structure, UserMessage.](image)

#### 3.5.1. UserMessage

**AS4** defines the ebMS3 **Messaging** SOAP header, which envelopes **UserMessage** XML structures, which provide business metadata to exchange payloads. In **AS4**, ebMS3 messages other than receipts or errors carry a single **UserMessage**.

An **MSH** must not include more than one **PartyId** element in the **UserMessage/PartyInfo/From** and **UserMessage/PartyInfo/To** elements.
A compliant product, acting as Sending MSH, must allow the Producer, when submitting a message, to:

- set the value for ConversationId. This enables the Consumer to correlate the message to related messages that are part of the same conversation.
- set the value for ReferToMessageId, for business response messages in a Two Way MEP. This allows the Consumer to determine to which previous AS4 request message the message is a response. Note that a shared value for ConversationId is not sufficient for correlating requests and responses as there may be more than one outstanding request in a single conversation.

A compliant product, acting as Receiving MSH, must provide the MessageId, ReferToMessageId and ConversationId values as metadata with any message it delivers to the Consumer.

To be able to relate a business response message to a previous business request, the Producer MUST know the MessageId of this request message. The implementation of this requirement is left to implementations, but MAY be one of the following:

- The MSH allows the Producer to specify, when submitting the message, a specific value to be used for MessageId.
- The MSH generates the MessageId value, but notifies the Producer of the generated value value

The ebMS3 and AS4 specifications do not constrain the value of MessageId beyond conformance to the Internet Message Format [RFC2822], which requires the value to be unique. It is RECOMMENDED that the value be universally unique. Products can do this by including a UUID string in the id-left part of the identifier set using randomly (or pseudo-randomly) chosen values.

As in the AS4 ebHandler profile, support for MessageProperties is required in this profile. It must be possible to set the type attribute for message properties (see https://issues.oasis-open.org/browse/EBXMLMSG-2).

The ebMS standard [EBMS3] defines the PMode[].BusinessInfo.Service element as: “This REQUIRED element occurs once. It is a string identifying the service that acts on the message and it is specified by the designer of the service.” The header is of XML schema type non-empty-string and its value is a PMode parameter, meaning communication partners are expected to define specific values for specific process modes, i.e. for various types of messages. The Service element can have a type attribute to categorize services. In [EBMS3], Action is defined as: “This REQUIRED element occurs once. It is a string identifying the action the User message is intended to invoke on a particular service and it is specified by the designer of the service.” For each message exchange, [EBMS3] requires setting the values PMode.Initiator.Role and PMode.Responder.Role, which are used to set From/Role and To/Role values.

E-SENS pilots using these implementation guidelines MUST define string values for Services and, for each Service, the associated Actions. For each Service and Action, the From Role and To Role MUST be defined. This profile only provides high-level constraints on naming conventions on Service and Action.

- The value of Service SHOULD identify a set of related business transactions or other message exchanges in the context of a business process or use case.
- The value of Action SHOULD identify the different types of business transactions or other message exchanges in the context of an identified Service. This MAY be an identifier of a document type, if the exchange of a document of that type unambiguously identifies the purpose and requested action in the context of the Service.

When used in conjunction with the e-SENS SMP - 1.9.0, the following XPath expressions in the SMP document MUST return identical values to corresponding PMode parameter values:

- SignedServiceMetadata/ServiceMetadata/ServiceInformation/Processlist/Process/ProcessIdentifier: PMode[1].BusinessInfo.Service
- SignedServiceMetadata/ServiceMetadata/ServiceInformation/Processlist/Process/ServiceEndpointList/Endpoint
  /EndpointReference/Address: PMode[].Protocol.Address

### 3.5.2. Payloads

Section 5.1.1 of the ebMS3 Core Specification [EBMS3] requires implementations to process both non-multipart (simple SOAP) messages and multipart (SOAP-with-attachments) messages, and this is a requirement for the AS4 ebHandler Conformance Profile. AS4 messages based on this profile must not include any payload content in the SOAP body. Due to the mandatory use of the AS4 compression feature in this profile (see section 2.2.3.3), XML payloads MAY be converted to binary data, which is carried in separate MIME parts and not in the SOAP Body. Compliant AS4 message always have an empty SOAP Body.

The ebMS3 mechanism of supporting "external" payloads via hyperlink references (as mentioned in section 5.2.2.12 of the ebMS3 Core Specification [EBMS3]) must not be used.

If AS4 is used to exchange multiple payload parts in a single message, where there are cross-references between payloads encoded using Content-ID resource locator syntax, the submitting Producer MUST be able to set Content-ID values for MIME payload parts and the Receiving MSH MUST make payload part Content-ID information available to the the Consumer. Note that as an alternative to the use of the MIME Content-Id AS4 offers the option to assign part properties to each payload included in the message which can also be used for cross-references between payloads. To obviate the need for cross-references between payloads the Producer can also package them into one container part before submitting to the MSH.

