ISO 20022
Business Application Header
Message Usage Guide
Version 1.8

Date: April 2016

This Message Usage Guide for the Business Application Header was drafted by the ISO 20022 Technical Support Group and approved by the Registration Management Group.

1 Table of Contents

1.1 Purpose and Use of this Guide 3

1.2 Intended Audience 3

1.3 Terminology 3

1.4 How this Guide was created 3

1.5 The ISO 20022 Standard 3

1.6 Separation of layers 4

1.7 Link between a Business Application Header and its Message 5

1.8 Related Documents and Guides 5

2 Scenarios 7

2.1 Introduction 7

2.2 Business Application Header deployment options 7

2.3 Business Application A sends a Business Message to Business Application B 8

2.4 There may be several CreationDate elements but they may not have the same definition. 10

2.5 Business Application A informs Business Application B that Text based MessageElements (in the BAH or the BusinessMessage) may contain non-Basic-Latin characters 11

2.6 Business Application A informs Business Application B of the Business Service within which this BusinessMessage is exchanged. 12

2.7 Business Application A suspects Business Application B has not received the BusinessMessage 13

2.8 Business Application A sends a copy of a previously sent Business Message. 14

2.9 Business Application A sends a duplicate of a previously sent Business Message. 15

2.10 Business Application A sends a duplicate of a previously sent copy of a Business Message.16

2.11 Business Application A sends a Business Message B that relates to BusinessMessage A, but which is not a duplicate or a copy. 17

2.12 Business Application A sends a Business Message to BusinessApplication B with a pre-agreed priority 18

2.13 Business Application A sends a signed Business Message to BusinessApplication B 19

3 Mapping of the BAH to other headers 20

3.1 Introduction 20

3.2 BAH to SWIFTNet Application Header 20

3.3 BAH to ebXML/ebMS Header 22

3.4 BAH to FpML 23

4 ANNEX A - Signature Guidelines 24

4.1 Preface 24

4.2 Introduction 24

4.2.1 Current Signature Guidelines 24

4.2.2 Short introduction to W3C Signatures 25

4.2.3 Scope of this documents Signature Guidelines 26

4.3 Structure of the Signature 27

4.4 Process for signing and verifying 31

4.5 Example 34

INTRODUCTION

1.1 Purpose and Use of this Guide

This guide explains how to use the ISO 20022 Business Application Header (BAH) in the context of the business processes it addresses. It provides a comprehensive view of how the Business Application Header complements any ISO 20022 Message. This guide acts as a supplement to the Message Definition Report and the XML schema, which are published on the ISO 20022 website (www.iso20022.org).

The guide provides information regarding the implementation of the Business Application Header in any relevant general context. Additional documents, published by individual user communities, may be available that discuss the implementation of the BAH in a more specific context.

This guide should serve as the general basis for the more specific community implementation guides that are developed.

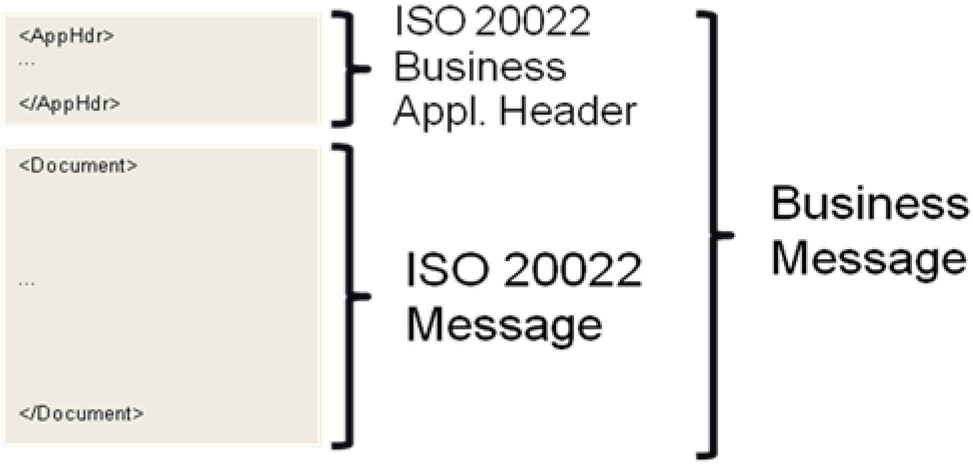
Currently, the included descriptions and examples that are used in this document are based exclusively on the ISO 20022 XML syntax. In the future, there may be a requirement to include descriptions or examples in other syntaxes such as ASN.1 depending on the demand for this in the ISO 20022 community.

1.2 Intended Audience

Both business people and message developers can use this guide.

1.3 Terminology

An ISO 20022 Message together with its Business Application Header forms a Business Message.



1.4 How this Guide was created

This guide was created through the combined efforts of the ISO 20022 Standards Evaluation Groups (SEG) for the collection of the requirements, the ISO 20022 Technical Support Group (TSG) for drafting the solution and the ISO 20022 Registration Management Group (RMG) for approving this document.
Maintenance will be occurring through the TSG.

1.5 The ISO 20022 Standard

ISO 20022 is owned by the International Organization for Standardization (ISO) under Technical Committee 68 (TC68), which is the Financial Services Technical Committee of ISO.

Complete information on the ISO 20022 standard can be found on www.iso20022.org.

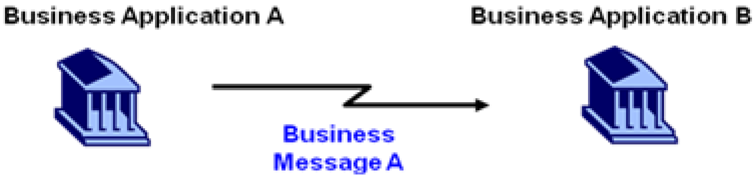
For more information on ISO itself, please see www.iso.org.

1.6 Separation of layers

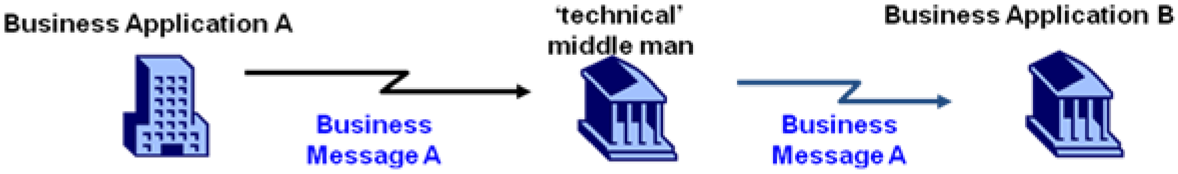
ISO 20022 messages and the BAH are designed to be transport protocol independent. The ISO 20022 standard does not provide any message transport conventions of its own (including header or trailer).

The Business Application Header is a business header and should not be confused with a file or transport header. It is created before the transport routing header is applied to the business message and is retained after the transport header is removed.
So any parties between the two business applications that don't perform a business function are not mentioned in the BAH. Such 'technical' middle men don't open or change the Business Message; they only forward it to the correct business application.

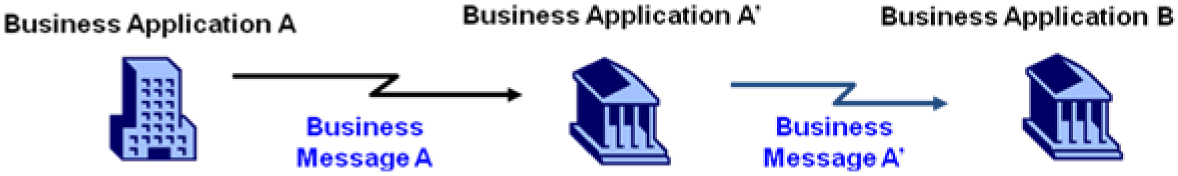
So, a transport scenario like below



is from a business point of view the same as a transport scenario like this:



However, as soon as a Business Application is in the middle (i.e. an Application that processes the Business Message), it is identified as the recipient in the BAH and therefore will send a different business message.

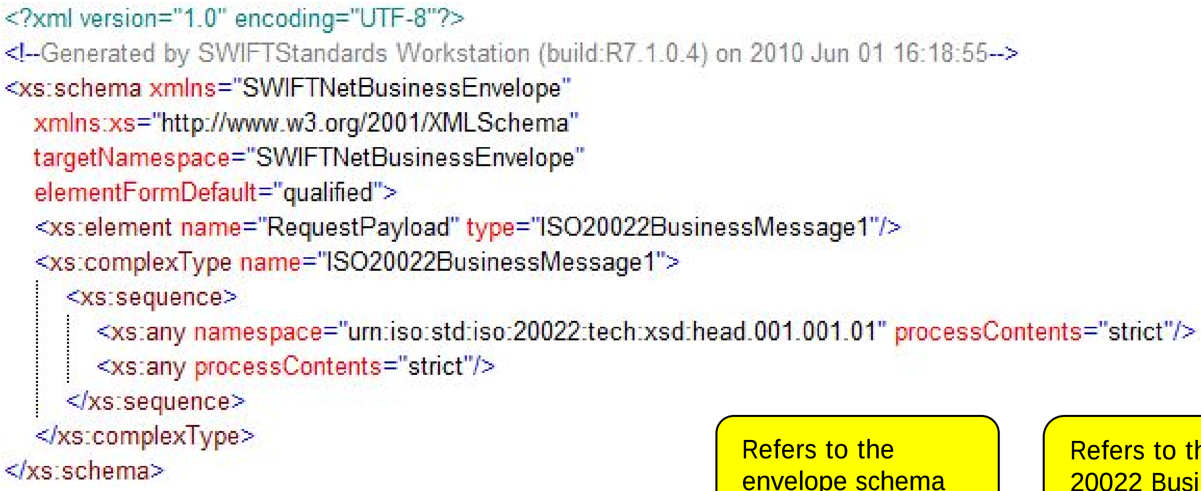


Although the BAH is not the transport header, data in the BAH can be used by transport applications to determine the routing header since it does contain the business sender, receiver and document details. It can also be used by the business applications to determine the appropriate process to perform on the business message.

