Direct message exchange with Finnish Customs

Technical guidebook

Updated 9 November 2023

Contents

Tables:

Figures:

1 Purpose of this guidebook

This guidebook contains a description of the technical details of Finnish Customs' direct message exchange web service, which are required for connecting the customer's system to the service of Customs, instructions on acquiring a server certificate and a description of the direct message exchange web service. The annexes also provide a lot of useful information.

The recent changes/amendments are highlighted in yellow.

The following useful publications are also available at the Finnish Customs website

1. Introduction to message exchange with Finnish Customs
2. Terms of use for direct message exchange
3. Most common problem situations in direct message exchange

Introduction to message exchange with Finnish Customs *is helpful when choosing the service channel.* It also describes the functions and roles in direct message exchange in more detail. Like all message exchange, direct message exchange also requires an authorisation. You will find more information about this in Introduction to message exchange with Finnish Customs.

http://tulli.fi/en/e-services/services/message-exchange

Terms of use for direct message exchange *can be found in the annexes to the application.* You should read them if you are going to start using direct message exchange.

http://tulli.fi/en/e-services/forms/message-exchange-forms

Most common problem situations in direct message exchange (FAQ checklist) *contains descriptions of problems that customers have had when using direct message exchange as well as solutions to these problems.* The list is kept up to date, so it is recommended to take a look at it from time to time.

http://tulli.fi/en/e-services/services/direct-message-exchange

Customs has also provided some **example implementations** that customers can use when creating their own applications.

There are example implementations for the following services:

direct message exchange (Java)

receipt of message notifications (Java)

The example implementations are free of charge and they can be requested from the EDI support of the Electronic Service Centre. Customs does not provide any support for their use.

On the Customs website, there is a wide selection of **example messages**. They can be found in connection with the direct message exchange version notices.

http://tulli.fi/en/e-services/services/direct-message-exchange

2 Web service for direct message exchange

In direct message exchange, businesses can send messages to and download messages from Customs systems either via an EDI operator or directly over the Internet. The latter is called direct message exchange, for which Customs provides a web service to its business customers.

The idea of direct message exchange is that the customer's system sends messages to Customs and downloads waiting messages from Customs. Customs never sends its reply messages to the customer. The customer will find out about the identifying data of any messages waiting to be downloaded by making a DownloadList request. The customer can then download the messages

based on the list. The customer can also start using the Message Notification Service, which will notify the customer's system of messages waiting to be downloaded.

Customs will provide direct message exchange interface with the following systems: AREX, ELEX, EMCS, ALA, ITU, Transit and Intrastat.

In their own environment, the customers must implement services that enable the use of services provided by Customs.

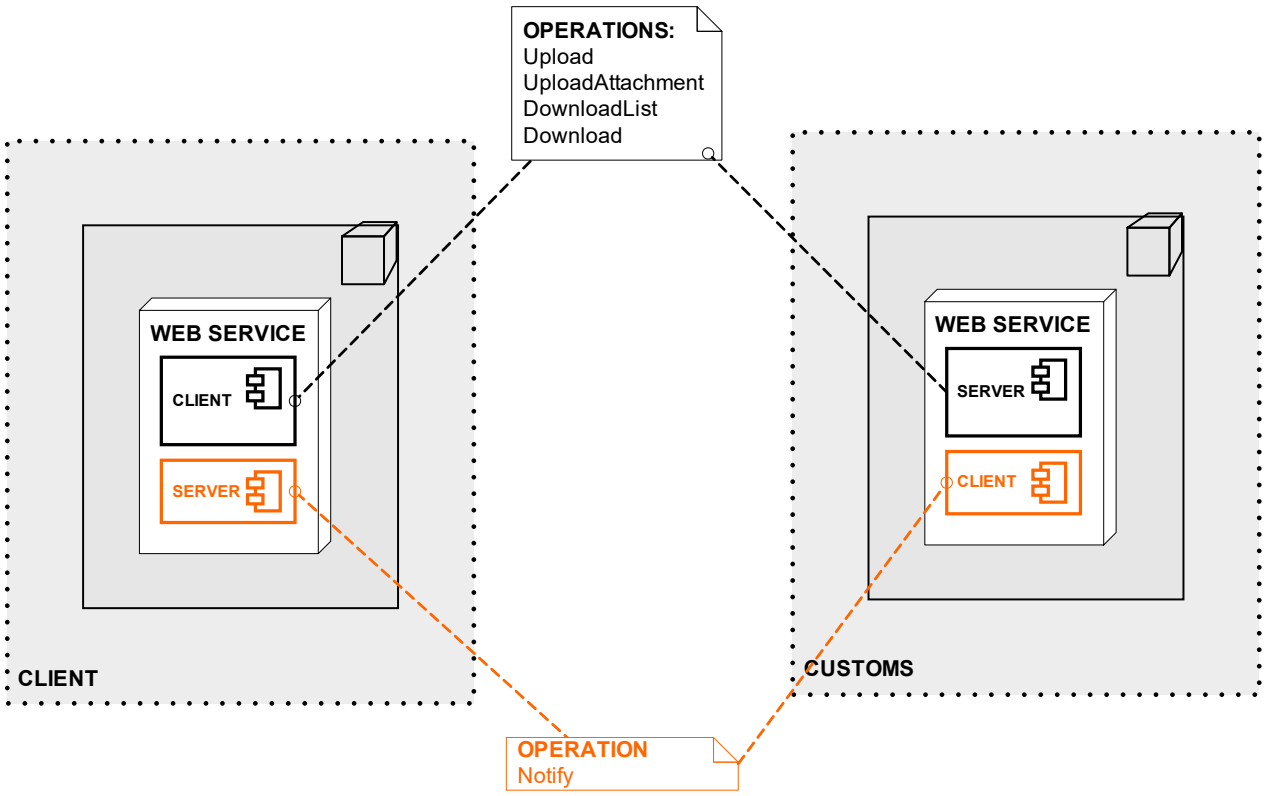

Figure 1: **Example of the web services**

2.1 Logical architecture of direct message exchange

Via the direct message exchange interface, the customer can use one or several of Customs' data systems. From the customer point of view, the simplest and most straightforward way is to build one integration system to generate and send all the messages to Customs' systems. In such a case, only one server certificate is needed and the implementation of new customer applications into the direct message exchange service channel is easier for the customer.

It is also easier to implement the Message Notification Service into a solution where the customer only has one channel that receives all notifications from the Message Notification Service.

As for challenges caused by other solutions, please see: Most common problem situations:

http://tulli.fi/en/e-services/services/direct-message-exchange

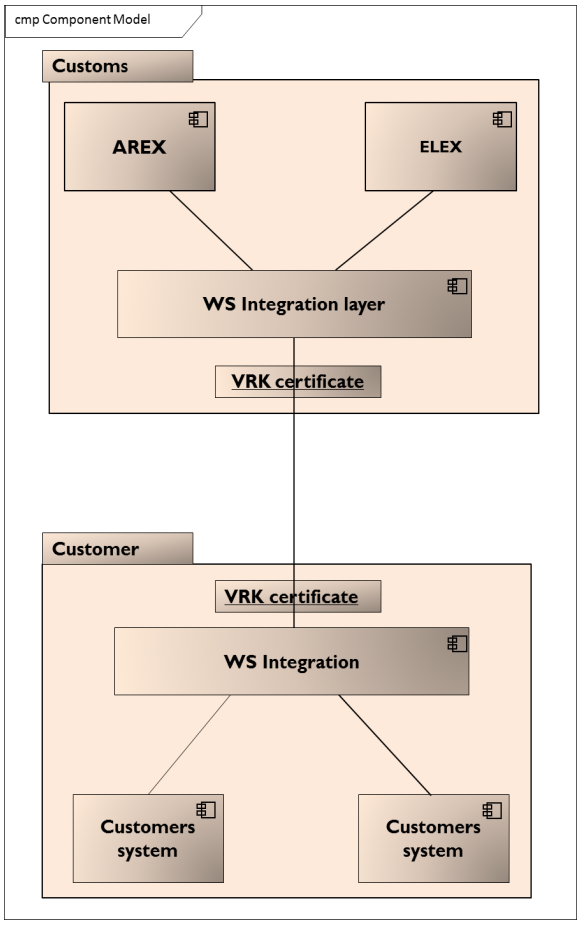

Figure 2: Example of a Customs and customer solution

2.2 Data security of the service

The confidentiality of the message transfer in direct message exchange is based on the use of a sufficiently strong SSL/TLS encryption in all message transfer transactions directed via a public network.

In direct message exchange, the authentication of the customer is based on two-way SSL/TLS authentication. By asking the customer for the public key certificate signed by the Digital and Population Data Service Agency (DVV) in connection with the SSL/TLS handshake, Customs' direct message exchange interface authenticates the identity of the customer and checks the authorisations issued to the customer for using Customs' programs. Customs' interface also uses the DVV's certificate revocation list service and rejects any message transfer attempts made with certificates placed on the revocation list. The customer can also authenticate the identity of Customs' interface system by the public key certificate presented by it in the SSL/TLS handshake.

The SSL/TLS encryption used in the direct message exchange interfaces for the identification of both parties also curbs denial-of-service attacks directed at the interface by outside parties, because Customs' interface will reject any SSL/TLS connection being established if the message sender does not have a valid certificate issued by DVV.

The integrity of the customs clearance messages sent by the customer is ensured by using an XML signature (XMLDSig), by means of which Customs' interface application checks the integrity of the message and the identity of the signer.

Customs' direct message exchange interface stores both the messages sent by the customers and the messages returned to the customer at least as long as required by law. During this time, it is possible to check the contents and transmission data, if required.

Customs' direct message exchange interface checks the messages sent by the customer in case of various XML threats and SQL injection attacks.

Customs' interface also limits the traffic volume from customers by preventing any denial-of-service attacks caused by intended or unintended overload situations from succeeding.

Customs' interface stops the traffic volume from customers by preventing any denial-of-service attacks caused by intended or unintended overload situations from succeeding.

In connection with the application for message exchange, the customer approves the terms of use for Customs' direct message exchange, including the customer's obligations and responsibilities relating to data security (Customs form 934e_15).

2.3 Message Notification Service

If the customer uses **the Message Notification Service**, Customs will send the identifying data of the message waiting to be downloaded to the URL provided by the customer as soon as the message is ready to be downloaded. This enables a nearly real-time processing of messages and helps to avoid unnecessary traffic between the customer's and Customs' systems. Customs recommends the use of the Message Notification Service.

The Message Notification Service is specific for each Business ID.

To receive the message notification, the customer needs the following:

- Web service server functionality
- The customer's own internal processing logic for message notifications
- Server environment (e.g. application server + https server)
- Opening of firewall ports in the customer's network
- Certificate containing a server certificate (DVV)
- A company can only have one message notification receipt service (per Business ID) for all applications and software in direct message exchange. If needed, the customer's service will have to be able to distribute the received notifications to the right system.
- When the customer starts using the Message Notification Service, it will be applied to all the customer's applications. Of the notifications, the customer has to filter the ones it wants to process, if it does not want to use notifications in all applications.

In this respect, the customer is responsible for the implementation and maintenance.

The data transfer connections for the Message Notification Service in a public network are always created using SSL/TLS encryption. In terms of data security, it is advisable that the customer only allows the communications between the message notification server it provides and the address sending Customs' message notifications. It is also recommended that when connecting, the customer checks – by means of a public key certificate – the identity of the customer software calling its message notification service (HTTPS client authentication). Therefore, the customer software would identify Customs' service by the service certificate sent by it, so that the customer's service could verify the sender of the message notification.

Even when the Message Notification Service is in use, the customer should, from time to time, make a DownloadList request, because in case of disruptions, Customs will not try to resend a message notification. The customer must then find out the identifiers of the messages waiting to be downloaded.

In this case, a DownloadList request can be made less frequently than usual, once every hour at most.

2.4 Attachment message service

Customers can submit declaration attachments to Customs by using the Liitu attachment service. The sending of attachment files has been executed through the UploadAttachment operation, which has been added to the direct message exchange transaction interface.

After submitting the declaration and after processing the acceptance message retrieved from Customs, customers can send declaration attachments to Customs. Alternatively, Customs can, in connection with the control of the matter, request that the customer supplement the customs declaration with specific attachment documents.

The sending of attachment files is executed by requesting the UploadAttachment operation in the Customs direct message exchange interface. One attachment file is sent to Customs in one UploadAttachment message. If the wish is to send several attachment files, they are then sent in separate UploadAttachment messages.

The Customs integration layer converts the attachment file, sent by the customer, into a format required by the customs clearance applications and a rejection or acceptance message is then sent to the customer (LiituResponseMessage). The actual control of the content of the message is carried out in the appropriate customs clearance application, when the declaration is checked.

The message sending sequence is presented in the 'Introduction to message exchange with Finnish Customs' in chapter 2.1.4 'Operations and processes when using the attachment file message service'.

3 Customs' SOAP messages

3.1 Structure of SOAP messages

The direct message interface (web service) uses SOAP request and response messages for transmitting payloads. The structure of the SOAP requests and responses are described with WSDL and XML schema descriptions.

Each transaction is composed of a SOAP request message from the customer and a SOAP response message from Customs (request/response).