### 3.5.3. Compression

The AS4 specification defines Payload Compression as one of its additional features. Payload compression is a useful feature for many content types, including XML content.
To compress the payload(s) of a message payload, the GZIP [RFC1952] compression algorithm must be used (GZIP is the only compression type currently supported in AS4). 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/or 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 value of this property MUST conform to the values defined in section 4.3.3 of [XML10].

### 3.5.4. 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 = "CompressionType" >application/gzip</ eb:Property >
  </ eb:PartProperties >
</ eb:PartInfo >
```

An additional PMode 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 does not result in good compression ratios and is therefore not required.

**Property Absent** (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.

The AS4 compression feature specifies the use of compression at the message layer, the structure and format of an AS4 message that uses AS4 compression, and the behavior of the sending and receiving MSH. The AS4 compression feature recommends, but does NOT REQUIRE, the sending MSH to compress the payloads, as noted that https://issues.oasis-open.org/browse/EBXMLMSG-79. Whether or not compression is applied, for messages for which AS4 compression is specified in its P-Mode, is therefore left to the implementation of
the sending MSH. A receiving MSH MUST NOT reject messages with payloads that are not compressed even though AS4 compression is specified in the P-Mode. However, the receiving MSH is REQUIRED to decompress any compressed payloads for messages for which the P-Mode specifies the use of AS4 compression, and for which the CompressionType part property is set to application/gzip. Decompression failures MUST be reported using the EBMS:0303 error code.

3.6. Error Handling

For the error handling this profile specifies that errors must be reported and transmitted synchronously to the Sender and should be reported to the Consumer.

- The parameter `PMode[1].ErrorHandling.Report.AsResponse` must be set to the value `true`.
- The parameter `PMode[1].ErrorHandling.Report.ProcessErrorNotifyConsumer` should be set to the value `true`.

If a message has not been successfully processed, instead of a receipt, the Receiving MSH should return an error.

- The parameter `PMode[1].ErrorHandling.DeliveryFailuresNotifyProducer` should be set to the value `true`.

3.7. Reliable Messaging and Non-Repudiation of Receipt

For Reliable Messaging this profile specifies that non-repudiation receipts must be sent synchronously for each message type. Note that non-repudiation is only "per hop" in the case of the four-corner-model, in particular the hop from corner two to corner three. In e-SENS, the optional end-to-end services module supports the traceability across the four corners.

- The parameter `PMode[1].Security.SendReceipt.NonRepudiation` must be set to the value `true`.
- The parameter `PMode[1].Security.SendReceipt.ReplyPattern` must be set to the value `Response`.

An AS4 receipt indicates that the message has been "successfully processed by the Receiving MSH (i.e. not just "received")". In a four corner topology, the Receiving MSH is C3. The AS4 receipt therefore does not express successful delivery to the end receiver C4. The scope of AS4 non-repudiation of receipt is therefore limited to C2-C3 Message Exchange and does not provide end-to-end non-repudiation of receipt by C4.

This profile requires the use of the AS4 Reception Awareness feature. This feature provides a built-in Retry mechanism that can help overcome temporary network or other issues and detection of message duplicates.

- The parameter `PMode[1].ReceptionAwareness` must be set to `true`.
- The parameter `PMode[1].ReceptionAwareness.Retry` must be set to `true`.
- The parameter `PMode[1].ReceptionAwareness.DuplicateDetection` must be set to `true`.

The parameters `PMode[1].ReceptionAwareness.Retry.Parameters` and related `PMode[1].ReceptionAwareness.DuplicateDetection`. Parameters are sets of parameters configuring retries and duplicate detection. These parameters are not fully specified in [AS4] and implementation-dependent. Products must support configuration of parameters for retries and duplicate detection. Reception awareness errors generated by the Sender must be reported to the Submitting application:

- The parameter `PMode[1].ErrorHandling.Report.SenderErrorsToThirdParty` must not be set. There is no support for reporting sender errors to a third party.

3.8. Security

AS4 message exchanges can be secured at multiple communication layers: the network layer, the transport layer, the message layer and the payload layer. The first and last of these are not normally handled by B2B communication software and therefore out of scope for this section. Transport layer security is addressed, even though its functionality may be offloaded to another infrastructure component.

This section provides parameter settings based on multiple published sets of best practices. It is noted that after publication of this document, vulnerabilities may be discovered in the security algorithms, formats and exchange protocols specified in this section. Such discoveries should lead to revisions to this specification.

3.8.1. Transport Layer Security

When using AS4, Transport Layer Security (TLS) is an option to provide message confidentiality and authentication. Server authentication, using a server certificate, allows the client to make sure the HTTPS connection is set up with the right server.

- When a message is pushed, the Sender authenticates Recipient's server to which the message is pushed.
- When a message is pulled, the Receiver authenticates Sender's server from which the message is pulled.

Guidance on the use of Transport Layer Security is published in the ENISA Algorithms, Key Sizes and Parameters Report 2013[ENISAALKS P] and in a Mindeststandard of the Bundesamt für Sicherheit in der Informationstechnik [BSITLS]. If TLS is handled by the AS4 message handler (and not off-loaded to some infrastructure component), then:
• It must be possible to configure the accepted TLS version(s) in the AS4 message handler. The ENISA and BSI reports state that TLS 1.0 and TLS 1.1 should not be used in new applications. Older version such as SSL 2.0 [RFC6176] and SSL 3.0 must not be used. Products compliant with this profile should therefore support TLS 1.2 [RFC5246].
• It must be possible to configure accepted TLS cipher suites in the AS4 message handler. IANA publishes a list of TLS cipher suites [TLSSP], only a subset of which the ENISA Report considers future-proof (see [ENISAAKSP], section 5.1.2). Products must support cipher suites included in this subset. Vendors must add support for newer, safer cipher suites, as and when such suites are published by IANA/IETF.
• Support for SSL 3.0 and for cipher suites that are not currently considered secure should be disabled by default.
• Perfect Forward Secrecy, which is required in [BSITLS], is supported by the TLS_ECDHE_* and TLS_DHE_* cipher suites, which are therefore preferred and should be supported.

If TLS is not handled by the AS4 message handler, but by another component, then these requirements are to be addressed by that component.