1.7 Link between a Business Application Header and its Message

The name of envelope element that binds a Business Application Header to the ISO 20022 Message to which it applies is implementation/network specific. In any case, the BusinessApplicationHeader root element AppHdr and the ISO 20022 MessageDefinition root element Document must always be sibling elements in any XML document.
The AppHdr element must be located before the Document element.

Example:
Below XML Schema describes an element RequestPayload (used on SWIFTNet) that contains two elements: AppHdr and any other element.



```
<?xml version="1.0" encoding="UTF-8"?>
<!--Generated by SWIFTStandards Workstation (build:R7.1.0.4) on 2010 Jun 01 16:18:55-->
<xs:schema xmlns="SWIFTNetBusinessEnvelope"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   targetNamespace="SWIFTNetBusinessEnvelope"
   elementFormDefault="qualified">
   <xs:element name="RequestPayload" type="ISO20022BusinessMessage1"/>
   <xs:complexType name="ISO20022BusinessMessage1">
      <xs:sequence>
         <xs:any namespace="urn:iso:std:iso:20022:tech:xsd:head.001.001.01" processContents="strict"/>
         <xs:any processContents="strict"/>
      </xs:sequence>
   </xs:complexType>
</xs:schema>
```

A valid XML instance would be structured as follows:

```
<Env:RequestPayload xmlns:Env=" SWIFTNetBusinessEnvelope" >
          <Ah:AppHdr xmlns:Ah=" urn:iso:std:iso:20022:tech:xsd:head.001.001.01">
          ...
          </Ah:AppHdr>
          <Doc:Document xmlns:Doc=" urn:iso:std:iso:20022:tech:xsd:xxxx.nnn.nnn.nn">
          ...
          </Doc:Document>
</Env:RequestPayload>
```

Refers to the envelope schema

Refers to the ISO 20022 Business Application Header

Refers to the ISO 20022 Message

1.8 Related Documents and Guides

The complete catalogue of ISO 20022 messages, including the Message Definition Reports and XML schemas, is available on the ISO 20022 website: www.iso20022.org. Current and historical versions of the schemas are available free of charge. Other useful documentation available from the ISO 20022 website includes:

- ISO 20022 Financial Repository - Data Dictionary.
- Introduction to ISO 20022 – Universal financial industry message scheme. An introductory PowerPoint on the ISO 20022 standard family.

Useful documents are available from the following sources:

- An in-depth knowledge of XML can be found at: http://www.w3c.org/TR/2000/REC-xml-20001006

- An in-depth knowledge of XML Schema can be found at:
  http://www.w3c.org/TR/xmlschema-0/, http://www.w3c.org/TR/xmlschema-1/
  and
  http://www.w3c.org/TR/xmlschema-2/
- The UNICODE character set database can be found at:
  http://unicode.org/Public/UNIDATA/Blocks.txt

2 Scenarios

2.1 Introduction

This section explains how to use the BAH in specified scenarios. The list of scenarios is not exhaustive and new scenarios may be added.

2.2 Business Application Header deployment options

The choice of whether to send a BAH with a given message is taken by each implementing community.

Depending on how a given message was originally designed and/or any maintenance changes that may have occurred over time, an implementing community can be confronted with the following deployment options:

Message deployment	ISO 20022 message ...	
	with duplicated BAH elements	without duplicated BAH elements[1]
with BAH	Business rules and market practice should determine the use of BAH elements in the BAH itself and any duplicated BAH elements in the ISO 20022 message	Only BAH elements within BAH are available for use
without BAH	Only BAH elements duplicated within ISO 20022 message are available for use	BAH elements are not available for use, neither in ISO 20022 message nor through the BAH

In the case of ISO 20022 messages without duplicated BAH elements, message deployment without the BAH can produce challenging situations (i.e. bottom right scenario in above table).

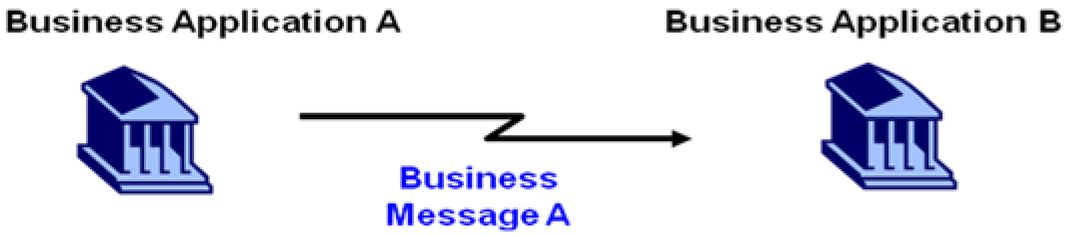
In these scenarios, essential BAH elements like the unique message identifier (BAH element: BusinessMessageIdentifier) or the message creation date (BAH element: CreationDate) cannot be conveyed through the ISO 20022 message, nor through the BAH.

If for certain implementation communities the deployment of the BAH does not constitute a valid choice, implementation communities are requested to get in contact with the ISO 20022 Registration Authority (RA) for advice on how to proceed in order to employ a fully ISO 20022-compliant implementation (e-mail: iso20022ra@iso20022.org).

Implementation communities may have alternative means to carry this information through non-ISO 20022 mechanisms (e.g. a network header). It should be noted that, in case of non-ISO 20022 mechanisms, not all scenarios covered in this BAH MUG can be fully implemented. For further information see chapter 2.3.

[1] ISO 20022 messages without duplicated BAH elements are labelled, in the Business domain catalogues on the ISO 20022 web-site, with the sentence “The message definitions below are intended for use with the Business Application Header.”.

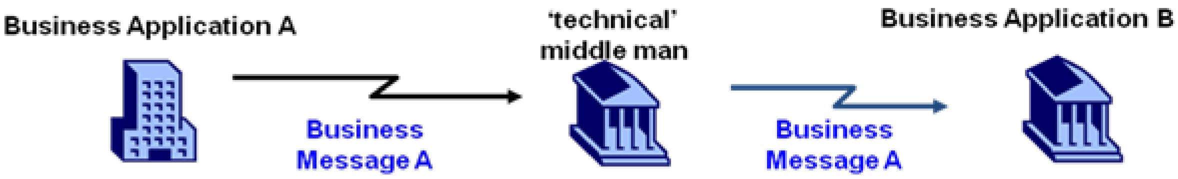
2.3 Business Application A sends a Business Message to Business Application B



Above is the most common, vanilla scenario, i.e. no special features of this BAH are used.

When there is a 'middle man' between the two Business Applications, it is the function/role of that middle man that will determine whether the Business Message from the middle man to Business Application B is a new Business Message.

If the middle man only forwards the Business Message, i.e. it does not process the Business Message, then only the transport header changes, but the Business Message (with its BAH) remains the same.



Following MessageElements must be used:

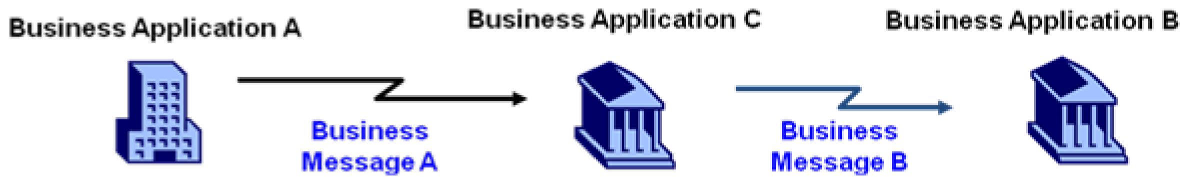
MessageElement Name	Usage
CharacterSet	
From	**Id of BusinessApplication A**
To	**Id of BusinessApplication B**
BusinessMessageIdentifier	**Identification of the BusinessMessage A**
MessageDefinitionIdentifier	**Identification of the MessageDefinition**
BusinessService	
CreationDate	**Date (and time) of the creation of this BusinessApplicationHeader and BusinessMessage**
CopyDuplicate	
PossibleDuplicate	
Priority	
Signature	
Related	

Note: In the case of ISO 20022 messages without duplicated BAH elements, the above 'technical middle man' scenario can pose implementation challenges, if the implementer community has chosen not to use the BAH.