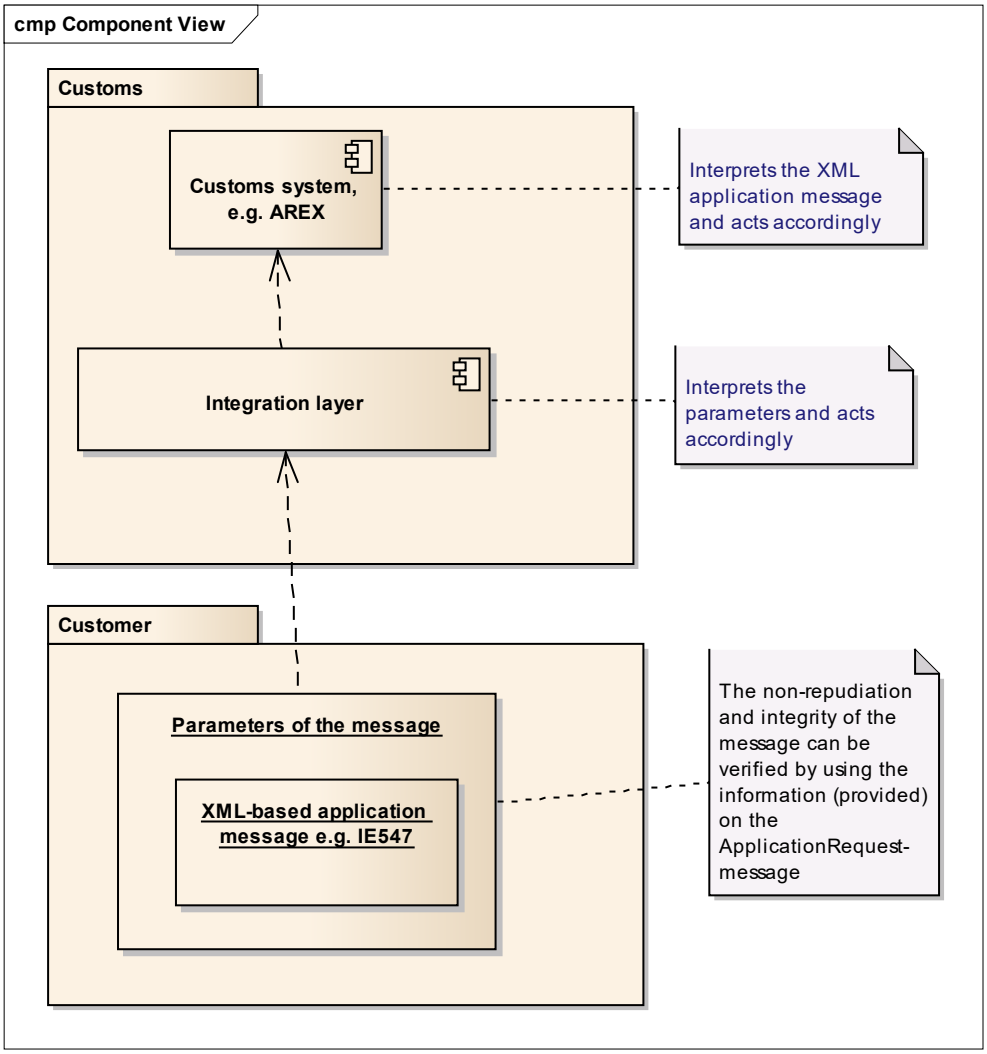

Figure 3: Message arriving from a customer

3.1.1 SOAP request message

SOAP messages have an outer envelope, *SOAP Envelope*. The envelope contains a mandatory *SOAP Body*. The SOAP Body always contains a standard XML element, *RequestHeader.* In addition, the request contains an operation-specific XML element (a description of the operations is presented in this document).

The XML structures (the parameters of the message) of the SOAP messages of Finnish Customs are standardised and independent of the application, whereas the XML payloads of Finnish Customs are uniquely defined for the application.

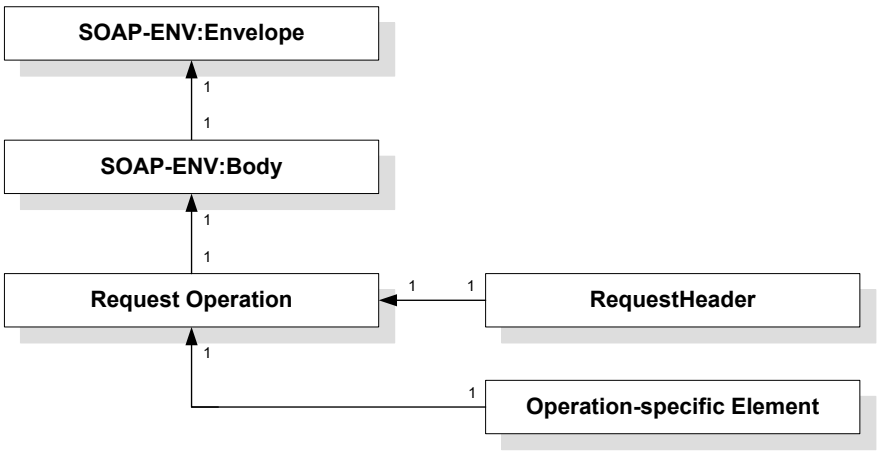

Figure 4: SOAP request message structure, high-level model

3.1.2 SOAP response message

The body of a SOAP response message always contains a standard XML element, *ResponseHeader.* In addition, the response contains an operation-specific XML element (or elements).

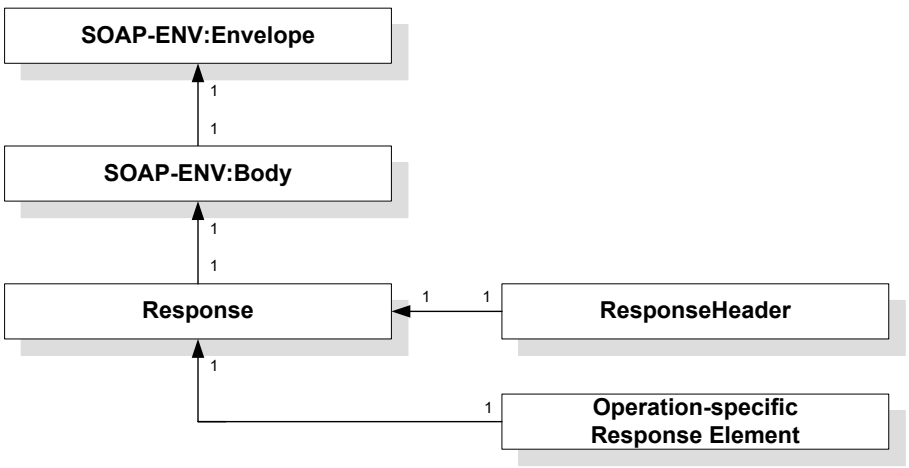

Figure 5: SOAP response message structure, high-level model

3.2 Response in normal and error situations

3.2.1 ResponseHeader response codes

In a normal situation, the message interface returns the SOAP response. The HTTP return value of the response is ’200 OK’. In this case, the value of the ResponseCode XML element in the ResponseHeader is ’000’ and the content of the Response Text XML element is ’OK’.

In an error situation, the message interface always strives to return the normal SOAP response. The HTTP return value of the response is ’200 OK’. In an error situation, the value of the ResponseCode XML element in the ResponseHeader and the content of the Response Text XML element are in accordance with Appendix 1.

3.2.2 SOAP fault

As a rule, Customs strives to return information to the customer’s system about any detected error situations in the ResponseHeader data element. However, in rare and unexpected error situations it can be impossible to process the received SOAP message, in which case a response with an adequate ResponseHeader cannot be returned. For these situations, a fault element has been defined in the WSDL description.

In error situations such as these, the message interface returns a SOAP fault, in which the HTTP return value is ’500 Internal Error’.

The following data is entered into the fault element:

- XML element code, into which 999 is always entered in cases of unexpected errors.
- XML element text, into which the text Unexpected error or Backend technical error is entered in cases of unexpected errors.

Customs reserves the right to enter content other than the above-mentioned in the fault element. There are also known error situations, where a SOAP fault containing only the generic SOAP fault elements faultcode and faultstring are returned to the customer.

3.2.3 Preparation of customer software for error situations

Customer software should be prepared for the most common error situations. The reaction into an error situation depends on the type of the error. In Appendix 1, the errors are classified into different categories depending on:

- whether the customer software should try to resend the message
- whether the error in the message should be corrected and then resent
- whether there any errors in the authorisation data which require contacting the Customer Support of Customs

Customer software also has to be prepared for processing error situations where the Customs system cannot even return a SOAP response, but only a SOAP fault.

3.2.4 Resending of a message

If a message is re-sent, the customer has to be careful about when the interchange identifier is changed, so that no double messages are accidentally sent to Customs’ systems. The following figure illustrates the situation.

A double message is a message that is identical to a message already received by the Customs system. Double messages will lead to a need for corrections and, consequently, cause more work both for the customer and for Customs.

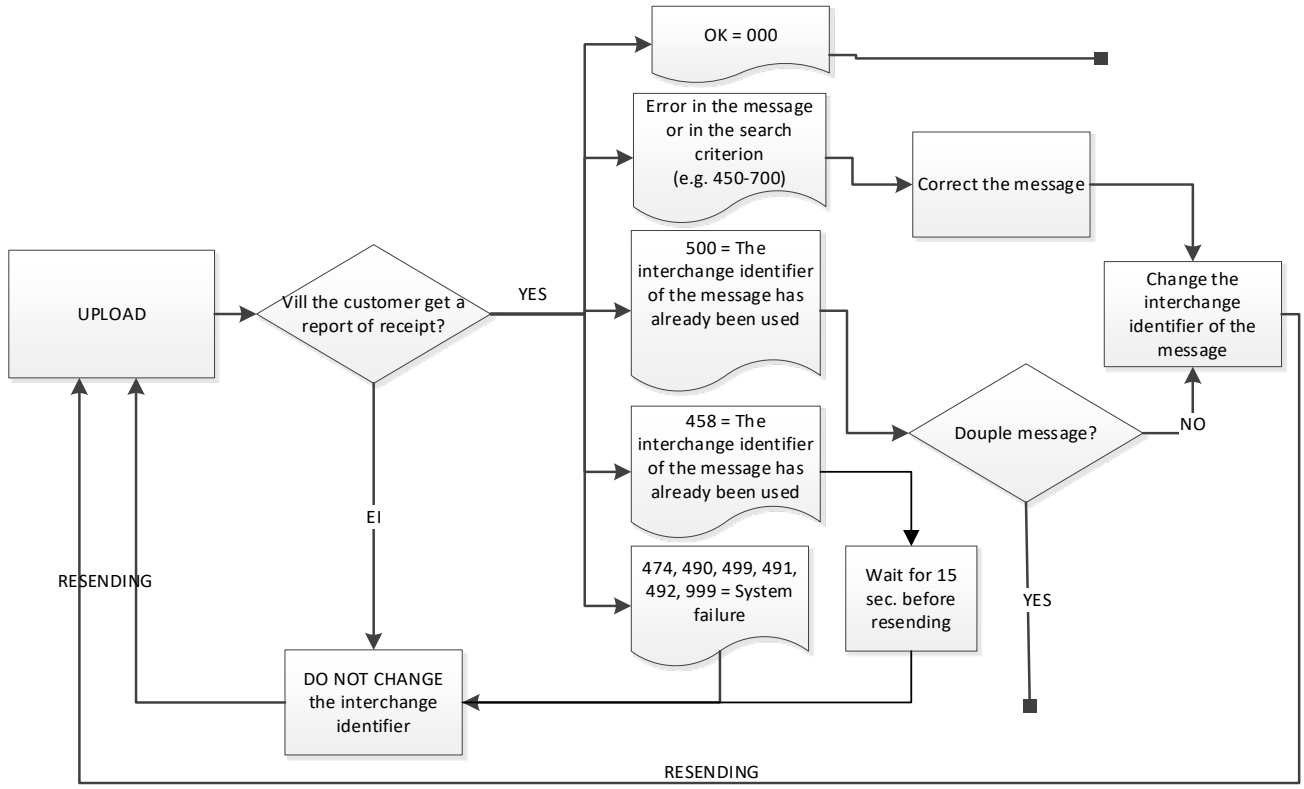

Figure 6: Resending of the message and the interchange identifier

The timeout for the https calls sent by Customs' interface should be at least 120s. This is to ensure that the customer software will not try to stop the message when Customs systems are still possibly processing it.

3.3 The phases of building and transmitting the message (UPLOAD)

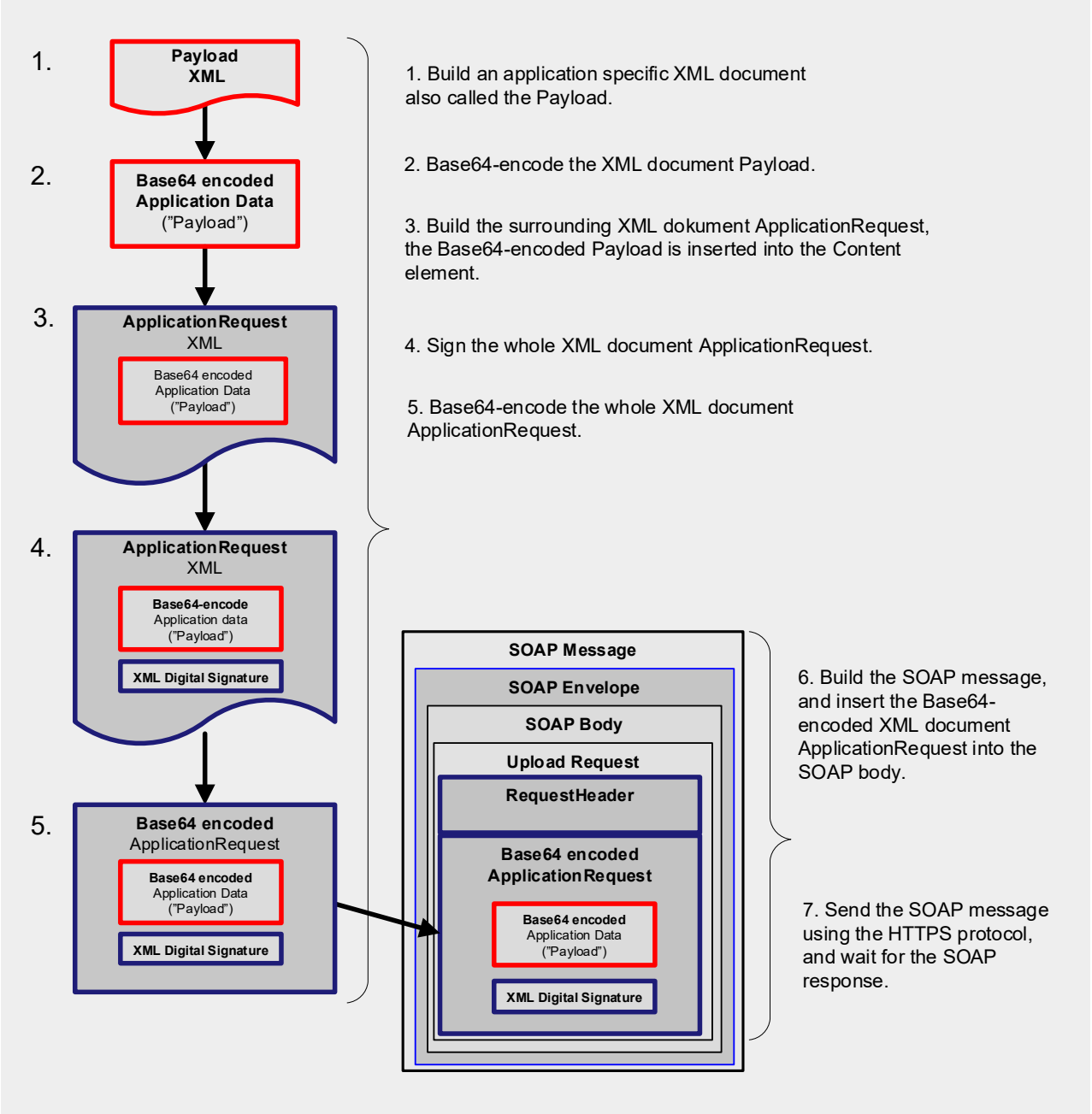

Figure 7: Building the message to Customs

An ApplicationRequest block is built around the actual payload. It contains a Content element built from the payload and the builder's signature, which covers the Content element.