Transport Layer client authentication authenticates the Sender (when used with the Push MEP binding) or Receiver (when used with Pull). Since this profile uses WS-Security for message authentication, the use of client authentication at the Transport Layer can be considered redundant. Whether or not client authentication is to be used depends on the deployment environment. To support deployments that do require client authentication, products must also support Transport Layer client authentication to be configured for an AS4 HTTPS endpoint. Optionally, 2-Way TLS Authentication is a combination of client and server authentication.

3.8.2. Message Layer Security

To provide message layer protection for AS4 messages, this profile requires the use of the following Web Services Security version 1.1 OASIS Standards, profiled in ebMS3.0 [EBMS3] and AS4 [AS4]:

- Web Services Security SOAP Message Security [WSSSMS].
- Web Services Security SOAP Message with Attachments (SwA) Profile [WSSW].
- Web Services Security XML Signature [XMLDSIG].
- The current version is the April 2013, Version 1.1 specification [XMLSIG1], which defines important new algorithm identifiers, including identifiers for SHA2, and deprecates SHA1, in line with guidance from ENISA [ENISAAKSP].

This e-SENS AS4 profile uses the following AS4 parameters and values:

- The PMode[1].Security.X509.Sign parameter must be set in accordance with section 5.1.4 and 5.1.5 of [AS4].

This anticipates an update to the AS4 specification to reference this newer version of the XML Signature specification.

The use of XML Signature in AS4 provides Non Repudiation of Origin (NRO) at Message Exchange level. If this profile is used in a four corner topology, the originator is understood to be C2. Any end-to-end Non Repudiation of Origin, committing C1 is out of scope for this profile.

For encryption, WS-Security leverages the W3C XML Encryption recommendation. The following AS4 configuration options configure this feature:

- The PMode[1].Security.X509.Encryption.CipherSuites parameter must be set in accordance with section 5.1.6 and 5.1.7 of [AS4].

AS4 also references an older version of XML Encryption than the current one ([XMLENC] instead of [XMLENC1]). However, the AES 128 algorithm [AES] was already referenced in that earlier version. AES is fully consistent with current recommendations for “near term” future system use [ENISAAKSP]. However, the newer W3C specification recommends AES GCM strongly over any CBC block encryption algorithms.

Key Transport algorithms are public key encryption algorithms especially specified for encrypting and decrypting keys, such as symmetric keys used for encryption of message content. No parameter is defined to support configuration of key transport in [EBMS3]. Implementations are recommended to support the following algorithms:

- For encryption method algorithm, http://www.w3.org/2009/xmlenc11#rsa-oaep . This is the algorithm used as value for the Algorithm attribute of xenc:EncryptionMethod on xenc:EncryptedKey.
- As mask generation function, http://www.w3.org/2009/xmlenc11#mgf1sha256 . This is the algorithm used as value for the Algorithm attribute of xenc:MGF1 in xenc:EncryptionMethod.
- As digest generation function, http://www.w3.org/2001/04/xmlenc#sha256 . This is the algorithm used as value for the Algorithm attribute of xenc:DigestMethod in xenc:EncryptionMethod.
3.9. Usage Profile

This section contains implementation guidelines that specify how products that comply with the requirements of the e-SENS AS4 ebHandler should be configured and deployed. This is similar to the concept of Usage Agreements in section 5 of [AS4] as it does not constrain how AS4 products are implemented, but rather how they are configured and used. The audience for this section are operators/administrators of AS4 products and B2B integration project teams. The structure of this chapter also partly mirrors the structure of [EBMS3], and furthermore covers some aspects outside core pure B2B messaging functionality.

3.9.1. Message Packaging

For message packaging the usage profile constrains values for several elements in the AS4 message header and the overall message structure.

3.9.2. UserMessage and Gateway Addressing

In the usual scenarios where the ebMS protocol is used for point-to-point communication between end entities, the From and To fields in the UserMessage will be used to identify the sender and receiver respectively (UserMessage/PartyInfo/From/to/PartyId elements). However, in a four-corner-model, the sender and recipient of ebMS messages are the gateways (C2, C3), not the end entities (C1, C4). To facilitate the use of commercial or open source out-of-the-box messaging products and to simplify configuration of gateways, From/PartyId and To/PartyId shall therefore in this case address the identifiers of gateways. This is consistent with current practice for ebMS3 in e-CODEX and with the PEPPOL AS2 profile.

For e-SENS, the identifier system to be used for addressing is specified in the e-SENS ABB “Addressing”.

3.9.3. UserMessage and End Entity Addressing

To be able to forward a received message, the receiving gateway (C3) needs to be able to determine the end entity (C4) that an e-SENS AS4 message is intended for. This information is generally available in the business document. However, using information from the business document assumes an understanding of the schema on which the document is based. Since gateways need to be able to process documents of any type, it is desirable to adopt a mechanism that is independent of particular schemas.

The e-CODEX documentation for its use of ebMS3/AS4 [ECODEXD5.11] uses the ebMS3 property mechanism to attach arbitrary pairs of property-values to a message to address C1 and C4:

- The property named originalSender addresses the original (end entity) sender party.
- The property named finalRecipient addresses the final (end entity) recipient party

The type attribute may be used to categorize party identifier types. Implementations of the e-SENS e-Delivery AS4 profile must support this mechanism: the sender gateway (or integration middleware) must set, and the receiver gateway (or integration middleware) must get, these properties and values.

A key advantage of the use of these properties is that no constraints are imposed on message payload. It is possible to transport, route and forward any payload, even if unstructured, binary or encrypted.

This profile defines an additional, optional third property:

- The property named trackingIdentifier provides a mechanism to include an identifier (in arbitrary string format) that allows end-to-end tracking of messages in a four-corner exchange. Its value could be set to the value of an identifier for the message from C1 to C2 that the AS4 message relates to. This allows tracking and tracing of messages from C1, via C2 to (at least) C3.

3.9.4. Standard Business Document Header (SBDH)

End entities and other end-to-end information, such as the receiver and the sender address, the type of the payload and the business scope, can also be encoded outside the ebMS header, in a specialized XML payload, a kind of internal header. The business document that is being exchanged does not need to be modified, so this specialized payload would be an additional message part. Like any payload, it needs to be referenced from the ebMS UserMessage header.

An option for e-SENS is to use UN/CEFACT Standard Business Document Header [SBDH] that allows end entities to encode information on business process, business transaction, agreement, and business quality-of-service. The SBDH is widely adopted in e-business communities like GS1. The following example shows how an SBDH used to exchange documents looks like, with the remark that for e-SENS should be used a Manifest block for sending non-XML documents or files.