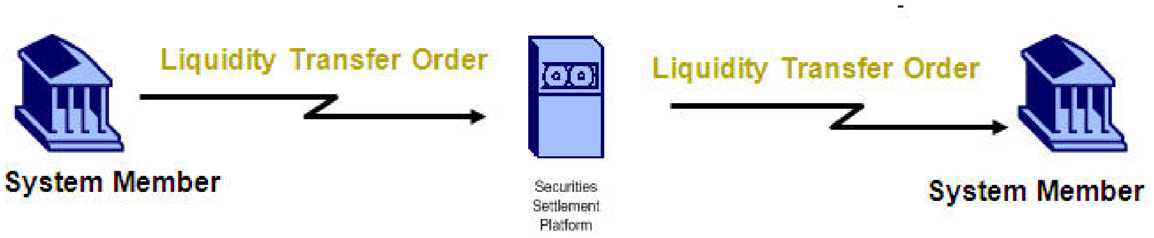
If the 'technical middle man' were to realise the link between two separate networks (e.g. each with its distinct network header), it may be that the non-ISO 20022 mechanisms employed are not sufficiently compatible in order for the business elements, otherwise conveyed through the BAH and/or the ISO 20022 message, to travel from Business Application A to Business Application B.

In order to avoid such situation, the process outlined in chapter 2.2 can be employed to realise a fully ISO 20022-compliant implementation without requiring the mandatory deployment of the BAH.

If the middle man processes the Business Message then the middle man is considered a Business Application and hence a new Business Message is created and sent to Business Application B. (see scenario 2.11)



EXAMPLE
A Securities Settlement platform serves as technical middle man



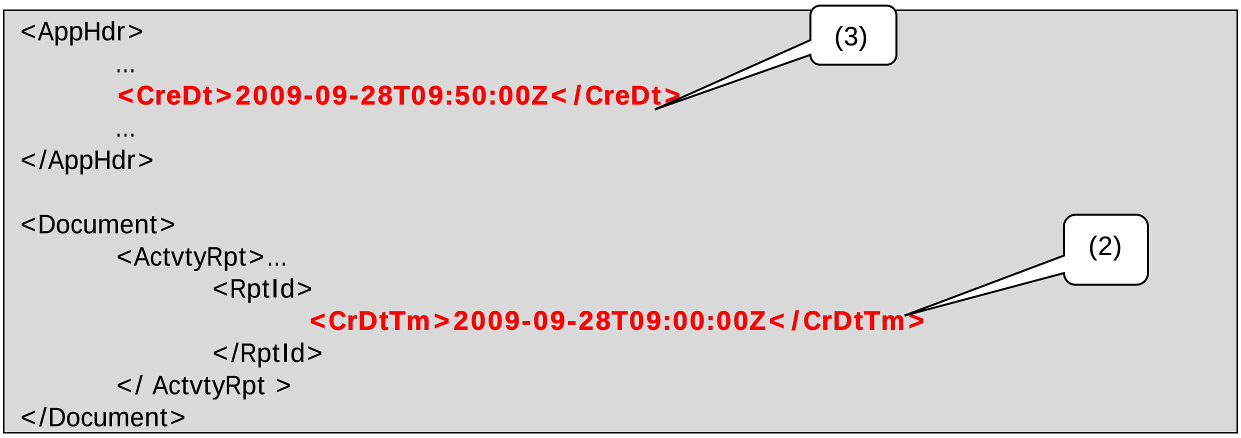
```
<AppHdr>
        <Fr>System Member A</Fr>
        <To>System Member B</Fr>
        …
        <BizSvc>S&R</BizSvc>
        …
</AppHdr>

<Document>
        <LqdtyCdtTfr>…..</LqdtyCdtTfr>
</Document>
```

2.4 There may be several CreationDate elements but they may not have the same definition.

As a general rule, the CreationDate in the BusinessApplicationHeader contains the date and time at which the Business Message was produced by the BusinessApplication. The term BusinessApplication must be interpreted in a broad sense: it may be a payments factory application; it may also be an operator at a screen who is manually inputting the business information.
Depending on the definition that is given to a CreationDate in the Document itself, the date and time could vary from the date and time in the Business Application Header. The creation date in the BAH applies to the entire Business Message whereas other creation dates apply to only parts of the Businesss Message.

EXAMPLE

1. John Doe at Bank A prepares the payments
2. the same person creates an activity report on August 28th at 09:00AM UTC.
3. the same person signs it at 9:50AM UTC with the signature in the BAH
4. The BAH contains the CreationDate of the BusinessMessage, the Document contains the CreationDate of the activity report.



```
<AppHdr>
        ...
        <CreDt>2009-09-28T09:50:00Z</CreDt>
        ...
</AppHdr>

<Document>
        <ActvtyRpt>...
                <RptId>
                        <CrDtTm>2009-09-28T09:00:00Z</CrDtTm>
                </RptId>
        </ ActvtyRpt >
</Document>
```

2.5 Business Application A informs Business Application B that Text based MessageElements (in the BAH or the BusinessMessage) may contain non-Basic-Latin characters

In this case, each additional character set will be specified in the CharacterSet MessageElement, separated by a semicolon.
All relevant Text based Datatypes may then contain characters belong to the character sets specified in this MessageElement.
Some Text based DataTypes may be further constraint than what is specified here in which case the character set restrictions specified in the BAH do not apply.

Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	**character set 1;character set 2**
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader (and BusinessMessage)
CopyDuplicate	
PossibleDuplicate	
Priority	
Signature	
Related	

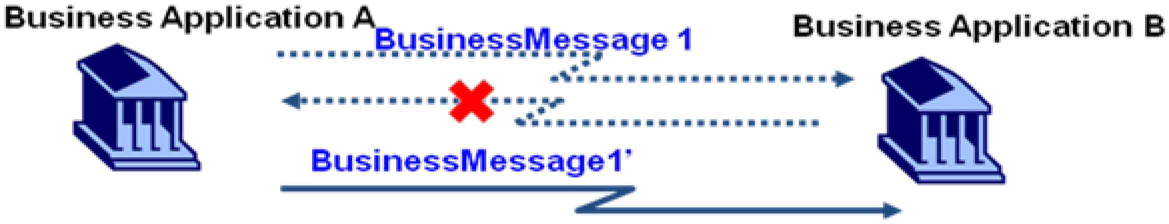
2.6 Business Application A informs Business Application B of the Business Service within which this BusinessMessage is exchanged.

This enables to Receiver to unambiguously relate the BusinessMessage to the BusinessService in which it is used.
Normally this MessageElement is used when this BusinessMessage is used in multiple services.
It can be the service identified by the service provider or a bilaterally / multilaterally agreed service.

Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	**Identification of the Service within which this Message is exchanged.**
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader (and BusinessMessage)
CopyDuplicate	
PossibleDuplicate	
Priority	
Signature	
Related	

2.7 Business Application A suspects Business Application B has not received the BusinessMessage

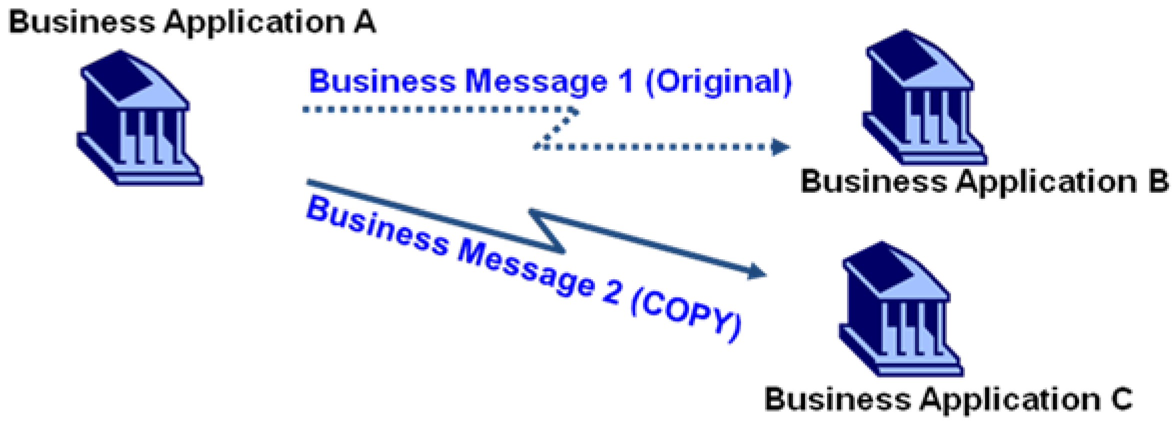


PossibleDuplicate is used when the Business Application that sent the message hasn't received any reply (because of technical or other problems).
It will therefore resent THE SAME BusinessMessage, adding the relevant header elements from the original BusinessMessage in the 'Related' MessageElement.
If the receiver did receive the original BusinessMessage identified in the 'Related', then this BusinessMessage MUST BE IGNORED.
If the receiver did NOT receive the original BusinessMessage, then it must treat this BusinessMessage as if it was the original BusinessMessage.

Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader
CopyDuplicate	
PossibleDuplicate	**YES**
Priority	
Signature	
Related	**Copy of the relevant MessageElements of the BusinessApplicationHeader of the original BusinessMessage that was sent to Business Application B.**

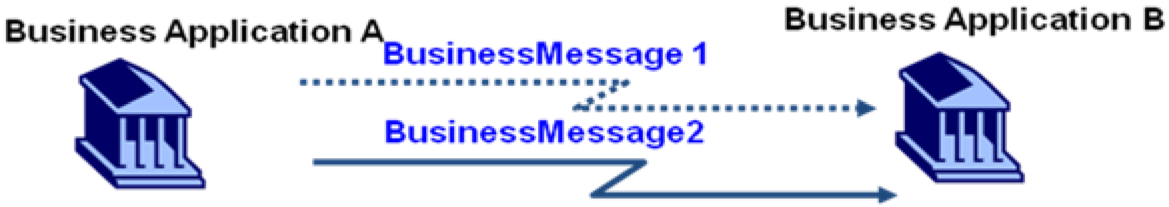
2.8 Business Application A sends a copy of a previously sent Business Message.



Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	**Id of BusinessApplication C**
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader
CopyDuplicate	**COPY**
PossibleDuplicate	
Priority	
Signature	
Related	**Copy of the relevant MessageElements of the BusinessApplicationHeader of the original BusinessMessage sent to Business Application B**

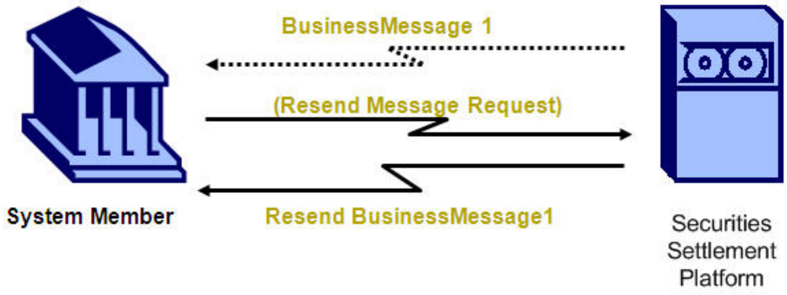
2.9 Business Application A sends a duplicate of a previously sent Business Message.



Following MessageElements must be used:

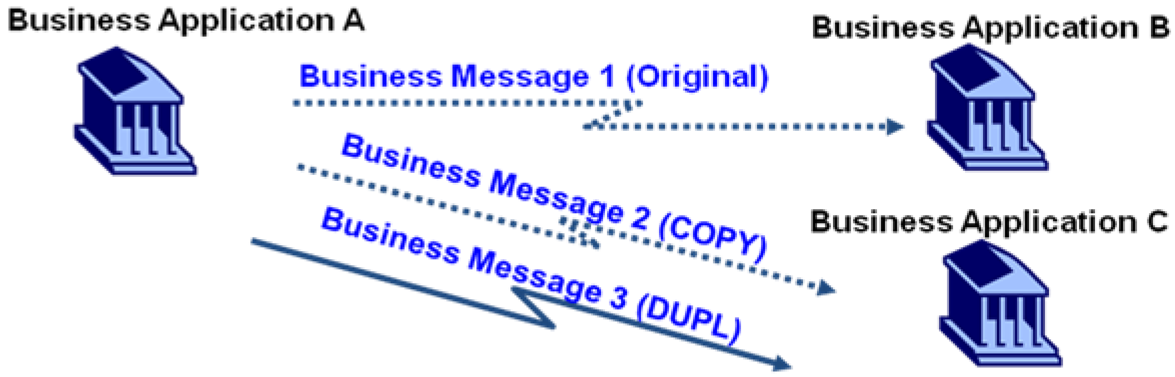
MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader
CopyDuplicate	**DUPL**
PossibleDuplicate	
Priority	
Signature	
Related	**Copy of the relevant MessageElements of the BusinessApplicationHeader of the original BusinessMessage**

EXAMPLE
Upon request, the Securities Settlement platform resends BusinessMessage1



```
<AppHdr>
        <Fr>Securities Settlement Platform</Fr>
        <To>System Member </Fr>
        ...
        <CpyDplct>DUPL</ CpyDplct>
        <Rltd>copy of original BAH</Rltd>
        ...
</AppHdr>
<Document>
        <LqdtyCdtTfr>.....</LqdtyCdtTfr>
</Document>
```

2.10 Business Application A sends a duplicate of a previously sent copy of a Business Message.


Business Application A
Business Application B
Business Application C
Business Message 1 (Original)
Business Message 2 (COPY)
Business Message 3 (DUPL)

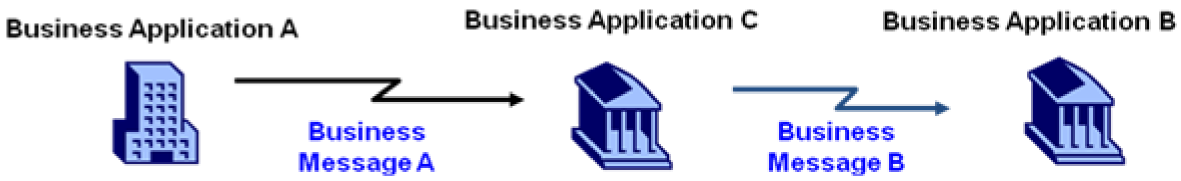
Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader
CopyDuplicate	**CODU**
PossibleDuplicate	
Priority	
Signature	
Related	**Copy of the relevant MessageElements of the BusinessApplicationHeader of the copy BusinessMessage**

2.11 Business Application A sends a Business Message B that relates to BusinessMessage A, but which is not a duplicate or a copy.

In this case, the BusinessApplicationHeader of the related BusinessMessage must be mentioned in the Related element.
This scenario is used when it is relevant for the recipient to know where the BusinessMessage A came from that triggered the creation of this Business Message B. It will show in the Related element who created the original BusinessMessage (in the From element of the Related element).
CopyDuplicate and PossibleDuplicate are not used.

For example when there is a 'middle man' that does process the Business Message (and as such is an active business partner in the transaction) as opposed to a middle man that only technically forwards a BusinessMessage and as such is not involved in a business transaction and therefore not mentioned in the BusinessMessage (see also chapter 2.3)



Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication C
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage B
MessageDefinitionIdentifier	Identification of the MessageDefinition B
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessMessage
CopyDuplicate	
PossibleDuplicate	
Priority	
Signature	
Related	**Copy of the relevant MessageElements of the BusinessApplicationHeader of BusinessMessage A**

2.12 Business Application A sends a Business Message to BusinessApplication B with a pre-agreed priority

The standard doesn't define the meaning of the values. The meaning is service/market/business area depended and must be pre-agreed within a specific context like the BusinessService, BusinessArea, etc.
Where required, the BusinessService must be used to identify that specific context (i.e. when there may be ambiguity) about the meaning of the value in Priority.

Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader
CopyDuplicate	
PossibleDuplicate	
Priority	**the priority as defined within the BusinessService**
Signature	
Related	

2.13 Business Application A sends a signed Business Message to BusinessApplication B

The signature must be structured as per the W3C XML Signatures specification and any additional constraints stated for the service / business area within which this BusinessMessage is exchanged.

Following MessageElements must be used:

MessageElement Name	Usage
CharacterSet	
From	Id of BusinessApplication A
To	Id of BusinessApplication B
BusinessMessageIdentifier	Identification of the BusinessMessage
MessageDefinitionIdentifier	Identification of the MessageDefinition
BusinessService	
CreationDate	Date (and time) of the creation of this BusinessApplicationHeader
CopyDuplicate	
PossibleDuplicate	
Priority	the priority as defined within the BusinessService
Signature	**signature specification containing the signature of the Sending Business Entity**
Related	

3 Mapping of the BAH to other headers

3.1 Introduction

Below list contains all MessageElements of the BusinessApplicationHeader.
For readability purposes, below mapping uses the MessageElement names of the BAH and not their equivalent XML name.