The intermediary embeds this complete, signed ApplicationRequest block, Base64 encoded, as the value for the ApplicationRequestMessage element.

Both the payload and the ApplicationRequest are Base64-encoded.

3.4 The phases of building and transmitting a message (UPLOADATTACHMENT)

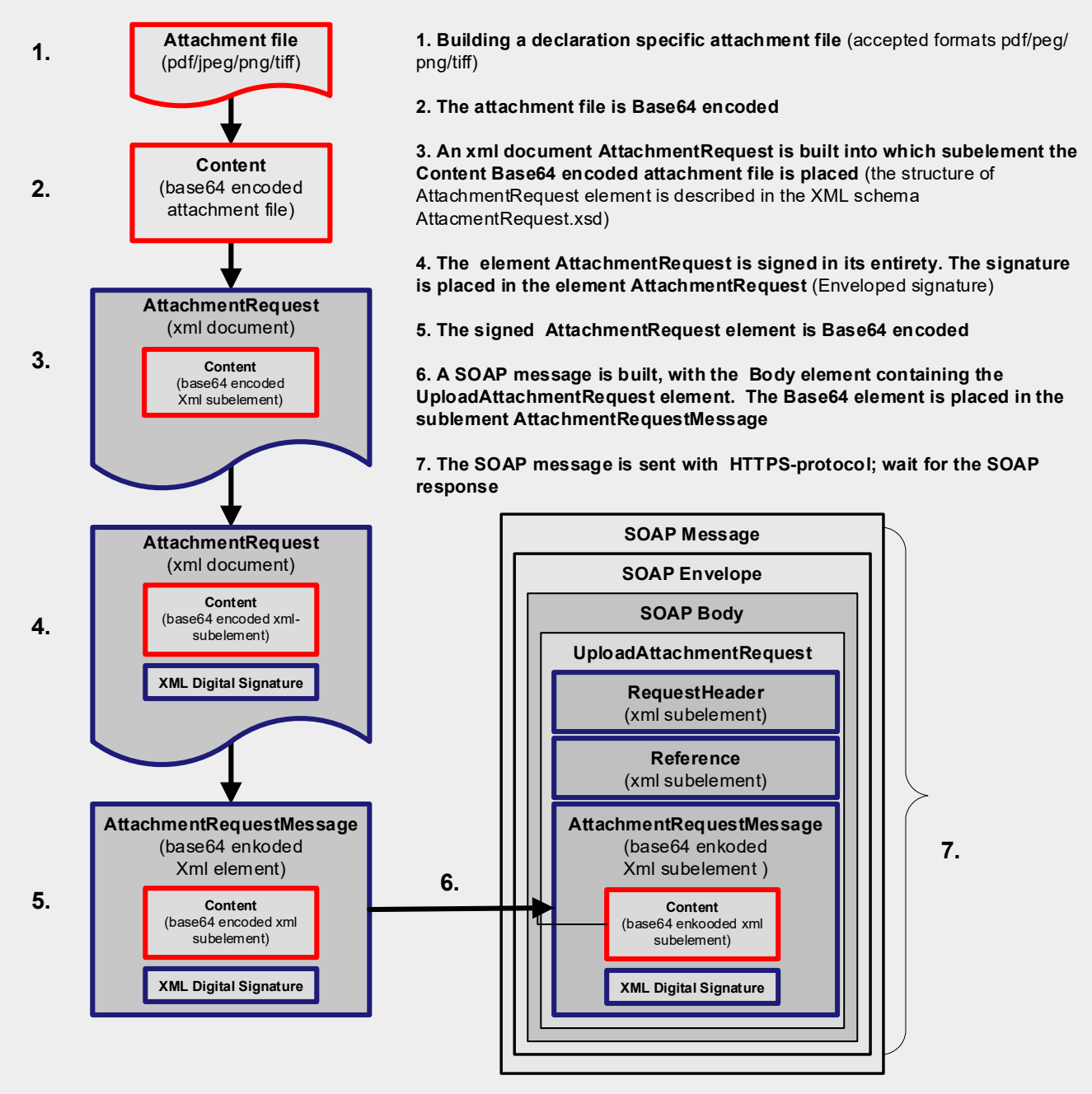

Figure 8: UploadAttachment-sanoman muodostaminen to Customs

An AttachmentRequest block is formed around the base64 encoded attachment file. In addition to the attachment file, it contains attachment file metadata as well as the signature of the builder. This signature covers the whole AttachmentRequest document.

The intermediary embeds this complete, base64 encoded and signed AttachmentRequest document, as the value for the AttachmentRequestMessage element.

4 WDSL and XDS files of Customs

The message exchange interface is described using a WSDL definition, which is an XML-based language for describing web services. The WDSL definition of the Customs direct message exchange is realised in the file:

- CustomsCorporateService.wsdl

The WSDL refers to the following XML schema files, in which the used message structures are described:

- ApplicationRequest.xsd
- ApplicationResponse.xsd
- ApplicationMessageTypes.xsd
- AttachmentRequest.xsd
- EchoContent.xsd
- WsdlTypes.xsd
- xmldsig-core-schema.xsd

If the customer wants to use the Message Notification Service, he should also look at:

The WSDL definitions for the Message Notification Service

- NotificationService.wsdl

which refers to the following XML schema file

- NotificationTypes.xsd

The files are provided as a package (ZIP archive) that can be downloaded from the following address on the Customs website:

http://tulli.fi/en/e-services/services/direct-message-exchange

The above address is the only place from where to download the WSDL. The WSDL **cannot** be downloaded by sending a HTTP GET '?wsdl' request to the URL of the web service.

5 Schema error message

Customs checks the application message once Customs has acknowledge the receipt of the direct message exchange Upload message. One way that Customs checks the correctness of the application message in XML format is by validating it against the message schema.

Customs rejects the application message, if it violates the XML schema. Customs send a schema error message of the rejection to the customer. The customer retrieves the schema error message via the direct message exchange in the same way as other Customs' response messages. Data on the sender of the schema error message is Customs' application, to which the customer's application message with errors was sent.

All Customs applications, except the summary declaration service AREX and the export system ELEX, send a harmonised schema error message when Customs rejects the customer's application message as being in violation of the XML schema. The structure of the schema error message is described below. (in this error situation, AREX and ELEX send a response message, which has been described in the AREX and ELEX message descriptions.)

5.1.1 DmeErrorMessage structure

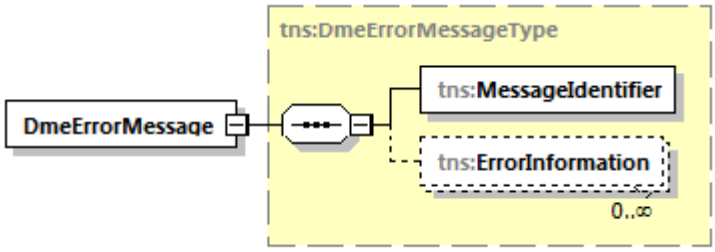

Figure 9: DmeErrorMessage structure

Figure 10: DmeErrorMessage – MessageIdentifier structure

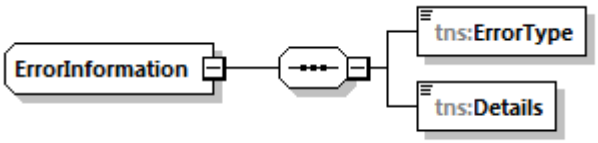

Figure 11: DmeErrorMessage – Error Identifier structure

5.1.2 DmeErrorMessage, element description

Element name	Description
DmeErrorMessage	Document root element

Subelement name	Description	Type	Required (Y/N)	Occurrence
MessageIdentifier	Element with header data	Message Identifier	K	[1..1]

ErrorInformation	Element with error information	ErrorInformation	K	[0..*]

Table 1: DmeErrorMessage - root element

Element name	Description
MessageIdentifier	Element with header data

Subelement name	Description	Type	Required (Y/N)	Occurrence
MessageSender	Country code and Business ID of the message sender (Customs)	string	K	[1..1]
MessageRecipient	Country code and Business ID of the recipient	string	K	[1..1]
ErrorDateTime	Time stamp for when the error was observed	dateTime	K	[1..1]
Application	Short name of the Customs application to which the schema error message is connected.	string	K	[1..1]
ControlReference	The identifier originally generated by the customer and with which the customer can bundle together all the messages related to the transaction (to the customs declaration). More information in chapter **Virhe. Viitteen lähdettä ei löytynyt.**. (Reference element)	string	K	[1..1]
OriginalMessageStorageId	Unique identifier of the message, through which the customs declaration connected to the attachment file is stored in the Customs system.	string	K	[1..1]

Table 2: DmeErrorMessage – MessageIdentifier element

Element name	Description
ErrorInformation	Element with error information

Subelement name	Description	Type	Required (Y/N)	Occurrence
ErrorType	Error type, always SCHEMA	Message Identifier	K	[1..1]
Details	Detailed information on the error; generally information printed by the XML parser or the schema validator	ErrorInformation	K	[1..1]

Table 3: DmeErrorMessage – ErrorInformation element

6 Attachment files in reply messages

Customs returns customs decisions in XML format in the response message to the customers Download request message. A few special cases of returning documents are presented in this chapter. Other customs decisions to be returned are presented in the documentation regarding the relevant customs clearance practice.

6.1 PDF files

With some response messages of Customs, an electronic document is also sent as a PDF file. The customer prints out the document to be used at the different stages of the Customs clearance process and for filing.

In the export and transit systems, the PDF files are packed into a ZIP archive, because they take up less space when compressed. The response messages concerning import are sent to the customer systems in PDF format.

The customer downloads the XML response message created by Customs and the documents in PDF format or the ZIP archive containing the documents simultaneously by sending a Download request to the web service for direct message exchange. Both of these data contents are included in the same Download response by the web service.

The XML response message and the PDF/ZIP archive are base64 encoded and added to the XML document ApplicationResponse. The ApplicationResponse is base64-encoded and inserted into the Download response, which the web service returns to the customer.

The size of the PDF file or the ZIP archive must not exceed 2 megabytes (20148 kilobytes). If the size of the PDF file or the ZIP archive exceeds 2 megabytes, it cannot be delivered to the customer through the direct message exchange web service. The customer can agree, case by case, with the Electronic Service Centre of Customs on the delivery of files exceeding the maximum size limit.

6.2 LiituResponseMessage (conversion result of the attachment files)

The Customs integration layer converts the attachment file, sent by the customer, into a format understood by the relevant customs clearance application. Every attachment file sent by the customer and converted by Customs integration layer receives a rejection or acceptance message, which is returned to the message storage of Customs' integration layer for retrieval by the customer. The system's application identifier (Application) of the rejection or acceptance message is always 'LIITU' no matter to which customs clearance application the customer has sent the attachment file.

The acceptance or rejection of the attachment file sent by the customer is presented to the customer in a returning LiituResponseMessage. The attachment file conversion is always either accepted or rejected. An accepted conversion does not require any further measures by the customer, rather the attachment file continues for control in the relevant customs clearance application. If an attachment is rejected, the processing is discontinued and the rejection message sent to the customer contains an itemised list of the reasons for the rejection. If the attachment has been requested by Customs, the customer must send a corrected attachment file to Customs.

6.2.1 LiituResponseMessage, structure

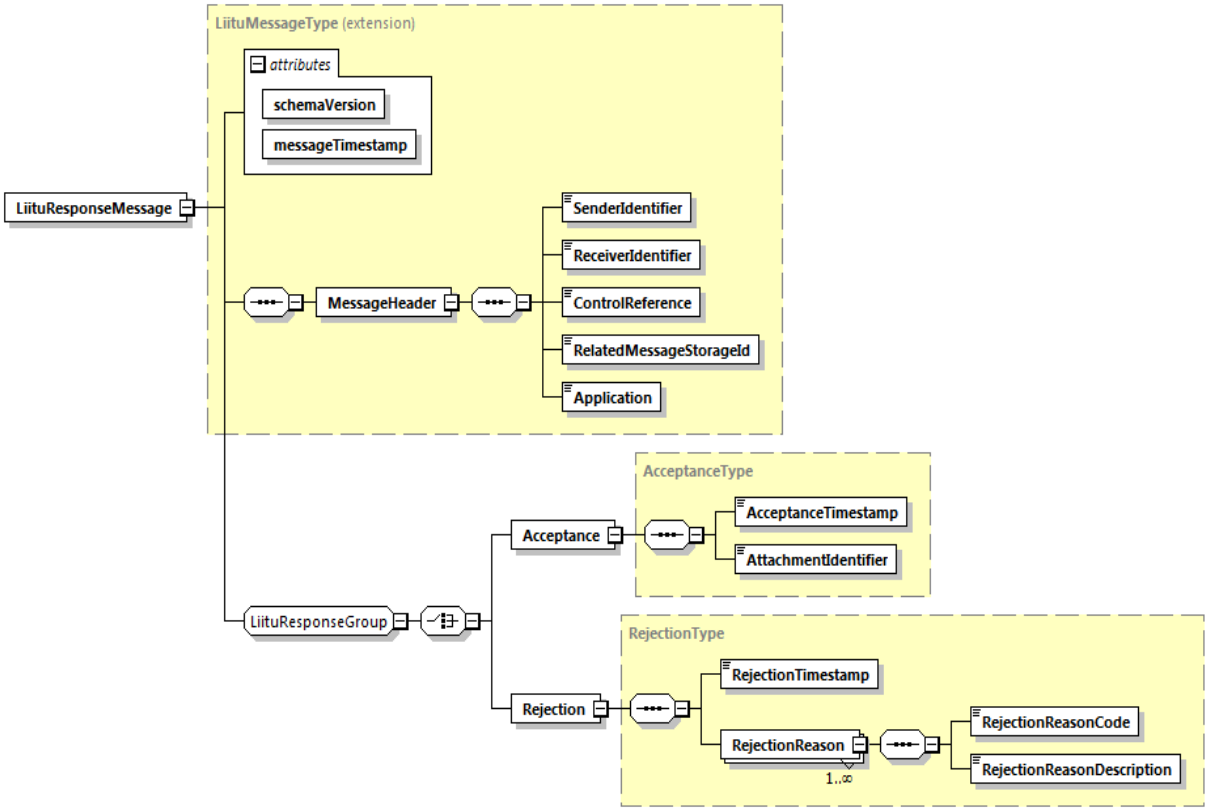

Figure 12: Description of LiituResponseMessage

6.2.2 LiituResponseMessage, description of elements

Element name	Description
LiituResponseMessage	Document root element

Subelement name	Description	Type	Required (Y/N	Occurrence
MessageHeader	Element containing joint message data	Message Header	Y	[1..1]
LiituResponseGroup	Made up of an AcceptanceType element or a RejectionType element, depending on whether or not the attachment has been accepted or not	AccpetanceType or RejectionType	Y	[1..1]