```xml
<?xml version = "1.0" encoding = "UTF-8" ?>
<sh:StandardBusinessDocumentHeader
>
<sh:HeaderVersion>1.0</sh:HeaderVersion>
<sh:Sender>
  <sh:ContactInformation>
    <sh:Contact>John Doe</sh:Contact>
    <sh:EmailAddress>John_Doe@purchasing.XYZretailer.com</sh:EmailAddress>
    <sh:FaxNumber>+1-212-555-1213</sh:FaxNumber>
    <sh:TelephoneNumber>+1-212-555-2122</sh:TelephoneNumber>
  </sh:ContactInformation>
</sh:Sender>
<sh:Receiver>
</sh:Receiver>
<sh:DocumentIdentification>
  <sh:TypeVersion>2.0</sh:TypeVersion>
  <sh:InstanceIdentifier>100002</sh:InstanceIdentifier>
  <sh:Type>OrderResponse</sh:Type>
  <sh:CreationDateAndTime>2011-08-22T11:31:52Z</sh:CreationDateAndTime>
</sh:DocumentIdentification>
<sh:BusinessScope>
  <sh:Scope>
    <sh:Type>BusinessProcess</sh:Type>
    <sh:InstanceIdentifier>ecae53d4-7473-45a6-ad70-61970dd7c4b0</sh:InstanceIdentifier>
    <sh:Identifier>cpa:123456789:192837465</sh:Identifier>
    <sh:BusinessService>
      <sh:BusinessServiceName>urn:www.cenbii.eu:profile:BII06:ver1.0</sh:BusinessServiceName>
      <sh:ServiceTransaction TypeOfServiceTransaction="RequestingServiceTransaction" IsAuthenticationRequired="true" IsNonRepudiationRequired="true" IsNonRepudiationOfReceiptRequired="true" IsIntelligibleCheckRequired="true" IsApplicationErrorResponseRequested="true" TimeToAcknowledgeReceipt="P12H" TimeToAcknowledgeAcceptance="P2D" TimeToPerform="P5D" Recurrence="3"/>
    </sh:BusinessService>
  </sh:Scope>
</sh:BusinessScope>
Using ebMS 3.0 AS4 and SBDH involves:

- SOAP 1.2 envelope;
- ebMS 3.0 AS4 Header that includes the Sender and Receiver Gateway information;
- the information contained by SBDH could be the receiver and the sender address, the type of the payload and the business scope.
- The specifications will be based on UN/CEFACT Standard Business Document Header (SBDH) standard;
- WS Security Header;
- the Payload Container an e-Documents that has capabilities to embed other e-Documents and it is content agnostic.