MessageElement Name	XML Name
CharacterSet	CharSet
From	Fr
To	To
BusinessMessageIdentifier	BizMsgIdr
MessageDefinitionIdentifier	MsgDefIdr
BusinessService	BizSvc
CreationDate	CreDt
CopyDuplicate	CpyDplct
PossibleDuplicate	PssblDplct
Priority	Prty
Signature	Sgntr
Related	Rltd

3.2 BAH to SWIFTNet Application Header

Business Application Header MessageElement	AppHdr V1
Sending Business Entity the Business Entity that **created** the BusinessMessage	<From>
Receiving Business Entity the Business Entity that will **process** the BusinessMessage	<To>
character set the additionally used character set(s) in the BusinessMessage for Text datatypes	–
BusinessService Example: E&I	<SvcName>
MessageDefinitionIdentifier the identification of MessageDefinition	<MsgName>
BusinessMessageIdentifier the identification of the BusinessMessage, unique to the SendingBusinessEntity	<MsgRef>
Copy / Duplicate / Functionality to indicate this message is a copy / duplicate of another BusinessMessage	–

Related Reference Reference of the original BusinessMessage	–
Possible Duplicate The BusinessMessage may have been sent before.	<Dup>
Priority Relative indication of the processing precedence of the message over a (set of) BusinessMessages with assigned priorities	–
Signature Contains the digital signature of the person authorised to sign this BusinessMessage (based on W3C's XML Signature standard)	–
CreationDateTime Creation date (and time) of the Business Message	<CrDate>

3.3 BAH to ebXML/ebMS Header

Business Application Header element	ebMS 3.0
Sending Business Entity the Business Entity that **created** the BusinessMessage	<eb:From>
Receiving Business Entity the Business Entity that will **process** the BusinessMessage	<eb:To>
character set the additionally used character set(s) in the BusinessMessage for Text datatypes	–
BusinessService Example: E&I	<eb:Service>
MessageDefinitionIdentifier the identification of MessageDefinition	<eb:Property>
BusinessMessageIdentifier the identification of the BusinessMessage, unique to the SendingBusinessEntity	<eb:MessageId>
Copy / Duplicate / Functionality to indicate this message is a copy / duplicate of another BusinessMessage	–
Related Reference Reference of the original BusinessMessage	<eb:RefToMessageId>
Possible Duplicate The BusinessMessage may have been sent before.	–
Priority Relative indication of the processing precedence of the message over a (set of) BusinessMessages with assigned priorities	–
Signature Contains the digital signature of the person authorised to sign this BusinessMessage (based on W3C's XML Signature standard)	–
CreationDateTime Creation date (and time) of the BusinessMessage	<eb:Timestamp>

3.4 BAH to FpML

Business Application Header element	**Message Header**
Sending Business Entity the Business Entity that **created** the BusinessMessage	<sentBy>
Receiving Business Entity the Business Entity that will **process** the BusinessMessage	<sendTo>
character set the additionally used character set(s) in the BusinessMessage for Text datatypes	<?xml version="1.0" encoding="UTF-8">
BusinessService Example: E&I	-
MessageDefinitionIdentifier the identification of MessageDefinition	<xsi:type from Document>
BusinessMessageIdentifier the identification of the BusinessMessage, unique to the SendingBusinessEntity	<messageId>
Copy / Duplicate / Functionality to indicate this message is a copy / duplicate of another BusinessMessage	<copyTo>
Related Reference Reference of the original BusinessMessage	<inReplyTo>
Possible Duplicate The BusinessMessage may have been sent before.	
Priority Relative indication of the processing precedence of the message over a (set of) BusinessMessages with assigned priorities	–
Signature Contains the digital signature of the person authorised to sign this BusinessMessage (based on W3C's XML Signature standard)	<dsig:Signature>
CreationDateTime Creation date (and time) of the Business Message	<creationTimestamp>

4 ANNEX A - Signature Guidelines

4.1 Preface

The optional Signature element within the Business Application Header (BAH) can only be used when the creator of the Signature and all verifiers of that Signature agree about the way the Signature is structured.
This annex documents a proposal for guidelines to allow implementors of applications to create and verify signatures.
The guidelines are pure technical guidelines which only discuss the formal aspects of the construction of the signature. All other aspects of the signature, such as who can sign and such as the message scenarios which require signing, is not part of this annex.

4.2 Introduction

4.2.1 Current Signature Guidelines

The *Business Application Header Message Usage Guidelines* specifies: "Signature contains the digital signature of the person authorised to sign this BusinessMessage (based on W3C's XML Signature standard)".

This means that:

- The Signature element conforms to the W3C Signature Syntax and Processing Recommendation
- The BusinessMessage is the ISO 20022 Business Application Header and the ISO 20022 Message
- The person authorised to sign is not further specified; this depends on the particular context in which the message is sent.

As a reminder, the terminology of BusinessMessage, ISO 20022 Business Application Header and ISO 20022 Message is shown in the following picture

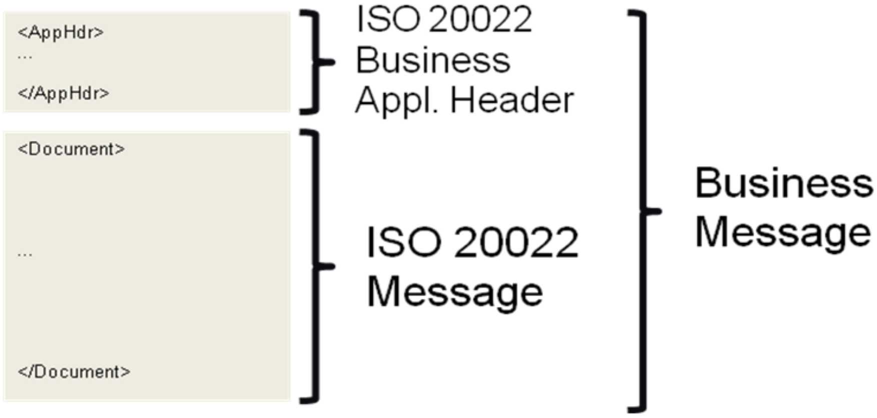


Figure 1 - Business Messagae

4.2.2 Short introduction to W3C Signatures

This short introduction provides an overview of signing and verifying based on the *W3C Signature Syntax and Processing (Second Edition) W3C Recommendation 10 June 2008*.
The following diagram shows the process followed to sign and verify an XML digital signature.

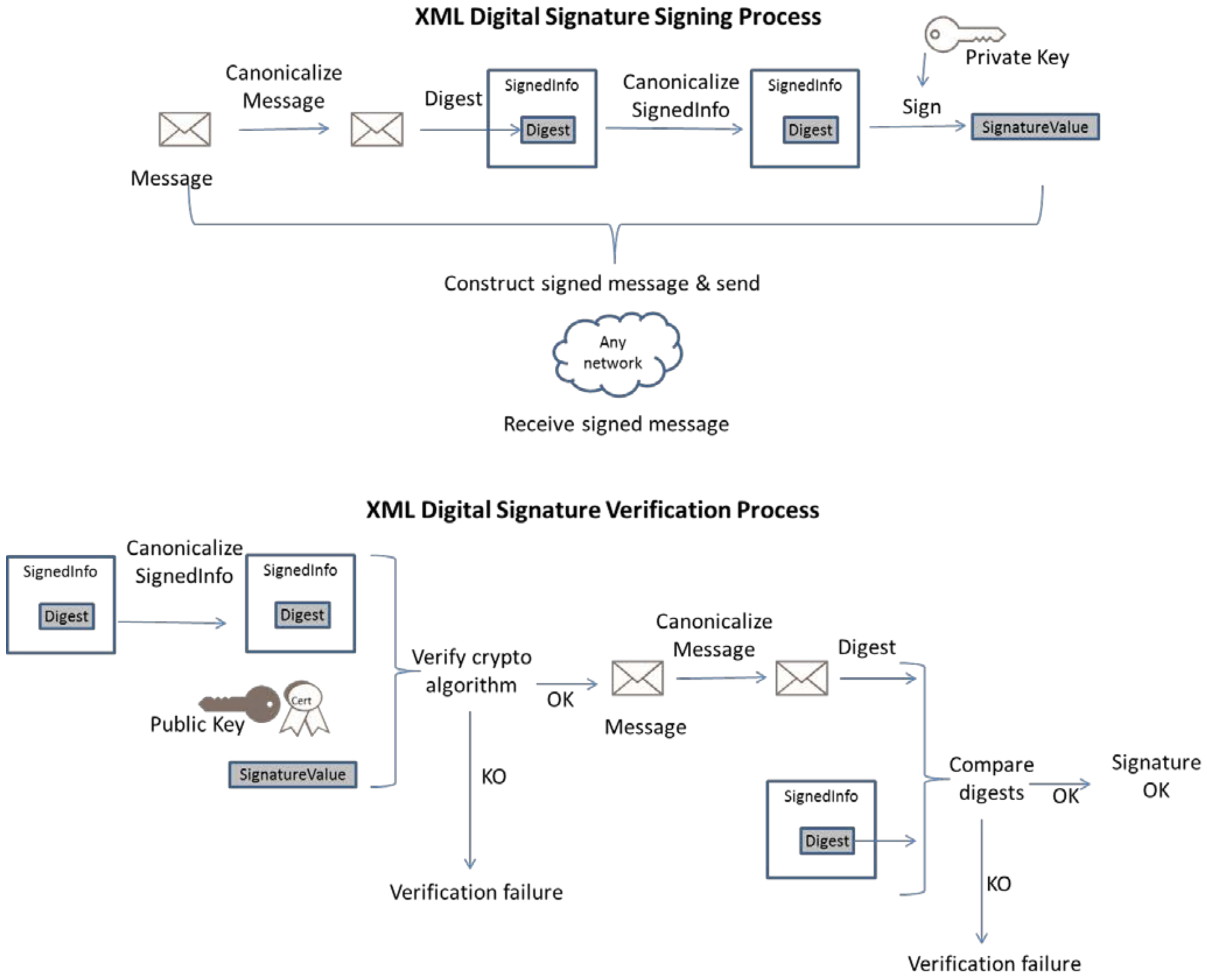
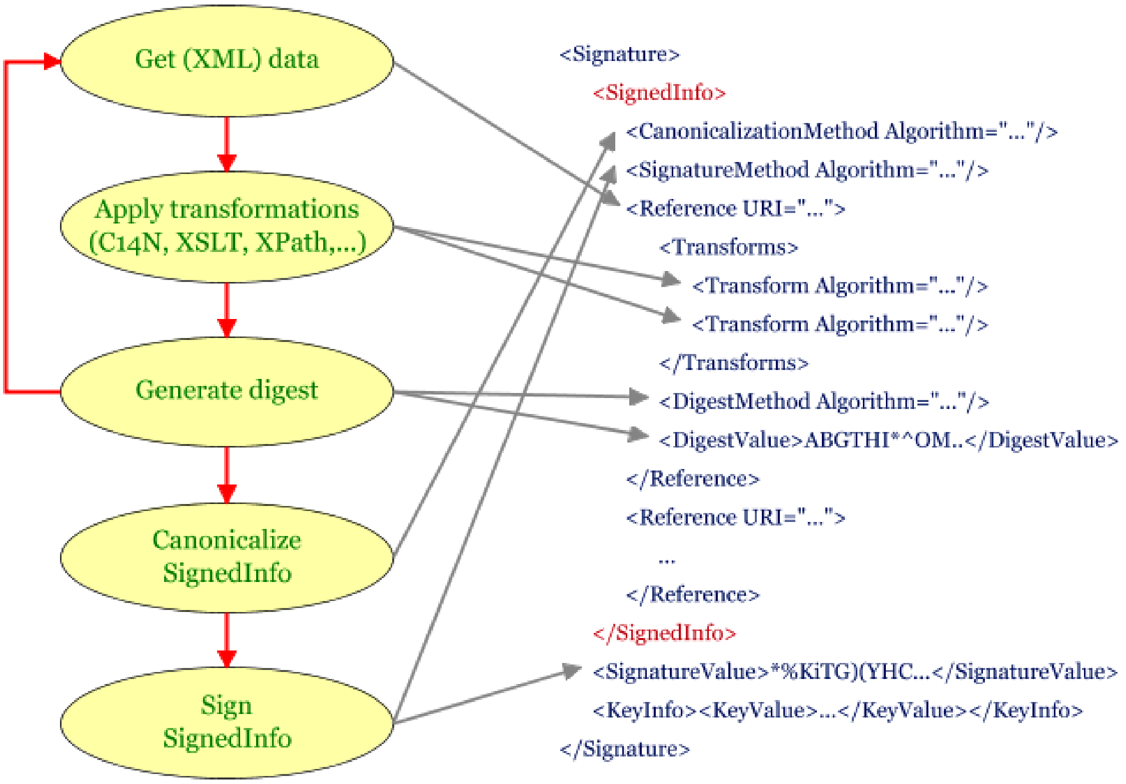