Table 4: LiituResponseMessage, root element

Element name	Description			
MessageHeader	Element with header data			
Subelement name	**Description**	**Type**	**Required (Y/N)**	**Occurrence**
SenderIdentifier	Sender (Customs) identifier: country code and Business ID	string	Y	[1..1]
ReceiverIdentifier	Receiver identifier: country code and Business ID	string	Y	[1..1]
ControlReference	The interchange identifier originally provided by the customer, with which the customer can bundle together all messages connected to the transaction (attachment to the transmission). More information in chapter **Virhe. Viitteen lähdettä ei löytynyt.**. (Reference element)	string	Y	[1..1]
RelatedMessageStorageId	A unique message identifier with which the customs declaration has been saved in Customs' systems. attachment file.	string	Y	[1..1]
Application	Short name of the Customs application to which the message has been sent and the attachment file belongs.	string	Y	[1..1]

Table 5: LiituResponseMessage – MessageHeader element

Element name	Description			
Acceptance	Element describing an accepted modification of the attachment file			
Subelement name	**Description**	**Type**	**Required (Y/N)**	**Occurence**
AcceptanceTimestamp	Timestamp of attachment modification	dateTime	Y	[1..1]
AttachmentIdentifier	Identifier in the Liitu system of the modified attachment	string	Y	[1..1]

Table 6: LiituResponseMessage – Acceptance element

Element name	Description			
Rejection	Element describing a rejected modification of an attachment			
Subelement name	**Description**	**Type**	**Required (Y/N)**	**Occurrence**
RejectionTimestamp	Timestamp of attachment modification	string	Y	[1..1]
RejectionReason	Element containing the modification error resulting in rejection of the attachment	RejectionReason	Y	[1..*]

Table 7: LiituResponseMessage – Rejection element

Element name	Description			
RejectionReason	Element describing a single modification rejection			
Subelement name	**Description**	**Type**	**required (Y/N)**	**Occurrence**
RejectionReasonCode	Identification code of attachment modification rejection	string	Y	[1..1]
RejectionReasonDescription	Description of attachment modification rejection	string	Y	[1..1]

Table 8: LiituResponseMessage – RejectionReason element

6.3 Metadata message of PDF document (DmeDocumentInfoMessage)

Customs' new customs clearance system (UTU) builds the decision documents based on customs declarations in a different way than the previous customs clearance systems. In UTU, the documents in XML format and PDF format must be retrieved from Customs' message storage with a separate web service operation, contrary to the old customs clearance systems.

A PDF metadata message is always connected to a new document in PDF format. The structure of the metadata message is presented in this chapter.

6.3.1 DmeDocumentInfoMessage, element description

Element name	Description			
DmeDocumentInfoMessage	Data structure containing metadata in the PDF document			
Subelement name	**Description**	**Type**	**Required (Y/N)**	**Occurrence**
MessageRecipient	Country code and Business ID of the recipient (customer) of the document	string	K	[1..1]
DocumentId	Unambiguous identifier of the PDF document	string	K	[1..1]
RelatedMessageStorageId	Unambiguous identifier of an XML based decision document in Customs direct message exchange interface	string	K	[1..1]
ControlReference	Unambiguous identifier of a customs clearance case that binds the customs declaration and the customs decisions together (both XML and PDF) Provided by customer	string	E	[0..1]
CustomsDocumentReference	Customs' system generated identifier for the customs clearance case, usually an MRN.	string	E	[0..1]
DocumentType	Document MIME media type, application/PDF	string	K	[1..1]
DocumentCreationTime	Creation time of the PDF document in the Customs system. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used as default.	datetime	K	[1..1]

Table 9: DmeDocumentInfoMessage, element description

6.3.2. PDF document retrieval from the direct message exchange interface

The customer receives information on PDF documents and metadata messages for PDF documents in the same way as for normal decisions in XML format: either from the direct message exchange DownloadList response, or alternatively, via a message sent by Customs from the Message Notification Service.

The metadata messages for PDF documents are retrieved from the direct message exchange interface in the same way as normal decisions in XML format: with the Download operation.

The document in PDF format is also retrieved with the Download operation from the direct message exchange interface, but then the unique DocumentId element of the metadata message for the PDF document must be added to the DownloadMessageFilteringCriteria element of the DownloadRequest message.

7 Operations provided by the service

The Direct Message Exchange web service contains the following SOAP operations:

7.1 Services implemented by Customs

Upload = Used for delivering a set of data, for example a customs declaration, to a Customs system.

UploadAttachment = Used for delivering a customs declaration attachment file to Customs' systems. The declaration has to be sent before the attachment that relates to it is sent.

DownloadList = Returns a list of messages that are waiting to be downloaded. The data obtained by using this operation can be used as a search key in the download operation.

Download = Used for downloading a set of data from a Customs system. For example, the operation can be used for downloading the response from Customs to a customs declaration sent earlier by the customer.

CheckConnectivity = Used in testing in order to verify the technical compatibility of the customer's systems with the Customs systems.

7.2 Service implemented by the customer

If the customer uses the Message Notification Service, Notify implemented by Customs calls the customer's service.

Notify = Calls the service implemented by the customer in order to notify the customer of a response message waiting in a Customs system to be downloaded.

7.3 Service operations in WSDL

In WDSL the service functionality is grouped into logical operations. An operation is usually made up of two related (main) data elements: request and response. The message exchange interface provides operations similar to the ones shown in the tables below:

7.3.1 Upload

Operation	Description
Upload	Used for uploading one set of data, e.g. a customs declaration to Customs' systems. As a result, a technical acknowledgement of reception of the message (= being processed) is received. After the technical acknowledgement, the set of data, e.g. a customs declaration, is processed in the Customs system. Further processing (usually) provides a response message, which has to be downloaded separately with a separate download operation. A list of downloadable messages can be

	downloaded with the DownList operation (see the following operations) or if the Message Notification Service is in use the Customs system sends a message notification of a message waiting to be downloaded.
Request	
UploadRequest	
RequestHeader	
ApplicationRequestMessage	
Response	
UploadResponse	
ResponseHeader	
MessageInformation	

Table 10: Upload

7.3.1.1 Upload, request

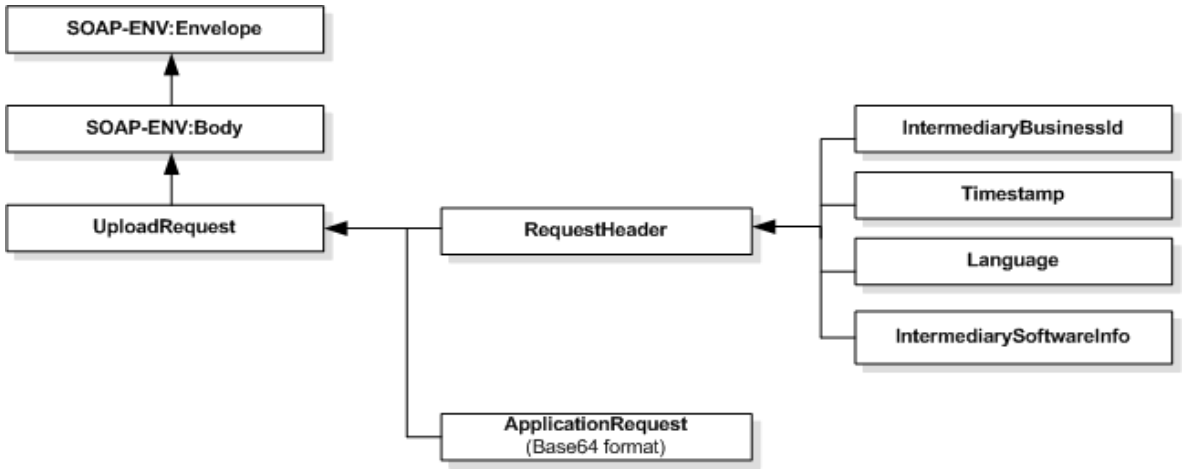

Figure 12: Upload, request

6.3.1.2 ApplicationRequest:

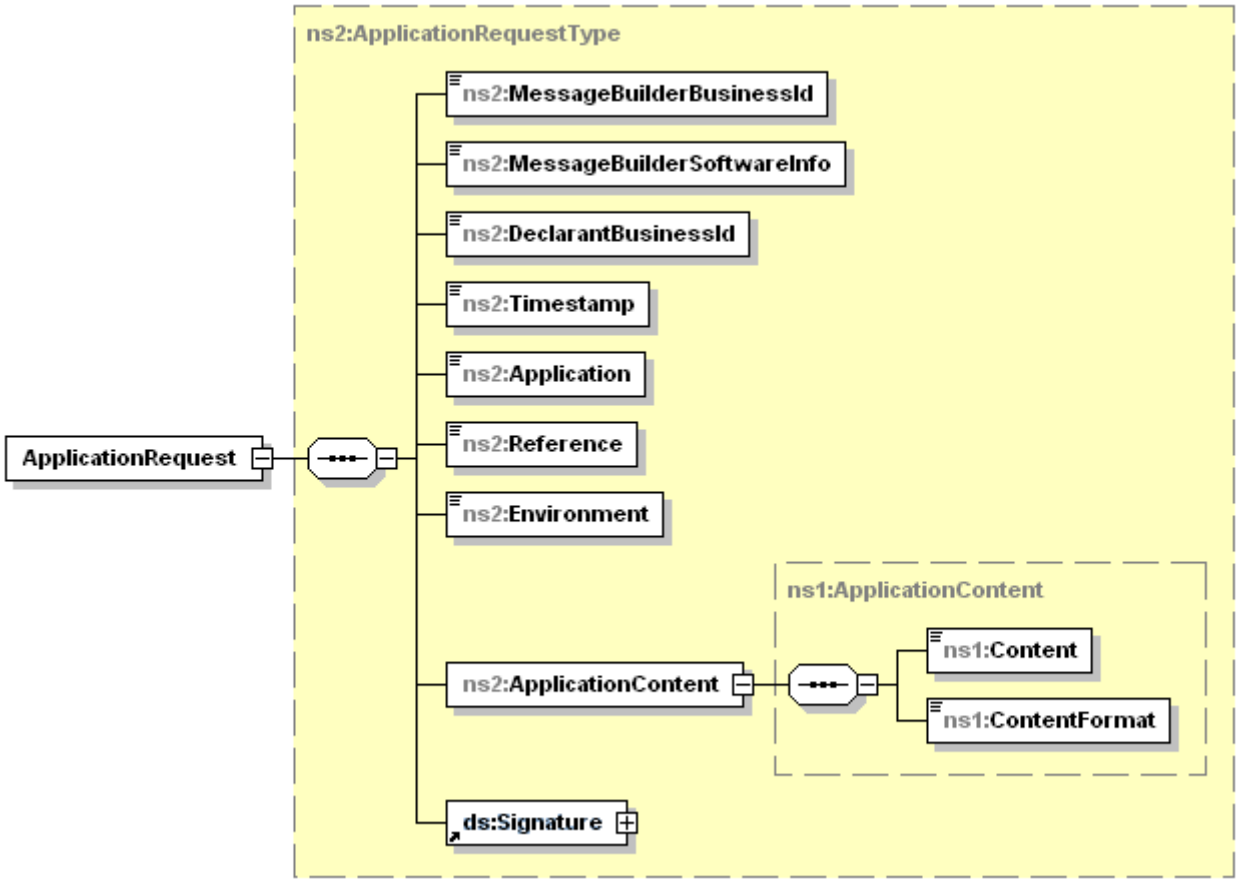

Figure 13: Description of ApplicationRequest

7.3.1.3 Upload, response

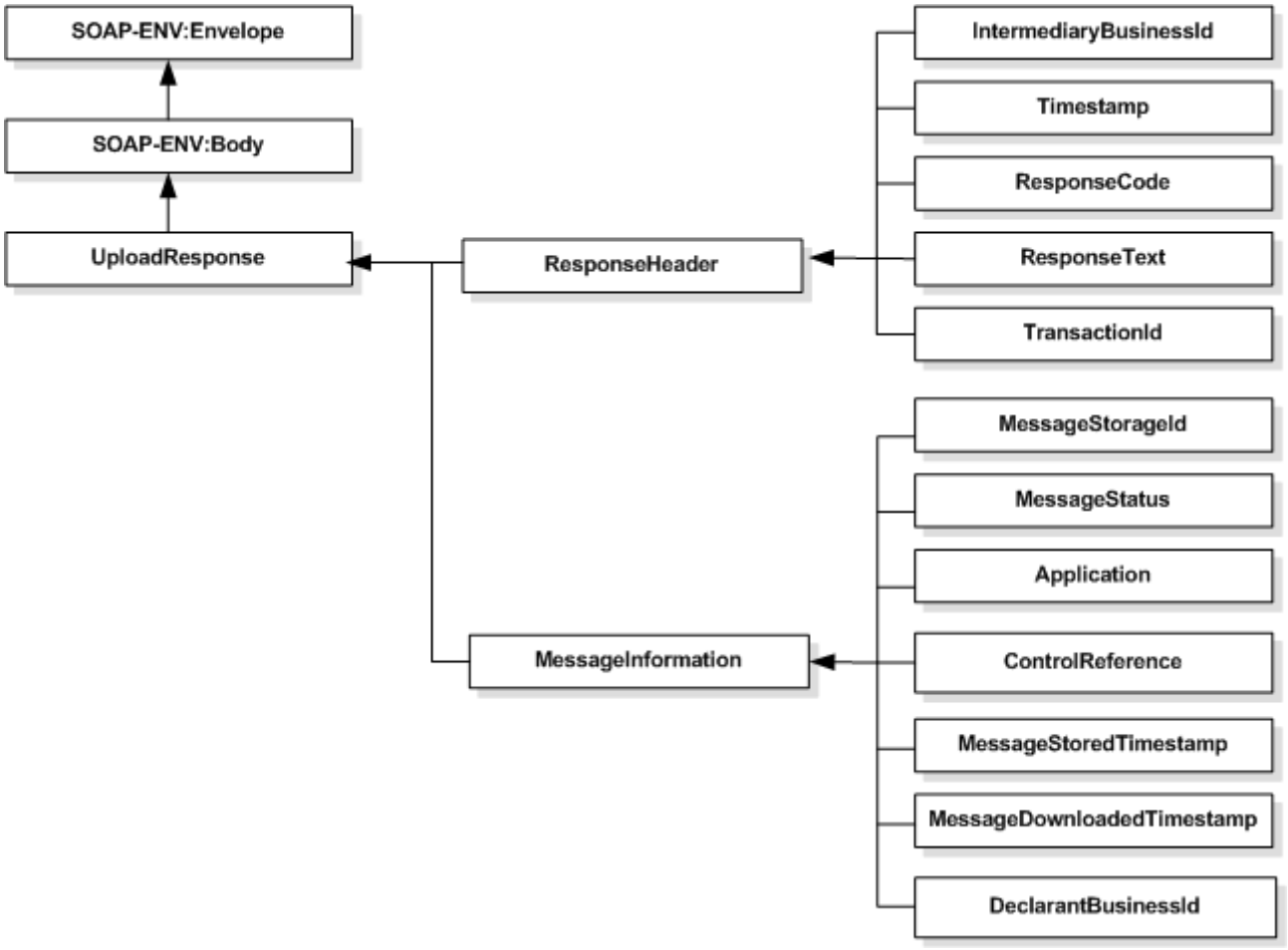

Figure 14: Upload, response

7.3.2 UploadAttachment

Operation	Description
UploadAttachment	Used for uploading a message containing one attachment to Customs' systems. As a result, a technical acknowledgement of reception of the message (= being processed) is received. After the technical acknowledgement, the message attachment is processed in the Customs system. The process finally produces a response message, which content indicates that the processing of the attachment was successful. The response message has to be retrieved separately with a separate download operation. A list of downloadable messages can be retrieved with the DownloadList operation (see the following operations) or if the Message Notification Service is in use, the Customs system sends a message notification of a message waiting to be downloaded. A response message stating that the attachment is being processed, does not guarantee that it has been accepted. The attachment is only accepted if it is a part of a declaration message that has been accepted. More information can be found in the