3.9.5. Figure 4. ebMS 3.0/AS4 using SBDH Scenario.

SBDH use cases

1. **Sending non-XML document(s)**
   When sending non-XML documents the SBDH and the payload have to be in separate MIME parts (see the figure above) because they have different content types: SBDH is XML and the payload non-XML.
   eDocuments solution is ASiC container that is a non-XML file (application/vnd.etsi.asic-e+zip content-type) so in order to correlate the Message exchange protocol solution to eDocuments the ASiC and SBDH cannot be in the same MIME part as they have different content-types. The container MIME part will be referenced using the URI tag of the Manifest group. The URI tag value will be a Content ID URI (RFC 2392) used generally to reference other body parts in the same message as the referring body part.

2. **Sending an XML document**
   The SBDH information can be packaged as a part of the business document-in a single MIME part, or for example as a separate part. There are many reasons why the implementer would choose an integrated packaging approach or a non-integrated approach. The following arguments favour the integrated approach:

   - If SBDH is an integral part of the XML instance document, the document can be parsed at a high level and routing and processing decisions can easily be made.
   - If the SBDH is contained in a separate body part, once the message is received by the Communications application, the linkage between the two body parts can be lost and the routing / processing functionality becomes more complex.
The AS4 header is part of the ebMS3 SOAP message. It is not a payload and it is processed by the AS4 Message Service Handler. The format and content of the AS4 user message header are similar to the header structure defined in the earlier ebMS 2.0 standard. The header allows:

- route or deliver messaging to specific back-end applications using delivery criteria;
- monitor business activity with specific partners, services, or business process;
- track messages based on AS4 headers only, and in a payload-agnostic fashion.

The following table provides a comparison between the AS4 Messaging Header and the SBDH, showing their similar functionalities:

<table>
<thead>
<tr>
<th>AS4 Header</th>
<th>SBDH</th>
</tr>
</thead>
<tbody>
<tr>
<td>The AS4 PartyInfo group contains information about the From and To parties. In this profile, they identify corners 3 and 4.</td>
<td>The corresponding SBDH elements are the Sender and Receiver elements. They identify corners 1 and 4.</td>
</tr>
<tr>
<td>The AS4 CollaborationInfo group contains an optional AgreementRef and mandatory Service, Action and ConversationId elements.</td>
<td>The optional BusinessScope group in the SBDH and the related BusinessScope schema provide the elements BusinessServiceName and ServiceTransaction that have a similar purpose.</td>
</tr>
<tr>
<td>The optional MessageProperties group contains a series of arbitrary name/value properties.</td>
<td>SBDH has a similar extensibility mechanism based on XML schema type substitution.</td>
</tr>
<tr>
<td>The AS4 PayloadInfo group contains information about the business document, or business documents and any attachments to those documents. The payloads themselves are stored in separate MIME parts in the AS4 MIME message and referenced via the href attribute.</td>
<td>In SBDH, the Manifest group is used for (non-XML) attachments. The SBDH itself is part of a standard business document, i.e. an XML payload. Attachments can be in separate MIME parts as is the case in AS4.</td>
</tr>
</tbody>
</table>

**AS4 header or SBDH**

AS4 Header provides most SBDH features so a scenario without SBDH should be considered:

- SOAP 1.2 envelope;
- ebMS 3.0 AS4 Header;
- WS Security Header;
- the Payload Container is an e-Document that has capabilities to embed other e-Documents and is content-agnostic.

![AS4 header or SBDH diagram](image-url)

**Figure 5. ebMS 3.0/AS4 Scenario.**

**Correlation**
AS4 provides multiple mechanisms to correlate messages within a particular flow.

1. **UserMessage/MessageInfo/RefToMessageId** provides a way to express that a message is a response to a single specific previous message. Presence of a RefToMessageId is required in response messages in Two Way message exchanges. By default, exchanges are considered One Way.

2. **UserMessage/CollaborationInfo/ConversationId** provides a more general way to associate a message with an ongoing conversation, without requiring a message to be a response to a single specific previous message, but allowing update messages to existing conversations from both Sender and Receiver of the original message.