Figure 2 Signing and verifying process

The message that is to be signed is made available. The first step in the signing process is to canonicalize the message. This is a transformation that constructs the canonical form of the XML that is being signed. By doing so, the input of the digest algorithm is robust enough to be independent on the parsing technology used.
The calculated digest is placed into the SignedInfo. The exact mapping is made clear in the picture below. The SignedInfo element itself is canonicalized, and as last step the SignedInfo itself is signed using the signing algorithm. The diagram shows a private key used for this purpose.
The message and its Signature is then sent to the receiver. The verification of the Signature follows the reverse process. First the SignatureValue is verified to be correct. The input to the verification algorithm is the canonicalized SignedInfo, the certificate containing the certified public key and the SignatureValue containing the result of the signing operation. In case the verification succeeds, the SignedInfo is authentic and signed by the signer. The verification process continues and checks if the digest within the SignedInfo matches the Message that is covered by the digest. This is done by

canonicalizing the Message and then digest the canonicalized Message.The verification is ok if the calculated digest is equal to the digest within the SignedInfo.
The Signature elements used are illustrated in the following picture:



Get (XML) data
Apply transformations (C14N, XSLT, XPath,...)
Generate digest
Canonicalize SignedInfo
Sign SignedInfo
<Signature>
<SignedInfo>
<CanonicalizationMethod Algorithm="..."/>
<SignatureMethod Algorithm="..."/>
<Reference URI="...">
<Transforms>
<Transform Algorithm="..."/>
<Transform Algorithm="..."/>
</Transforms>
<DigestMethod Algorithm="..."/>
<DigestValue>ABGTHI*^OM..</DigestValue>
</Reference>
<Reference URI="...">
...
</Reference>
</SignedInfo>
<SignatureValue>*%KiTG)(YHC...</SignatureValue>
<KeyInfo><KeyValue>...</KeyValue></KeyInfo>
</Signature>

Figure 3 - XML Signature elements

As shown, the SignedInfo contains one or more Reference elements. Each Reference identifies an XML resource that is to be signed. The example in the picture shows a KeyInfo element with as content a KeyValue. Using a Public Key Infrastructure

4.2.3 Scope of this documents Signature Guidelines

This document does not introduce any change in business requirements related to the signature within the Business Application Header.

This document proposes the approach to implement the current signature guidelines. In particular it addresses the following

- The structure of the Signature
  - Algorithms used
  - KeyInfo
  - Structure of Reference elements
- The process used for signing and verifying

4.3 Structure of the Signature

Overview

This section describes the structure of the Signature signing the Business Message. The elements used within the Signature of the BAH are discussed. An example of a Signature used in the BAH can be found in *Example* on page 34.
The next section describes the processing of signing and verifying the Signature of a Business Message.

Algorithms used

Security algorithms are selected to be widely available but yet strong enough to offer sufficient security. It is possible that algorithms need to be updated to a stronger version, for example SHA3 for the digest algorithm. Applications should take this into account.

The current selected algorithms are the following

- The signing algorithm is RSA based on the SHA-256 digest on the information signed
- The digest algorithm is SHA-256
- The canonicalization algorithm is the Exclusive XML Canonicalization. This algorithm is applied prior to digesting the XML. Using the canonicalization allows applications to use XML processing and parsing on the XML data without breaking the signature. The Exclusive Canonicalization algorithm is chosen to avoid any impact of enveloping elements when sending or receiving Business Messages.

This is shown in the following table:

Purpose	Element	Algorithm attribute value
PKI sign algorithm	SignMethod	http://www.w3.org/2001/04/xmldsig-more#rsa-sha256
Digest algorithm	DigestMethod	http://www.w3.org/2001/04/xmlenc#sha256
Canonicalization algorithm	CanonicalizationMethod	http://www.w3.org/2001/10/xml-exc-c14n#

KeyInfo

The `KeyInfo` must contain the X509v3 certificate containing the validation key. Optionally, it can also contain the certificates that are part of the certification chain. This depends on the allowed PKI infrastructures used within the business context. For instance, when between sender and receiver always the same PKI infrastructure is used, then it can be better to use the CA root certificate from the environment than to check that the correct CA root certificate is used within the `KeyInfo`.
No other elements are required in the `KeyInfo`.

Note It is recommended not to use any other elements within the `KeyInfo`.

The `KeyInfo` itself is signed and must contain the `Id` attribute. This attribute should contain a unique value.
The KeyInfo looks like:

```
<ds:KeyInfo Id="Unique-id-to-KeyInfo">
  <ds:X509Data>
    <ds:X509Certificate>MIID1DCCArygAwIB...   </ds:X509Certificate>
  </ds:X509Data>
</ds:KeyInfo>
```

The `X509Certificate` contains the certificate that is used to verify the `Signature`. In case the full certification chain is to be present at least the CA root certificate is added as shown in the following example. The order of the different certificates is not relevant.

```
<ds:KeyInfo Id="Unique-id-to-KeyInfo">
  <ds:X509Data>
    <ds:X509Certificate>MIID1DCCArygAwIB...   </ds:X509Certificate>
    <ds:X509Certificate>MIIDkDCCAnigUoSI...  ==</ds:X509Certificate>
  </ds:X509Data>
</ds:KeyInfo>
```

Note	Inclusion of the entire certification chain is recommended. Only when signatures must be performed using certified keys by a specific PKI infrastructure, only the signing certificate is sufficient.

SignedInfo

The `SignedInfo` contains the 3 `Reference` elements.

- `Reference` digesting the ISO 20022 Message
- `Reference` digesting the BAH
- `Reference` digesting the `KeyInfo`

The order of the `Reference` elements in the `SignedInfo` is not relevant.

Reference of ISO 20022 Message

The `Reference` is identified by the fact that the `URI` attribute is absent. The absence of the `URI` attribute indicates that the logic for de-referencing this reference is part of the application logic. The processing of such `Reference` is further described in

Process for signing and verifying in page 31.

Note Allthough the Business Message, as illustrated in Figure 1, consists of the BAH and the ISO 20022 Message, it is not possible to refer easily from the BAH to the ISO 20022 Message. Therefore, the option to use the absent `URI` attribute was selected, since this is possible on one `Reference` element within the `SignedInfo`.