	document ‘Introduction to message exchange with Finnish Customs’ in chapter 2.1.4 ‘Operations and processes when using the attachment file message service’.
Query	
UploadAttachmentRequest	
RequestHeader	
Reference	
AttachmentRequestMessage	
Response	
UploadAttachmentResponse	
ResponseHeader	
AttachmentRequestMessageInformation	

Table 11: UploadAttachment

7.3.2.1 UploadAttachment, request

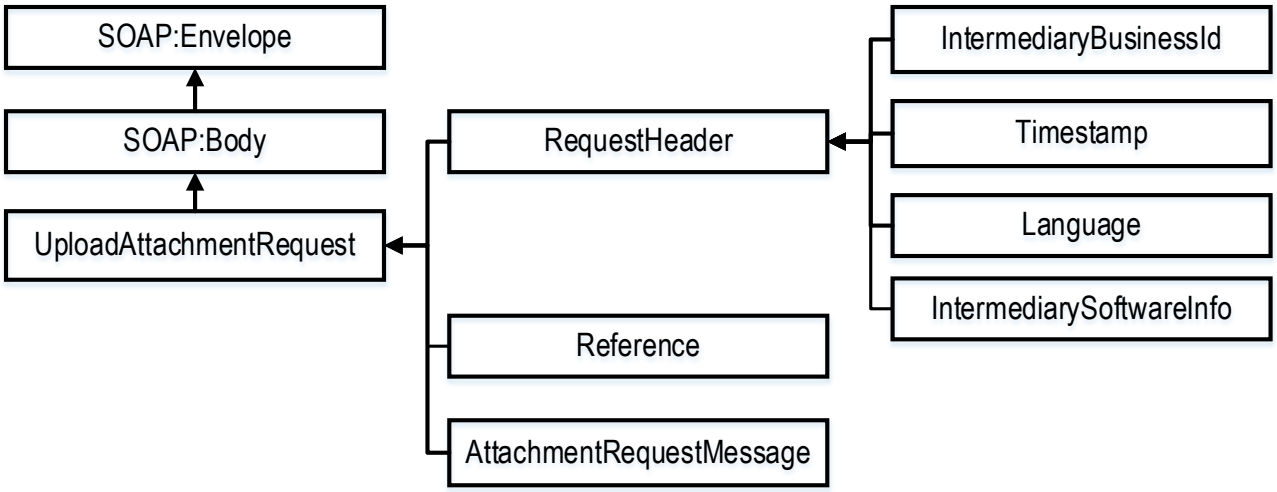

Figure 15: UploadAttachment, request

7.3.2.2 AttachmentRequest

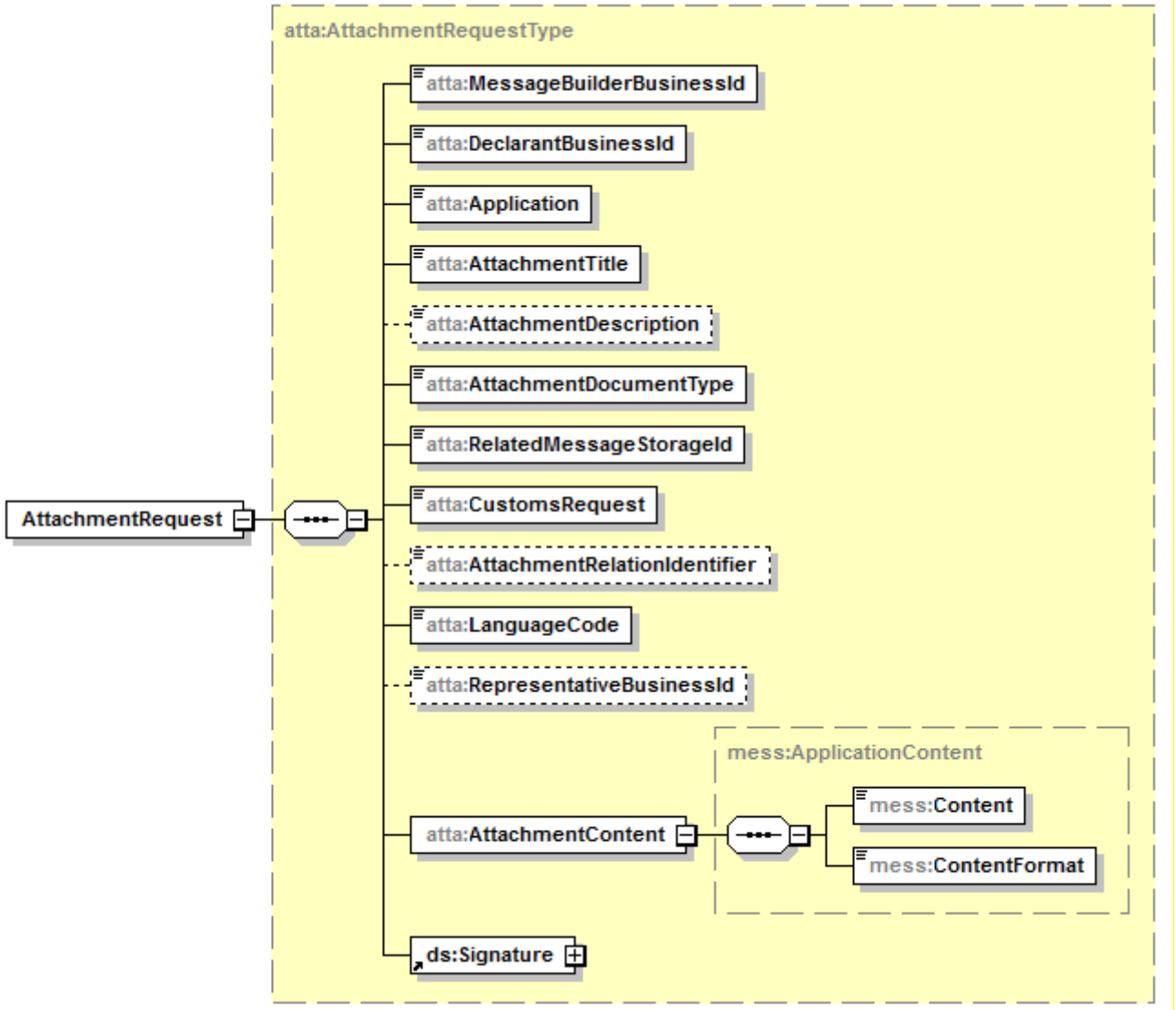

Figure 16: Description of AttachmentRequest

7.3.2.3 UploadAttachment, response

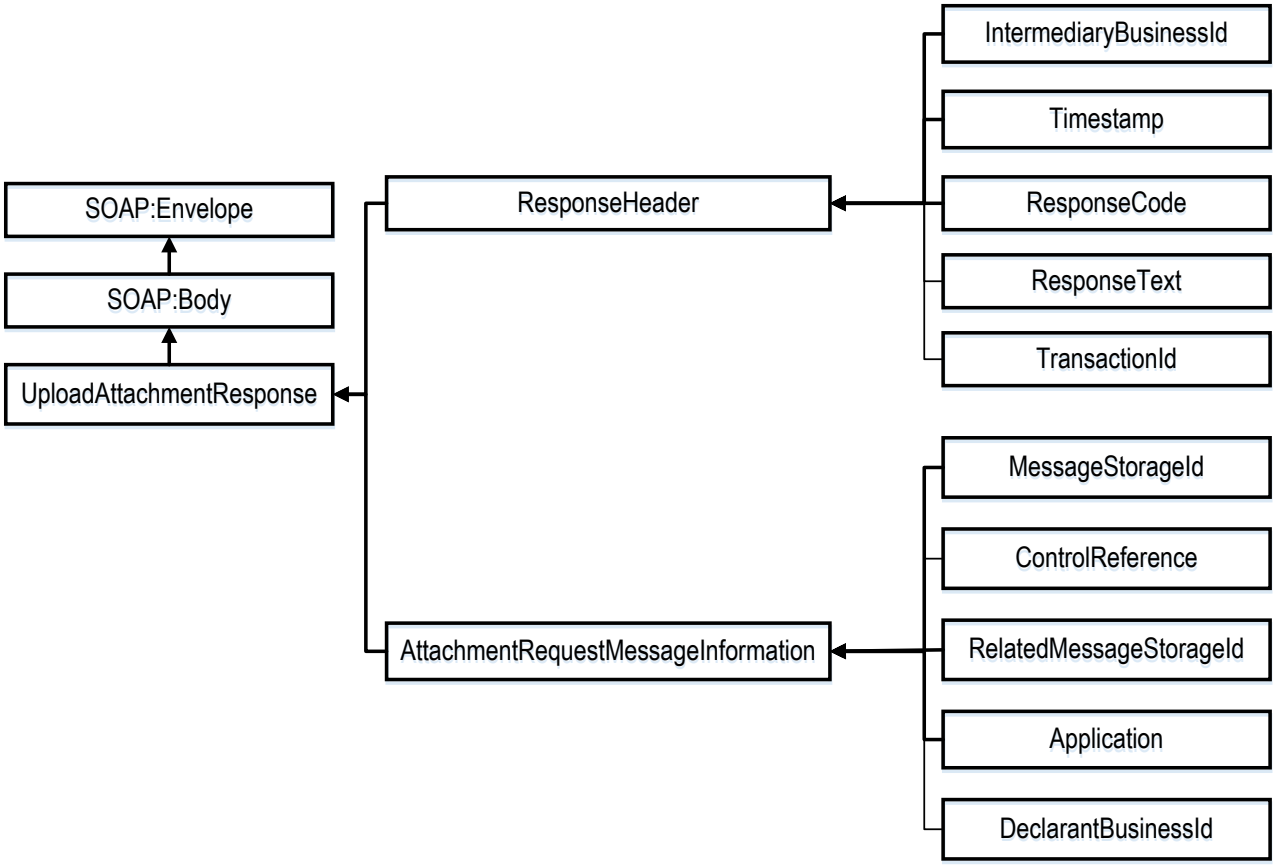

Figure 17: UploadAttachment, response

7.3.3 DownloadList

Operation	Description
DownloadList	Used for downloading a data list from the Customs systems. Contains a list of the basic information of response messages waiting to be downloaded. The information gained through the response of this operation can be used as a search key in the download operation.
	Request
	DownloadListRequest RequestHeader DownloadMessageListFilteringCriteria
	Response
	DownloadListResponse ResponseHeader DownloadMessageListFilteringCriteria MessageInformation

Table 12: DownloadList

7.3.3.1 DownloadList, request

The search time window for a DownloadList request can be determined by using Timestamp or Date elements.

Using Timestamps is recommended, because the Timestamp element can limit the search more, which makes the search function faster and more efficient. Especially if the DownloadList request is made e.g. every five minutes, it is better to request messages for the last hour or half an hour than for the last 24 hours.