The ebMS3 and AS4 specifications do not constrain the use of the elements RefToMessageId and ConversationId, but the following rule shall apply:

1. **UserMessage/MessageInfo/RefToMessageId** is to be used to support message exchanges that are modelled as request-response interactions. In the response message, the value of the element must be set to the value of the UserMessage/MessageInfo/MessageId element in the request message.

2. **UserMessage/CollaborationInfo/ConversationId** must be included in any AS4 message (as it is a mandatory element). Its value is to be defined in sub-profiles of this e-SENS profile for specific domains.

### 3.9.6. Security

This profile is intended to support exchange of AS4 messages using either the public Internet or private networks. When using the public Internet, each organization is individually responsible to implement security measures to protect access to its IT infrastructure. Data exchange may use IPv4 or IPv6.

Organizations may use firewalls to restrict incoming or outgoing message flows to specific IP addresses, or address ranges. Organizations therefore:

- Must use static IP addresses (or IP address ranges) for inbound and outbound AS4 HTTPS connections.
- Must communicate all IP addresses (or IP address ranges) used for outgoing and incoming connections to their trading partners, also covering any passive nodes in active-passive clusters. Note that the address of the HTTPS endpoint which an AS4 server is to push messages to or pull messages from may differ from the address (or addresses) used for outbound connections.
- Must notify about any IP address changes sufficiently in advance to allow firewall and other configuration changes to be applied.

The **Transport Layer Security** settings defined in section 1.5.3.5.1 may be implemented in the AS4 communication server but TLS may also be offloaded to a separate infrastructure component (such as a firewall, proxy server or router). In that case, the recommendations on TLS version and cipher suites [OSSLTLS] of 1.5.3.5.1 must be addressed by that component.

The TLS cipher suites recommended in section 1.5.3.5.1 are supported in recent versions of TLS toolkits and which therefore are available for use. Support for these suites is recommended. Whether or not less secure cipher suites (which are only recommended for legacy applications) are allowed is a local policy decision.

This profile does not require the use of client authentication. Client authentication may be a requirement in the networking policy of individual organizations that the AS4 deployment needs to meet, but is not recommended.

The following parameters control configuration of security at the message layer:

- The `PMode[1].Security.X509.Signature.Certificate` parameter must be set to a value matching the certificate of the sender.

This profile provides Non Repudiation of Origin and Receipt at the level of Message Exchange. Note that the use of non-repudiation information to resolve any disputes requires the communication partners (in a four-corner model, these are C2 and C3) to store data such as the exchanged AS4 messages, AS4 receipts corresponding to these messages, and possibly other verification data such as OCSP responses, for the period during which any such disputes may arise. The minimum retention period to be applied should be specified in a formal policy and be consistent with legal-regulatory requirements, business needs, and available storage/processing capacities. The AS4 gateways should be configured accordingly.

### 3.9.7. Message Payload and Flow Profile

A single AS4 UserMessage must reference, via the **PayloadInfo** header, a single structured business document and may reference one or more other (structured or unstructured) payload parts. The business document is considered the "leading" payload part for business processing. Any payload parts other than the business document are not to be processed in isolation but only as adjuncts to the business document. Business document, attachments and metadata must be submitted and delivered as a logical unit. The format of the business document should be XML, but other data types may be supported in specific business processes or contexts.

When using an SBDH, the SBDH is the initial leading document, which in turn references (or includes) the XML business document. Any other payload parts should be referenced from the SBDH, in addition to being referenced from the AS4 header.

For each business process, the Business Requirement Specification must specify the XML schema definition (XSD) that the business document must conform to. The mapping from **Service** and **Action** value pairs to XSDs must be unique, allowing Receivers to validate XML documents using a specific XML schema.

While the AS4 protocol has no inherent limitation on message size, users should take into consideration the practical limits that are imposed by some AS4 implementations (e.g. the use of certain data types in some security libraries imposes a maximum size of approximately 2 GB) or by deployments of AS4 implementations (e.g. amount of memory or storage available to the AS4 server).
3.9.8. Test Service

Section 5.2.2 of [EBMS3] defines a server test feature that allows an organization to “Ping” a communication partner. The feature is based on messages with the values of:

- UserMessage/CollaborationInfo/Service set to http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/service
- UserMessage/CollaborationInfo/Action set to http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/test

This feature must be supported so that business partners can perform a basic test of the communication configuration (including security at network, transport and message layer, and reliability) in any environment, including the production environment. This functionality may be supported as a built-in feature of the AS4 product. If not, a PMode must be configured with these values. The AS4 product must be configured so that messages with these values are not delivered to any business application.

3.9.9. Environments

B2B data exchange solutions are part of the overall IT service lifecycle, in which different environments are operated (typically in parallel) for development, test, pre-production (in some companies referred to as “acceptance environments” or “QA environments”) and production. Development and test are typically internal environments in which trading partners are simulated using stubs. When exchanging messages between organizations (in either pre-production or production environments), they must target the appropriate environment. In order to prevent a configuration error from causing non-production messages to be delivered to production environments or vice versa, organizations should configure processing modes at message handlers so that messages from one type of environment cannot be accepted inadvertently by a different type of environment.