The `Reference` looks like

```
<ds:Reference>
  <ds:Transforms>
    <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
  </ds:Transforms>
  <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
  <ds:DigestValue>
    QYYVi9JdlsOxjplrW3vIjT8cWYyzYD4ZnnNJ9SH+dvQ=
  </ds:DigestValue>
</ds:Reference>
```

Reference of BAH

The `Reference` of the BAH is identified by the empty `URI` attribute `URI=""`.
Such empty URI attribute URI="" refers to the XML document in which the Signature is placed. This means that the signing and verifying of the Signature within the BAH happens within the context of the BAH.
The complete BAH is signed except the `Signature` itself. This is indicated in the first `Transform` algorithm within the `Transforms`. The second algorithm contains the Exclusive XML canonicalization.

```
<Reference URI="">
  <Transforms>
    <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
    <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
  </Transforms>
  <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
  <DigestValue>
    +OS/MM1NBTiOZWWpvzOWkfRjyP2/F1lg9P+zvC+Gulk=
  </DigestValue>
</Reference>
```

Reference of KeyInfo

The reference to the `KeyInfo` requires that the `KeyInfo` can be identified with an `Id` attribute. To ensure that such an attribute can be handled correctly when the `Signature` is enveloped within other XML documents, that attribute should contain a unique value within the XML document enclosing the Signature.

Note The `Signature` is XAdES-BES compliant by adding the `Reference` within the `SignedInfo`.

```
<ds:Reference URI="#Unique-id-to-KeyInfo">
  <ds:Transforms>
    <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
  </ds:Transforms>
  <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
  <ds:DigestValue>
    QYYVi9JdlsOxjplrW3vIjT8cWYyzYD4ZnnNJ9SH+dvQ=
  </ds:DigestValue>
</ds:Reference>
```

The optional attribute `Type="http://www.w3.org/2000/09/xmldsig#KeyInfo"` can be present on the `Reference` element referring to the `KeyInfo`. There is no obligation to further process this attribute.

```
<ds:Reference URI="#Unique-id-to-KeyInfo"
Type="http://www.w3.org/2000/09/xmldsig#KeyInfo">
  <ds:Transforms>
    <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
  </ds:Transforms>
  <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
  <ds:DigestValue>
    QYYVi9JdlsOxjplrW3vIjT8cWYyzYD4ZnnNJ9SH+dvQ=
  </ds:DigestValue>
</ds:Reference>
```

4.4 Process for signing and verifying

Signing is performed as last step in generating the Business Message

When an application prepares a Business Message several steps may be needed. Indeed, some applications require an approval process where the message may be amended.
In some cases, some elements of the BAH may be created after the ISO 20022 Message has been created.
In any case, the last step in the creation of the Business Message is calculating the SignatureValue within the Signature. The creation of the signature can be implemented within the flow of sending the Business Message as long as the process as described in this section is applied.

Two node-sets

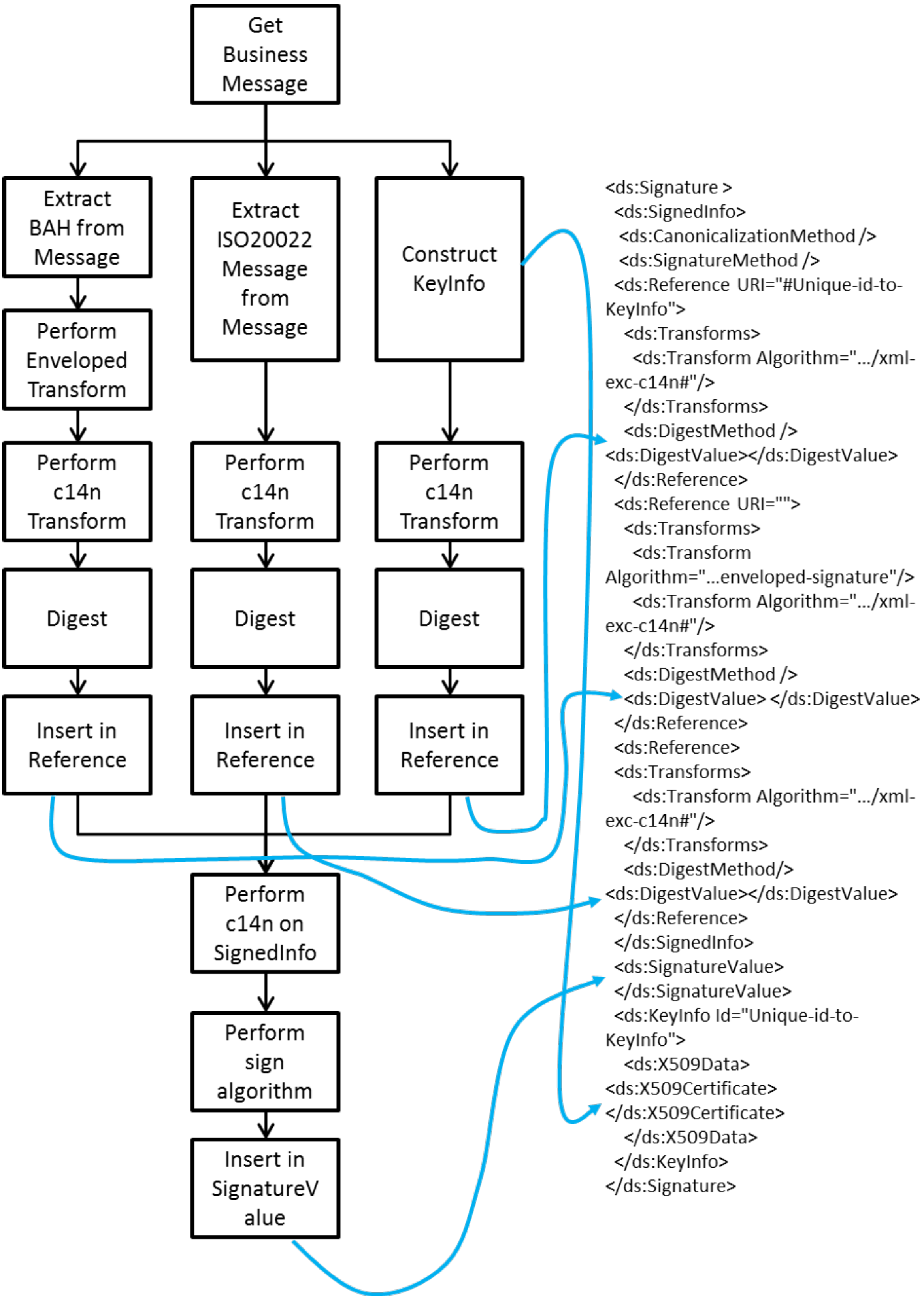
As illustrated in Figure 1, the Business Message consists of two XML documents. The way in which those two documents are created, sent, received and processed depends on the infrastructure used by the sender and receiver. It is therefore important that the process in creating the Signature is independent of the infrastructure used.

This is achieved by performing the Signature creation and verification using a well defined XML context. Two such context are used:

- The context of the BAH element (AppHdr)
- The context of the ISO 20022 Message element (for instance Document)