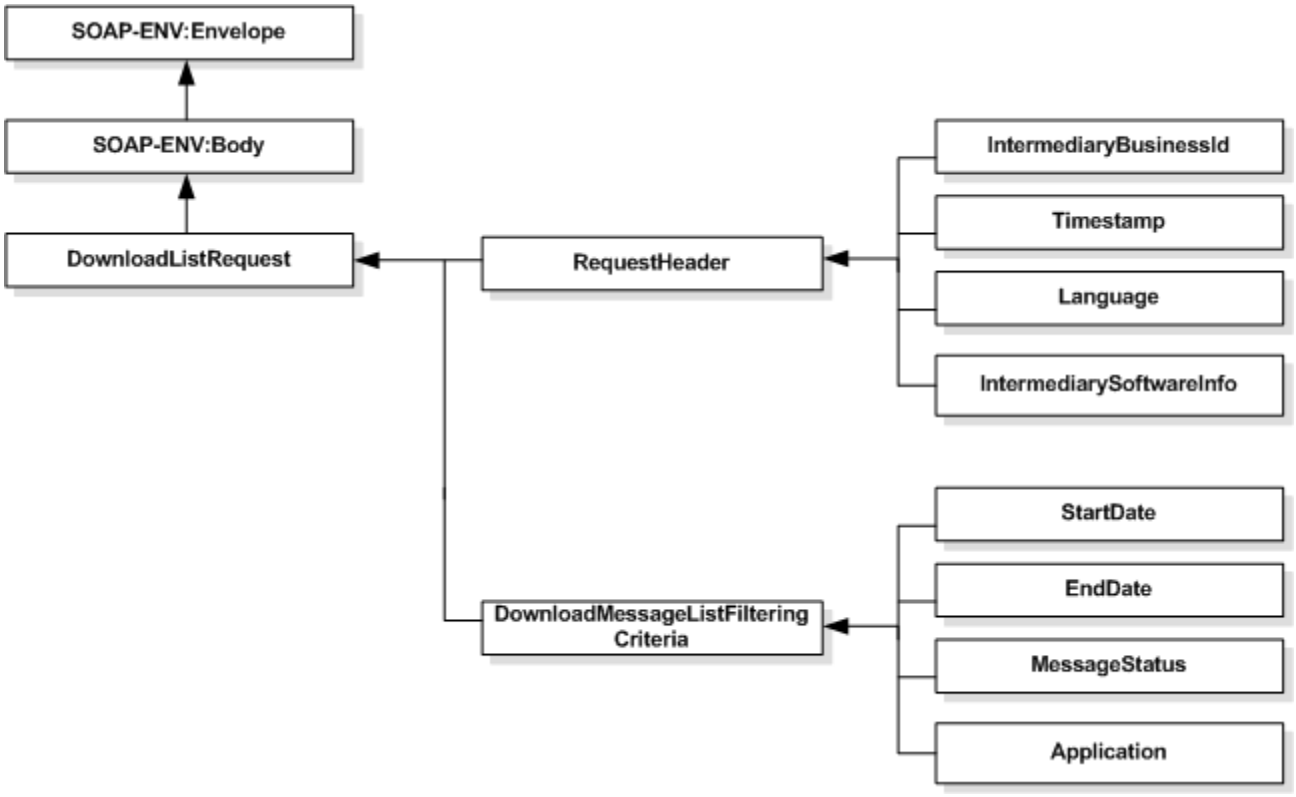

Figure 18: DownloadList, request with elements Startdate and EndDate

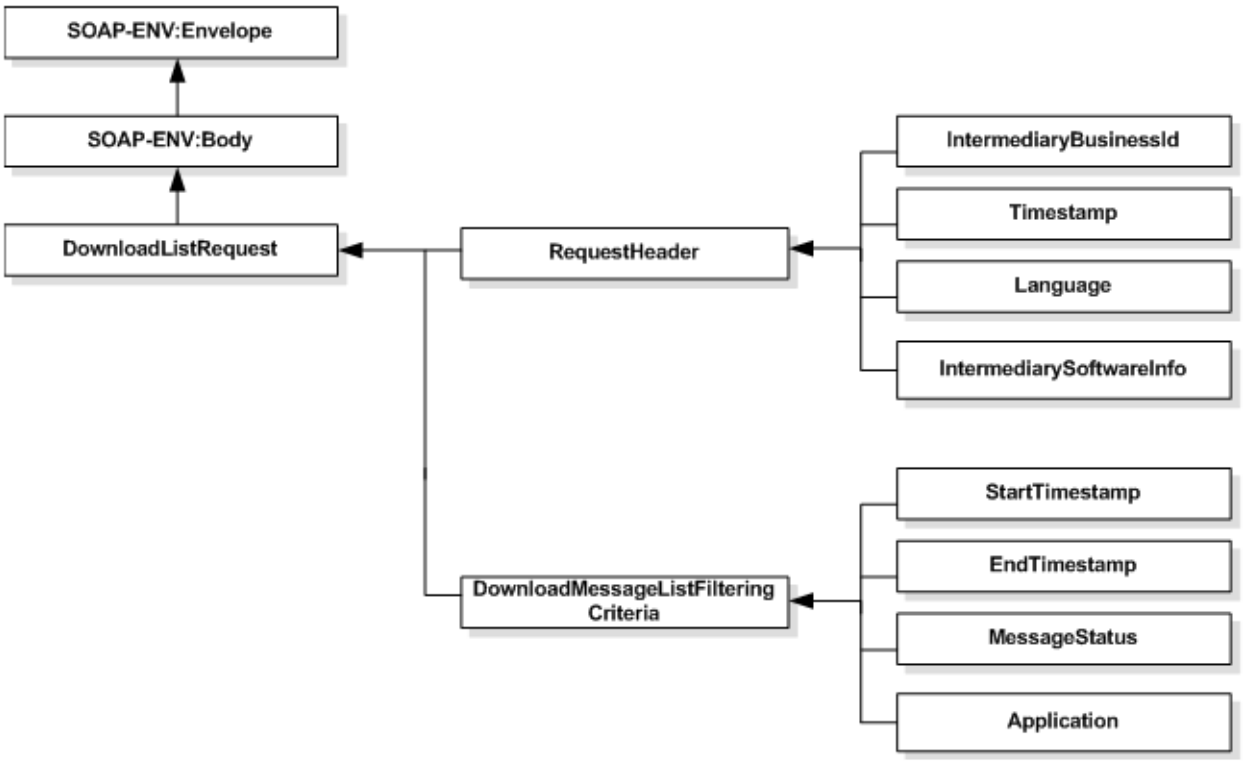

Figure 19: DownloadList, request with elements StartTimestamp and EndTimestamp

DownloadList, response

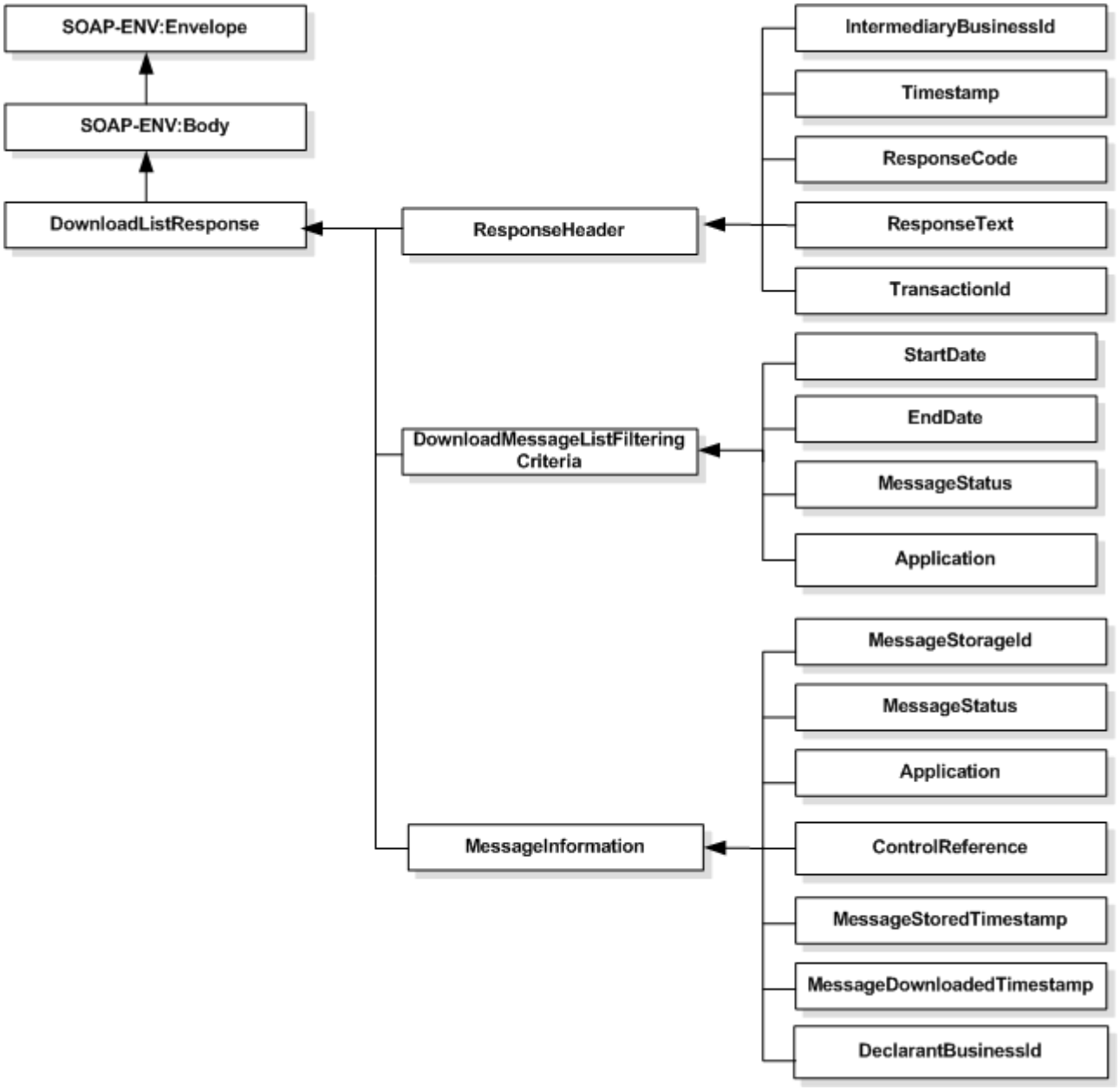

Figure 20: DownloadList, response

7.3.4 Download

Operation	Description
Download	Used for downloading one set of data from the Customs systems. For example, the operation can be used for downloading the response from Customs to a customs declaration sent earlier.
	Request
	DownloadRequest RequestHeader DownloadMessageFilteringCriteria
	Response
	DownloadResponse ResponseHeader ApplicationResponseMessageInformation ApplicationResponseMessage