3.9.10. Example

The following (non-normative) example contains the SOAP envelope of an AS4 message from a Seller to a Buyer in an e-procurement scenario involving the exchange of an order response document. The example uses end entities and gateways identified using GS1 GLN numbers encoded as ebCore Party Identifiers. Binary or other (for humans) meaningless text has been replaced by a range of @ symbols.

```
c--799398144.1400668830402.seller.eu
Content-Type: application/soap+xml; charset="UTF-8"
Content-Transfer-Encoding: binary
Content-ID: <320758080.1400668830402@seller.eu>

<S12:Envelope xmlns:S12="http://www.w3.org/2003/05/soap-envelope"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3.org/2003/05/soap-envelope"
>  
<S12:Header xmlns:eb="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/">
  <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
    S12:mustUnderstand="true">
      <xenc:EncryptedKey Id="EncKeyId-3A989A4B55B6996A61400668830398132">
```

```
<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <wsse:SecurityTokenReference>
        <ds:X509Data>
            <ds:X509IssuerSerial>
                <ds:X509IssuerName>CN=Some Certification Authority,
                          OU=Issuing Certification Authority,
                          O=Some Org,C=NL</ds:X509IssuerName>
                <ds:X509SerialNumber>@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
<eb:Messaging xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:ns2="http://docs.oasis-open.org/ebxml-bp-signals-2.0"
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:xlink="http://www.w3.org/1999/xlink" xmlns:xs="http://www.w3.org/2001/XMLSchema"
wsu:Id="id-126" S12:mustUnderstand="true" S12:role="">
  <eb:UserMessage>
    <eb:MessageInfo>
      <eb:MessageId>XRU30wpy5Zsu4kQzB8NeSKNkaiE@seller.eu</eb:MessageId>
    </eb:MessageInfo>
    <eb:PartyInfo>
      <eb:From>
        <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088">1234567890</eb:PartyId>
        <eb:Role>Seller</eb:Role>
      </eb:From>
      <eb:To>
        <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088">0987654321</eb:PartyId>
        <eb:Role>Buyer</eb:Role>
      </eb:To>
    </eb:PartyInfo>
    <eb:CollaborationInfo>
      <eb:Service>http://esens.eu/services/eprocurement/1.0</eb:Service>
      <eb:Action>ConfirmOrder</eb:Action>
      <eb:ConversationId>E5D7CFEE-E6A9-4855-A67E-6C24403E35E6</eb:ConversationId>
    </eb:CollaborationInfo>
    <eb:MessageProperties>
      <eb:Property name="originalSender" type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088">5209999001264</eb:Property>
      <eb:Property name="finalRecipient" type="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088">5209999001295</eb:Property>
      <eb:Property name="trackingIdentifier" order1235@endentity.it</eb:Property>
    </eb:MessageProperties>
  </eb:UserMessage>
</eb:Messaging>
# 3.10. Processing Mode Parameters

The following table summarizes the PMode settings as defined in this specification.

<table>
<thead>
<tr>
<th>Processing Mode Parameter</th>
<th>Value in the E-SENS Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMode.ID</td>
<td>Not profiled</td>
</tr>
<tr>
<td>PMode.Agreement</td>
<td>Not profiled</td>
</tr>
<tr>
<td>PMode.MEP</td>
<td><a href="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay">http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/oneWay</a></td>
</tr>
<tr>
<td></td>
<td><a href="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/twoWay">http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/twoWay</a></td>
</tr>
<tr>
<td>PMode.MEPBinding</td>
<td><a href="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/push</a></td>
</tr>
<tr>
<td></td>
<td><a href="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/pushAndPush">http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/pushAndPush</a></td>
</tr>
<tr>
<td>PMode.Initiator.Party</td>
<td>Identifier (and possibly a type attribute value) compliant with e-SENS Addressing ABB</td>
</tr>
<tr>
<td>PMode.Initiator.Role</td>
<td>Not profiled</td>
</tr>
<tr>
<td>PMode.Initiator.Authorization.username</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode.Initiator.Authorization.password</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode.Responder.Party</td>
<td>Identifier (and possibly a type attribute value) compliant with e-SENS Addressing ABB</td>
</tr>
<tr>
<td>PMode.Responder.Authorization.username</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode.Responder.Authorization.password</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode[1].Protocol.SOAPVersion</td>
<td>1.2</td>
</tr>
<tr>
<td>PMode[1].BusinessInfo.Service</td>
<td>SHOULD identify a set of related business transactions or other message exchanges in the context of a business process or use case.</td>
</tr>
<tr>
<td>PMode[1].BusinessInfo.Action</td>
<td>SHOULD identify the different types of business transactions or other message exchanges in the context of an identified Service. This MAY be an identifier of a document type, if the exchange of a document of that type unambiguously identifies the purpose and requested action in the context of the Service.</td>