Signing a BAH

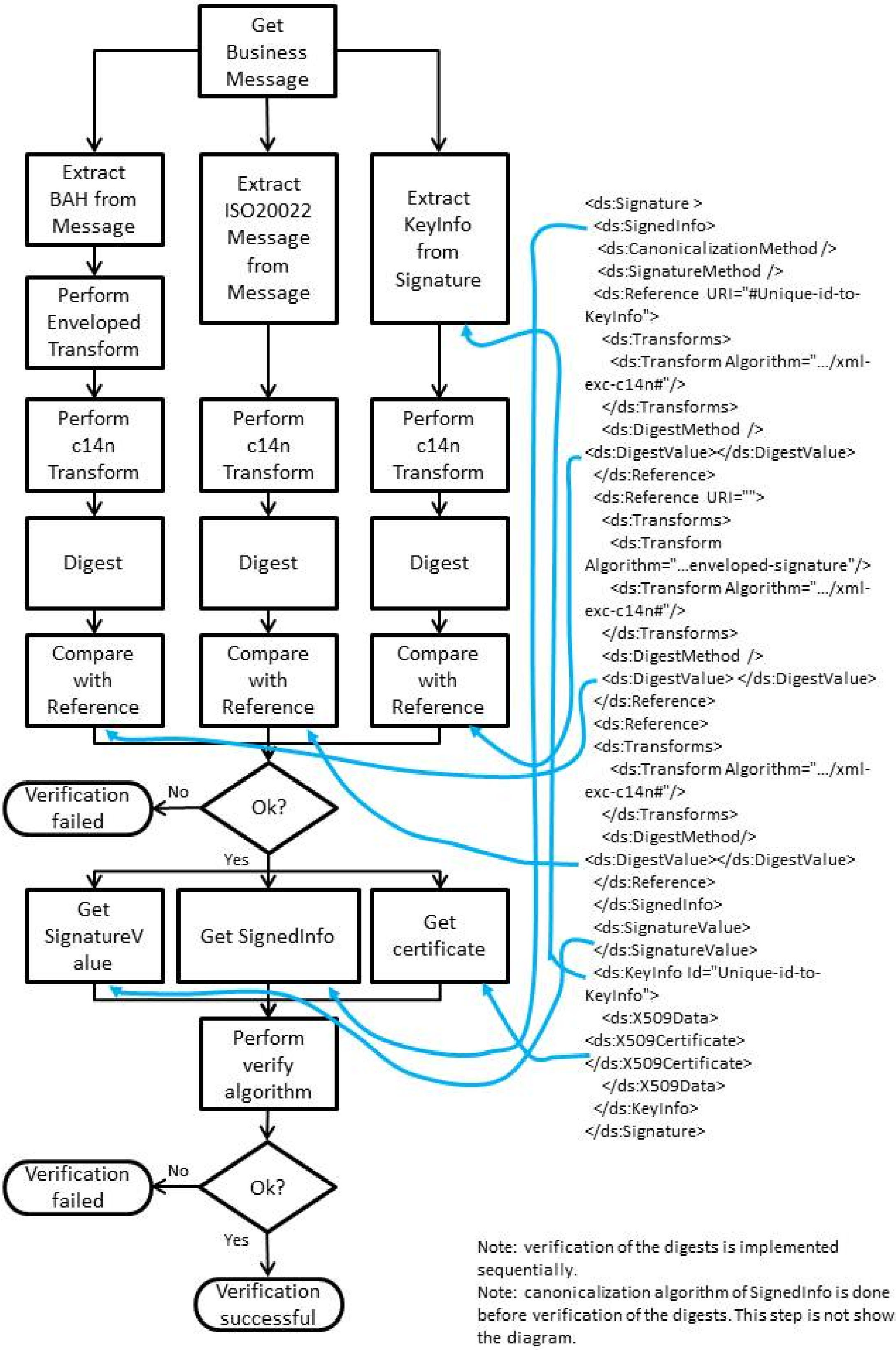
The process of signing is illustrated in the following flowchart. The start is a Business Message composed of a BAH and an ISO 20022 Message. The Signature is created, similar to the example in next section. The DigestValue and SignatureValue have to be created. This process is described in the flowchart.



```
<ds:Signature >
 <ds:SignedInfo>
  <ds:CanonicalizationMethod />
  <ds:SignatureMethod />
 <ds:Reference URI="#Unique-id-to-
KeyInfo">
   <ds:Transforms>
     <ds:Transform Algorithm=".../xml-
exc-c14n#"/>
   </ds:Transforms>
   <ds:DigestMethod />
<ds:DigestValue></ds:DigestValue>
 </ds:Reference>
 <ds:Reference URI="">
   <ds:Transforms>
     <ds:Transform
Algorithm="...enveloped-signature"/>
     <ds:Transform Algorithm=".../xml-
exc-c14n#"/>
   </ds:Transforms>
   <ds:DigestMethod />
   <ds:DigestValue> </ds:DigestValue>
 </ds:Reference>
 <ds:Reference>
 <ds:Transforms>
     <ds:Transform Algorithm=".../xml-
exc-c14n#"/>
   </ds:Transforms>
   <ds:DigestMethod/>
<ds:DigestValue></ds:DigestValue>
 </ds:Reference>
 </ds:SignedInfo>
 <ds:SignatureValue>
 </ds:SignatureValue>
 <ds:KeyInfo Id="Unique-id-to-
KeyInfo">
   <ds:X509Data>
<ds:X509Certificate>
</ds:X509Certificate>
   </ds:X509Data>
 </ds:KeyInfo>
</ds:Signature>
```

Figure 4 - Flowchart signing a BAH

Verifying a BAH Signature



```
<ds:Signature >
 <ds:SignedInfo>
  <ds:CanonicalizationMethod />
  <ds:SignatureMethod />
 <ds:Reference URI="#Unique-id-to-
KeyInfo">
   <ds:Transforms>
    <ds:Transform Algorithm=".../xml-
exc-c14n#"/>
   </ds:Transforms>
   <ds:DigestMethod />
<ds:DigestValue></ds:DigestValue>
 </ds:Reference>
 <ds:Reference URI="">
   <ds:Transforms>
    <ds:Transform
Algorithm="...enveloped-signature"/>
    <ds:Transform Algorithm=".../xml-
exc-c14n#"/>
   </ds:Transforms>
   <ds:DigestMethod />
   <ds:DigestValue> </ds:DigestValue>
 </ds:Reference>
 <ds:Reference>
 <ds:Transforms>
    <ds:Transform Algorithm=".../xml-
exc-c14n#"/>
   </ds:Transforms>
   <ds:DigestMethod/>
<ds:DigestValue></ds:DigestValue>
 </ds:Reference>
 </ds:SignedInfo>
 <ds:SignatureValue>
 </ds:SignatureValue>
 <ds:KeyInfo Id="Unique-id-to-
KeyInfo">
   <ds:X509Data>
<ds:X509Certificate>
</ds:X509Certificate>
   </ds:X509Data>
 </ds:KeyInfo>
</ds:Signature>
```

Note: verification of the digests is implemented sequentially.
Note: canonicalization algorithm of SignedInfo is done before verification of the digests. This step is not shown in the diagram.

Figure 5 - Verifying a BAH signature

4.5 Example

An example of a `Signature` within the BAH that can be verified is as follows:

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
   <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
   <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-
sha256"/>
  <ds:Reference URI="#Unique-id-to-KeyInfo">
    <ds:Transforms>
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    </ds:Transforms>
    <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
    <ds:DigestValue>
      QYYVi9JdlsOxjplrW3vIjT8cWYyzYD4ZnnNJ9SH+dvQ=
    </ds:DigestValue>
  </ds:Reference>
  <ds:Reference URI="">
    <ds:Transforms>
      <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/>
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    </ds:Transforms>
    <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
    <ds:DigestValue>
      +OS/MM1NBTiOZWWpvzOWkfRjyP2/F1lg9P+zvC+Gulk=
    </ds:DigestValue>
  </ds:Reference>
  <ds:Reference>
  <ds:Transforms>
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    </ds:Transforms>
    <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
    <ds:DigestValue>
      QYYVi9JdlsOxjplrW3vIjT8cWYyzYD4ZnnNJ9SH+dvQ=
    </ds:DigestValue>
  </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>
  IKDs7kwX14CxK3AZ1ojJLr35A4eU/98tp/10KFQTtPOwR5WCKyx
  4I05ZV1IljOpEgpkt6xejXhshaEnNBD5B5PII1VN6mviJJjU/njGikNeXzi1Djei2dPEap
  nPX1f26UnQcgYTAqaSVwAnIR7L8/W2UeT8J9z8Rd1OebYV5xE8jVehbgMcAmJwv2rC/c2d
  UkUe2/eBU0APyWGCgKawxGGAPLP3AS4+Mp0ODKlVp08rUzVOF+pFF/1dBknlK/v0dWkDdj
  YvwFRvZhHXue/PYvMNtQBytMUUdB1MiQrNX0jSCE6Y2nljhTXdcrb2lgfgwCclB6xArBRV
  WfMa0kQVQ4Q==
  </ds:SignatureValue>
  <ds:KeyInfo Id="Unique-id-to-KeyInfo">
    <ds:X509Data>
      <ds:X509Certificate>MIID1DCCArygAwIB   </ds:X509Certificate>
    </ds:X509Data>
  </ds:KeyInfo>
</ds:Signature>
```

Revision Record

Revision	Date	Author	Description	Sections affected
1.0	15/04/2010	ISO 20022 RMG/TSG	Initial version	All
1.1	30/06/2010	TSG secretariat	corrected typos in some examples using T2S and added this revision record	2.1.1.1 and 2.1.6.1
1.2	20/10/2010	TSG secretariat	corrected typos, clarified scenario 2, updated the TOC. Replaced MessageInstanceIdentifier by BusinessMessageIdentifier in chapter 3 Added new scenario 2.10	2.1.2, 3.x, 2.10, TOC
1.4	20/4/2011	TSG secretariat	corrected typos, added 1.7, corrected scenario 2.4	1.7, 2.4
1.5		TSG secretariat	Corrected typo; added an XML Schema that formally describes an ISO 200222 Business Message.	2.3 Diagram 1.7
1.6	8/5/2013	TSG secretariat	Added ANNEX A - Signature Guidelines	4
1.7	23/5/2013	TSG secretariat	Added last paragraph about use of ISO 20022 XML syntax	1.1
1.8	8/4/2016	TSG	Added new section on BAH deployment options	2.2 and 2.3

Disclaimer:
Although the Registration Authority has used all reasonable efforts to ensure accuracy of the contents of the iso20022.org website and the information published thereon, the Registration Authority assumes no liability whatsoever for any inadvertent errors or omissions that may appear thereon. Moreover, the information is provided on an "as is" basis. The Registration Authority disclaims all warranties and conditions, either express or implied, including but not limited to implied warranties of merchantability, title, non-infringement and fitness for a particular purpose. The Registration Authority shall not be liable for any direct, indirect, special or consequential damages arising out of the use of the information published on the iso20022.org website, even if the Registration Authority has been advised of the possibility of such damages.