Table 13: Download

7.3.4.1 Download, request

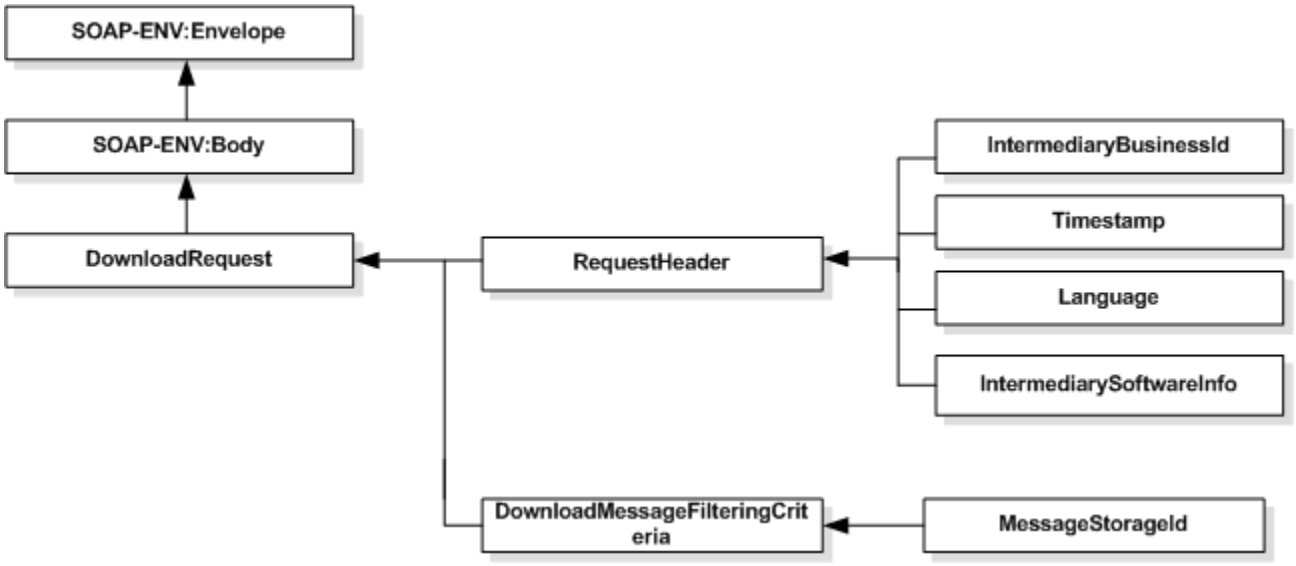

Figure 21: Download, request

7.3.4.2 Download, response

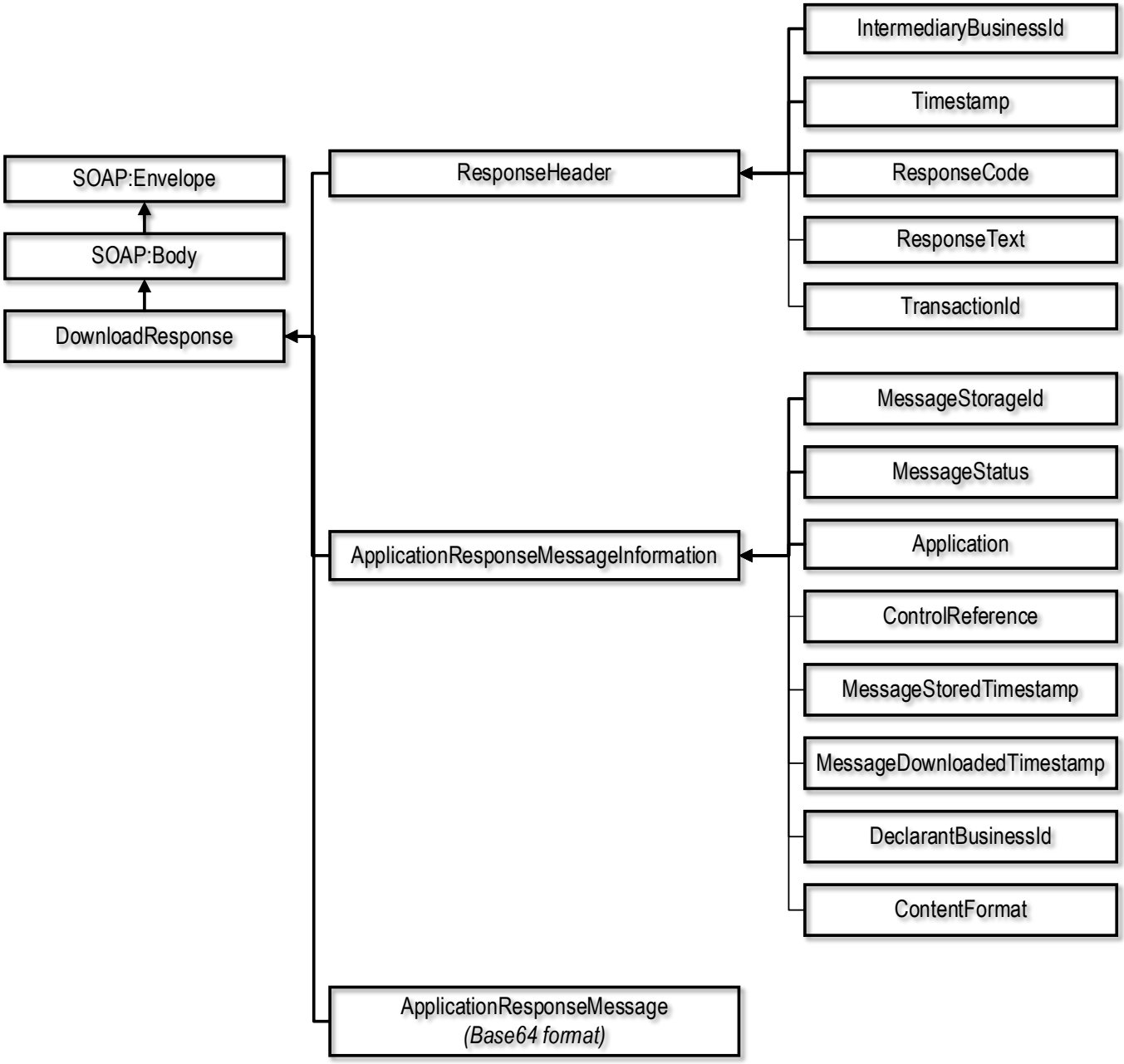

Figure 22: Download, response

7.3.4.3 ApplicationResponse

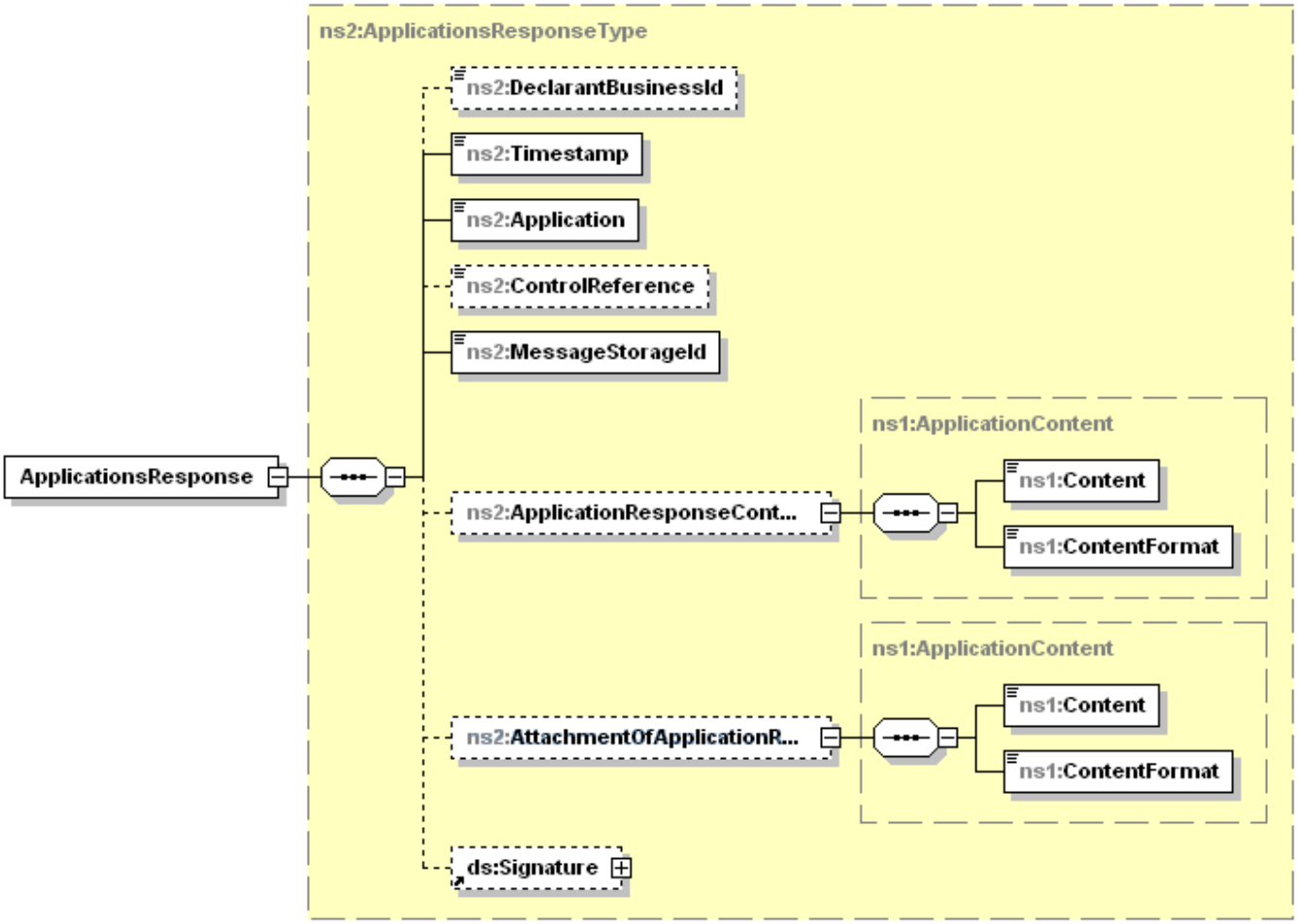

Figure 23: Description of ApplicationResponse

7.3.5 Notify

Operation	Description
Notify	Used when Customs notifies the customer of a message waiting to be downloaded. Requires that the customer has built a service that Customs can call. One service call is sent for every message waiting to be downloaded.
	Request
	NotifyRequest RequestHeader MessageInformation
	Response
	NotifyResponse ResponseHeader ResponseCode ResponceText

Table 14: Notify

7.3.5.1 NotifyRequest (From Customs)

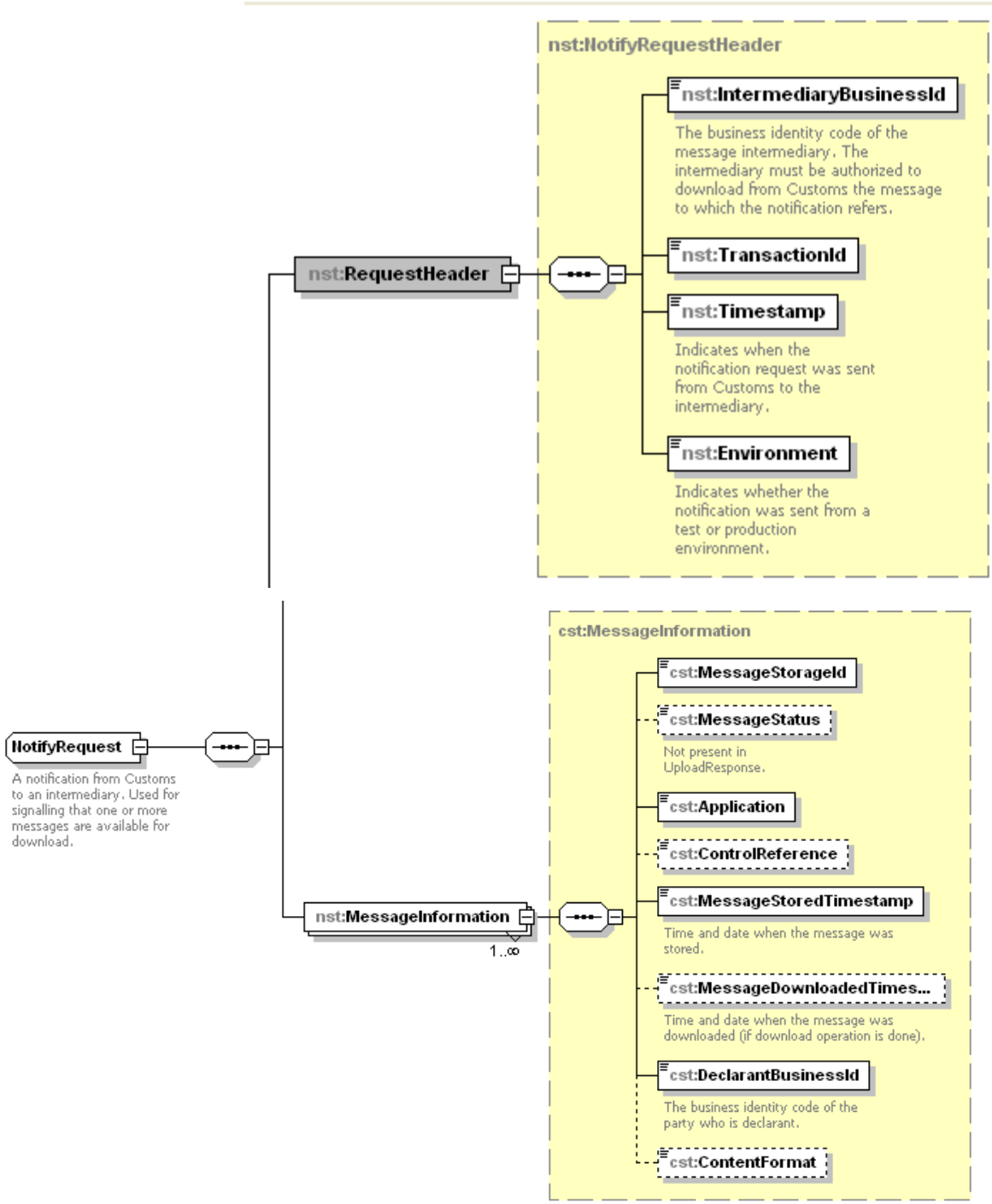

Figure 24: Description of NotifyRequest from Customs

7.3.5.2 NotifyResponse (from the customer)

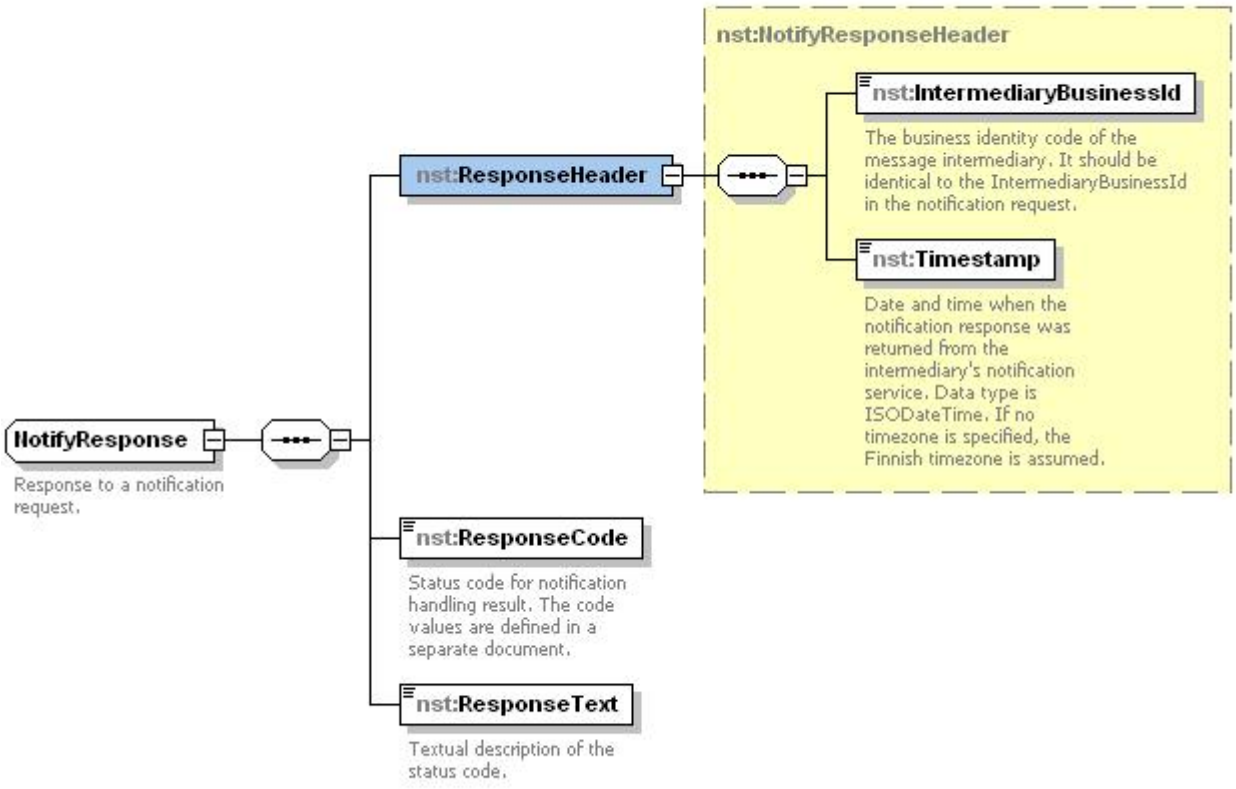

Figure 25: Description of NotifyResponse from the customer

7.3.6 CheckConnectivity

Operation	Description
CheckConnectivity	Used in testing in order to verify the technical compatibility of the customer's systems with the Customs systems. In practice, the verification relates to the functionality of the message exchange between the parties through the HTTPS/SOAP protocol. In addition, the functionality of the certificates of the intermediary and the XML message builder (the authentication and authorisation of the parties) is checked.
	Request
	CheckRequest RequestHeader EchoRequest
	Response
	CheckResponse ResponseHeader EchoResponse

Table 15: CheckConnectivity

8 Descriptions of XML element data

8.1 RequestHeader

Element name	Description
RequestHeader	This element is the header for each request sent to the WS interface. The intermediary constructs this header.

Element name	Description	Type	Required (Y/N)	Occurrence
IntermediaryBusinessId	Country code and Business ID of the intermediary	string	Y	[1..1]
Timestamp	Time and date of sending the request Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used as default time.	dateTime	Y	[1..1]
Language	Language code indicating which language is used in the informative data elements. Presently, only EN (English) is supported.	string	N	[0..1]
IntermediarySoftwareInfo	Name of the software and version number used by the intermediary.	string	Y	[1..1]

Table 16: RequestHeader

8.2 ApplicationRequestMessage

Element name	Description	Type	Required (Y/N)	Occurrence
ApplicationRequestMessage	Contains the XML message by the builder. XML is base64-encoded.	base64-Binary	Y	[1..1]

Table 17: ApplicationRequestMessage

8.3 ResponseHeader

Element name	Description
ResponseHeader	This element is the header for each responded request of the WS interface.

Element name	Description	Type	Required (Y/N)	Occurrence
IntermediaryBusinessId	Country code and Business ID of the intermediary of the request message.	string	Y	[1..1]
Timestamp	Time and date of sending the response. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	dateTime	Y	[1..1]
ResponseCode	Response code indicating the success or failure of the operations followed by the request (see Appendix 1).	string	Y	[1..1]
ResponseText	Verbal description of the response code (in English) (see Appendix 1).	string	Y	[1..1]
TransactionId	Unique transaction ID generated by the integration layer of Customs, with which the request and the response can be bundled together.	string	Y	[1..1]

Table 18: ResponseHeader

8.4 MessageInformation

Element name	Description
MessageInformation	This element contains the basic information about the message.