</tr>
<tr>
<td>PMode[1].BusinessInfo.Properties</td>
<td>Support required. In four corner exchanges, mandatory inclusion of originalSender and finalRecipient and optional inclusion of trackingIdentifier.</td>
</tr>
<tr>
<td>PMode[1].BusinessInfo.MPC</td>
<td>Not profiled</td>
</tr>
<tr>
<td>PMode[1].BusinessInfo.PayloadProfile</td>
<td>Not profiled</td>
</tr>
<tr>
<td>PMode[1].Errorhandling.Report(SenderErrorsTo)</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode[1].Errorhandling.Report(ReceiverErrorsTo)</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode[1].Errorhandling.Report.AsResponse</td>
<td>True</td>
</tr>
<tr>
<td>PMode[1].Errorhandling.DeliveryFailuresNotifyProducer</td>
<td>True (Recommended)</td>
</tr>
<tr>
<td>PMode[1].Reliability</td>
<td>Not used</td>
</tr>
<tr>
<td>PMode[1].Security.WSSversion</td>
<td>1.1.1</td>
</tr>
</tbody>
</table>
### 4. Ownership

The AS4 Profile of ebMS 3.0 Version 1.0 technical specification is Copyright © OASIS Open 2013. All Rights Reserved. The AS4 Profile of ebMS 3.0 Version 1.0 technical specification is created by the OASIS ebXML Messaging Services Technical Committee which operates under the RF on Limited Terms Mode of the OASIS IPR Policy.

AS4 is based on the OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features OASIS Standard, which is *Copyright © OASIS 1993–2007. All Rights Reserved.* The ebMS 3.0 Standard uses the SOAP protocol. The IPR declaration of SOAP submitters to W3C is available from [http://www.w3.org/Submission/2000/05/](http://www.w3.org/Submission/2000/05/).

Parts of the implementation guidelines in this document are derived, with permission, from parts of the ENTSOG AS4 Profile for TSOs. ENTSOG can be contacted at [http://www.entsog.eu/publications/data-exchange](http://www.entsog.eu/publications/data-exchange).

Parts of the implementation guidelines in this document are derived, with permission, from e-CODEX specifications. E-CODEX can be contacted as follows:

Ministry of Justice NRW, Martin-Luther-Platz 40 - 40212 Düsseldorf (GERMANY).

All other content of this specification was created in the former EU e-SENS project. The EU e-SENS project formally transferred ownership of its specifications to the European Commission, which accepted it for further maintenance in the context of the CEF Programme. Information on governance and procedures for eDelivery is available from [Governance and Procedures](http://www.entsog.eu/publications/data-exchange).

### References
6. Contributors

The CEF eDelivery team is in charge of maintaining the current version of the specifications.

Details about contributors of previous versions of this specification can be consulted on the e-SENS website.

7. History

<table>
<thead>
<tr>
<th>Ver.</th>
<th>Date</th>
<th>Changes made</th>
<th>Modified By</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.12</td>
<td>18.10.2017</td>
<td>Minor update as part of the migration of the specification</td>
<td>Gianmarco Piva,</td>
</tr>
</tbody>
</table>
CEF Digital, not requiring external review.

Layout changed to match CEF Digital template

Content copied from e-SENS architecture, with the following minor editorial changes and bibliographic updates:

- Set the version number to 1.12.
- Switched from e-SENS specification metadata to the CEF eDelivery specification metadata: Publication Date, Obsoletes, Obsoleted by.
- Removed the Standardization and Sustainability Assessment section.
- Updated the Contributor section, historical contribution information is left to the last referenced e-SENS project version.
- Updated the Ownership section to include the handover of the ownership of content from the e-SENS project to CEF Digital.
- Started with a clean history table (the e-SENS history is still available in the 1.11 version).
- Updated internal links to SMP to point to the CEF versions.
- Removed hyperlinks from some URIs defined in ebMS3 that are identifiers only and not hyperlinks and caused HTTP 404 errors.
- Updated the BDXL and SMP bibliographic references to reflect their current OASIS Standard status and to use their recommended citation formats.
- Removed e-SENS bibliographic references to deliverables D6.1 and D3.2, which were not referenced.
- Updated the reference for Domibus and update text to reflect its transfer from the e-CODEX project to the European Commission.
- Updated the reference for the ENTSOG AS4 Usage Profile.
- Updated the bibliographic reference to the e-SENS e-Confirmation pilot to point to the public e-SENS site.
- Removed the bibliographic reference of the e-SENS e-Invoicing pilot, as it was not referenced.
- Updated the reference for e-CODEX use of message properties for end entity addressing to the public D5.11.
- Added the URL to XML Signature 1.1, 2nd edition.
- Renamed the section “Specifications” to “Base Specification”.
- Changed “PR” to “e-SENS”.

Details about previous versions of the specifications can be consulted on the e-SENS website.