Element name	Description	Type	Required (Y/N)	Occurrence
MessageStorageId	Unique ID of the message, through which the response message can be downloaded from the Customs system.	string	Y	[1..1]
MessageStatus	Message status: • NEW – The message has not yet been downloaded • DLD – The message has already been downloaded • ALL – All messages	string	N	[0..1]
Application	Short name of the Customs application to which the message is connected.	string	Y	[1..1]
ControlReference	The interchange identifier originally generated by the customer (the builder of the XML message), with which the customer can bundle together all the messages related to one transaction. Additional information in the description of the subelement *Reference* of the element *ApplicationRequest*.	string	N	[0..1]
MessageStoredTimes-tamp	Date and time (time stamp) when the message was saved in the Customs system. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	dateTi-me	Y	[1..1]
MessageDownlo-adedTimestamp	Date and time (time stamp) when the message was downloaded from the Customs system (if the download operation has been performed). Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	dateTi-me	N	[0..1]
DeclarantBusinessId	Country code and Business ID of the message declarant	string	Y	[1..1]
ContentFormat	Format of the Content element. Possible values are MIME media types. As customers can only transmit XML messages to Customs, ’application/xml’ must be set as value. Earlier, the only allowed value was ’XML’. It can still be used. In DownloadResponse response messages with PDF attachments, ContentFormat is not a required fiels on SOAP level.	string	Y	[0..1]

Table 19: MessageInformation

8.5 AttachmentRequestMessage

Element name	Description	Type	Required (Y/N)	Occurrence
AttachmentRequestMessage	Contains a base64 coded AttachmentRequest element	base64Binary	Y	[1..1]

Table 20: AttachmentRequestMessage

8.6 AttachmentResponseMessage

Element name	Description
AttachmentResponseMessage	The message generated by the integration layer of Customs, contains the identifiers of the processed attachment. Included as a subelement of the successful UploadAttachmentResponse message.

Element name	Description	Type	Required (Y/N)	Occurrence
MessageStorageId	Unique ID of the message, by which the response message is saved in the Customs system.	string	Y	[1..1]
ControlReference	The interchange identifier originally generated by the customer and with which the customer can bundle together all the messages related to the transaction (the sending of the attachment). Additional information in the description of the subelement Reference of the element ApplicationRequest.	string	Y	[1..1]
RelatedMessageStorageId	A declaration file identifier, provided by Customs for the attachment file, can be found in the UploadResponse message of the relevant declaration message.	string	Y	[1..1]
Application	The short name of the Customs application to which the actual attachment file message placed in the AttachmentContent element is directed.	string	Y	[1..1]
DeclarantBusinessId	Country code and Business ID of the message declarant	string	Y	[1..1]

Table 21: AttachmentResponseMessage

8.7 DownloadMessageListFilteringCriteria

Element name	Description
DownloadMessageListFilteringCriteria	This element data is used as filtering criteria when requesting (downloadList) which response messages can be downloaded one by one.

Element name	Description	Type	Required (Y/N)	Occurrence
StartDate	The start date, during which or after which the message has been saved. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	date	Y	[1..1]
EndDate	The end date, during which or before which the message has been saved. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	date	Y	[1..1]
StartTimestamp	The start timestamp, during which or after which the message has been saved. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	dateTime	Y	[1..1]
EndTimestamp	The end timestamp, during which or before which the message has been saved. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used. It is possible to use only the combination StartDate and EndDate or the combination StartTimestamp and EndTimestamp.	dateTime	Y	[1..1]
MessageStatus	The request can be filtered by message status. The following status data can be used: • NEW - A list of the response messages that have not yet been downloaded is requested. • DLD - A list of the response messages that have been downloaded is requested. • ALL - A list of all the response messages (whether they have been downloaded or not) is requested.	string	Y	[1..1]
Application	The short name(s) of the Customs application(s) the message data of which is downloaded. If the element is left out, the message data of all Customs applications is requested.	string	N	[0..*]

Table 22: DownloadMessageListFilteringCriteria

8.8 DownloadMessageFilteringCriteria

Element name	Description
DownloadMessageFilte-ringCriteria	This element is needed as key data, when downloading one (response) message from the Customs system.

Element name	Description	Type	Required (Y/N)	Occurrence
MessageStorageId	A unique ID of the message, through which the response message can be retrieved from the Customs notification storage.	string	Y	[1..1]
DocumentID	A unique identifier of the PDF document	string	E	[0..1]

Table 23: DownloadMessageFilteringCriteria

8.9 ApplicationResponseMessage

Element name	Description	Type	Required (Y/N)	Occurrence
ApplicationResponse-Message	Contains a response to e.g. the declaration that has been sent through the upload operation. Contains a response in XML format. XML is base64-encoded.	base-64Binary	Y	[1..1]

Table 24: ApplicationResponseMessage

8.10 EchoContent

Element name	Description
EchoContent	Used for testing purposes: input and output data for the echo operation.

Element name	Description	Type	Required (Y/N)	Occurrence
Text	Freeform text	string	Y	[1..1]
Signature	XML signature. EchoContent is signed	string	N	[0..1]

Table 25: EchoContent

8.11 ApplicationRequest

Element name	Description
ApplicationRequest	XML constructed by the message builder. Contains application-specific data (the actual payload and the parameters associated with it). The XML signed by the message builder.

Element name	Description	Type	Required (Y/N)	Occurrence
MessageBuilder-BusinessId	Country code and Business ID of the XML message builder	string	Y	[1..1]
MessageBuilderSoft-wareInfo	Name of the software and version number used by the message builder.	string	Y	[1..1]
DeclarantBusinessId	Country code and Business ID of the message declarant	string	Y	[1..1]
Timestamp	Timestamp of when the message was built. Data type is ISODateTime. If the time zone has not been defined, UTC is used as default time zone, not Finnish local time.	dateTi-me	Y	[1..1]
Application	The short name of the Customs application to which the actual application message placed in the Content element is directed.	string	Y	[1..1]
Reference	The interchange identifier generated by the customer, with which the customer can bundle together the message sent by the customer and the response messages generated by Customs. The interchange identifier is unique for each application and message declarant. It is formed as follows: the five-letter abbreviated name of the company (created by Customs) combined with a running number. The minimum length of the element is 6 characters and the maximum length 14 characters. The abbreviated name is created from the name of the business operating as a message declarant. Usually the builder of the XML message generates the interchange identifier. Example of the first interchange identifier of a business for which Customs has created the abbreviated name FIRMA: FIRMA000000001	string	Y	[1..1]
Environment	The Customs environment concerned. Possible values: • PRODUCTION • TEST	string	Y	[1..1]
ApplicationContent	Contains the actual payload, which is directed to the Customs application. See	element	Y	[1..1]

	description of the ApplicationContent element.			
Signature	XML signature. The whole ApplicationRequest is to be signed. The signature is used at Customs to identify the message builder and to guarantee the integrity of the data.	Signa-ture	Y	[0-1]

Table 26: ApplicationRequest

8.12 AttachmentRequest

Element name	Description
AttachmentRequest	XML constructed by the message builder. Contains attachment-specific data (the actual attachment file and metadata associated with it). The element is XML signed (enveloped signature) by the message builder. The DVV server certificate granted to the builder of the message is used to sign the message.

Element name	Description	Type	Required (Y/N)	Occurrence
MessageBuilderBusinessId	Country code and Business ID of the XML message builder.	string	Y	[1..1]
DeclarantBusinessId	Country code and Business ID of the message declarant	string	Y	[1..1]
Application	Short name of the Customs application to which the message is connected.	string	Y	[1..1]
AttachmentTitle	Name of the attachment file without the directory path	string	Y	[1..1]
AttachmentDescription	Free-form text related to the attachment file	string	N	[0..1]
AttachmentDocumentType	Attachment file code from Customs' document codes. For example N380 (trade invoice)	string	Y	[1..1]
RelatedMessageStorageId	A declaration file identifier, provided by Customs for the attachment file, can be found in the UploadResponse message of the relevant declaration message.	string	Y	[1..1]
CustomsRequest	Is the attachment file requested by Customs?	boolean	Y	[1..1]
AttachmentRelationIdentifier	The transaction identifier/MRN.	string	Y	[1..1]
LanguageCode	Attachment file language code. Allowed values: fi,sv,en	string	Y	[1..1]
RepresentativeBusinessId	Business Id of the declarant or representative	string	N	[0..1]
AttachmentContent	Contains the actual attachment file. See description of the AttachmentContent element.	Element	Y	[1..1]
Signature	XML signature (Enveloped signature). The whole AttachmentRequestMessage is signed. The signature is used at Customs to identify the message builder and to guarantee the integrity of the data.	Signature	Y	[1..1]

Table 27: AttachmentRequest

8.13 ApplicationContent

Element name	Description
ApplicationContent	The payload with additional data.

Element name	Description	Type	Required (Y/N)	Occurrence
Content	Contains the base64-encoded payload (e.g. AREX or ELEX message).	string	Y	[1..1]
ContentFormat	Data format of the Content element. Possible values are MIME media types. As customers can only transmit XML messages to Customs, 'application/xml' must be set as value. Earlier, the only allowed value was 'XML'. It can still be used. In a response message with a zip attachment, the field value is 'application/zip'.	string	Y	[1..1]

Table 28: ApplicationContent

8.14 AttachmentContent

Element name	Description
AttachmentContent	The payload with additional data.

Element name	Description	Type	Required (Y/N)	Occurrence
Content	Contains a base 64 encoded attached file	string	Y	[1..1]
ContentFormat	Data format of the Content element. Possible values are MIME media types. Accepted values 'application/pdf', 'image/jpeg', 'image/png' and 'image/tiff'.	string	K	[1..1]

Table 29: AttachmentContent

8.15 ApplicationResponse

Element name	Description
ApplicationResponse	The response message by Customs. This is typically a response to the message transmitted through the Upload operation. Contains application-specific data (the actual payload and the parameters associated with it). The XML signed by the message builder.

Element name	Description	Type	Required (Y/N)	Occurrence
DeclarantBusinessId	Country code and Business ID of the message declarant	string	Y	[1..1]

Timestamp	Timestamp of when the message was built. Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used as default time zone.	dateTi-me	Y	[1..1]
Application	The short name of the Customs application, from which the actual payload placed in the ApplicationResponseContent element is directed.	string	Y	[1..1]
ControlReference	Originally the customer's interchange identifier. It is unique for each application and message declarant. With the interchange identifier, the customer can bundle together the message sent by the customer and the response messages generated by Customs. Additional information in the description of the subelement *Reference* of the element *ApplicationRequest*.	string	N	[0..1]
MessageStorageId	Unique ID of the message.	string	Y	[1..1]
ApplicationResponse-Content	Contains the actual response message built by the Customs application. See description of the ApplicationContent element.	Element	N	[0-1]
AttachOf Application-ResponseContent	Attachment. The message from Customs to the customer may contain on ZIP archive as attachment. In such a case, Customs sets 'application/zip' as the value of the element ContentFormat. See the description of the ApplicationResponseContent element.	Element	N	[0..1]
Signature	XML signature. Used only upon specific agreement.	Signatu-re	N	[0..1]

Table 30: ApplicationResponse

8.16 NotifyRequest

Element name	Description
NotifyRequestHeader	This element is the header for each Message Notification Service request sent from the WS interface. This header is built by Customs.

Element name	Description	Type	Required (Y/N)	Occurrence
IntermediaryBusines-sId	Country code and Business ID of the intermediary	string	Y	[1..1]
Timestamp	Time and date of sending the request Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used.	dateTi-me	Y	[1..1]
TransactionId	Unique identifier for the request		Y	[1..1]
Environment	Whether the notification has been sent from the test environment or from the production environment.		Y	[1..1]

	Possible values: • PRODUCTION • TEST
MessageInformation	(see 7.4)

Table 31: NotifyRequest

8.17 NotifyResponse

Element name	Description
NotifyResponseHeader	This element is the header for each Message Notification Service request sent from the WS interface. This header is built by the customer.

Element name	Description	Type	Required (Y/N)	Occurrence
IntermediaryBusines-sId	Country code and Business ID of the intermediary	string	Y	[1..1]
Timestamp	Time and date of sending the request Data type is ISODateTime. If the time zone has not been defined, Finnish local time is used as default time zone.	dateTi-me	Y	[1..1]

Table 32: NotifyResponseHeader

Element name	Description	Type	Required (Y/N)	Occurrence
ResponseCode	Response code indicating the success or failure of the operations followed by the request. The customer has to use the following values: 000 = ok 899 = unexpected error The customer can also use the values listed in Appendix 2 to describe the error more closely.	string	Y	[1..1]
ResponseText	Verbal description of the response code (in English).	string	Y	[1..1]

Table 33: ResponseCode

8.18 DocumentInformation

Element name	Description
DocumentInformation	This optional element is present in the MessageInformation block sent by Customs if the block describes the PDF document in the XML decision. This header is built by Customs.

Element name	Description	Type	Required (Y/N)	Occurrence
DocumentId	A unique identifier of the PDF document	string	K	[1..1]
RelatedMessageStorageId	Unique identifier in the Customs system of the XML document, which was the base for the generated PDF document.	string	K	[1..1]

Table 34: DocumentInformation

9 Authentication and authorisation of technical customer parties

The guide **Introduction to message exchange with Finnish Customs** describes the central roles in direct message exchange: message declarant, builder and intermediary. The roles refer to these parties in a purely technical sense.

The SOAP message contains information about the technical customer parties and their roles in the following XML elements:

Element name	Description	Type	Required (Y/N)	Occurrence
DeclarantBusinessId	Country code and Business ID of the message declarant	string	Y	[1..1]
MessageBuilder-BusinessId	Country code and Business ID of the XML message builder	string	Y	[1..1]
IntermediaryBusines-sId	Country code and Business ID of the intermediary	string	Y	[1..1]

Table 35: Technical customer parties

When a message is processed by Customs, a number of data security checks are performed. Data security checks related to data transfer and business level data security checks have been separated. This makes it possible for the business customers to use a service provider as message builder and intermediary. In such a case, the message declarant has to authorise the service provider to operate in these roles. The authorisation information is gathered through the applications for authorisation for customers of direct message exchange. The service provider can send or retrieve the message declarant's messages only if the service provider has been authorised by the message declarant.

9.1 Authentication of the message declarant

The message declarant is authenticated with the country code and the business ID in the XML element DeclarantBusinessID. Customs' procedure may require that an EORI number is used as the value of DeclarantBusinessId.

The message declarant cannot be authenticated against the server certificate of the XML signature, as the message declarant does not necessarily build the messages.

9.2 Authentication of the builder/the intermediary

For the HTTPS connection and for the XML signature, the builder/the intermediary of the message needs a server certificate. Customs' message interface authenticates the party by comparing the server certificate with the message data. When the connection is being created, Customs also checks that the certificate is not on the certificate revocation list.

9.3 Authentication of the Message Notification Service provider

In the application for authorisation submitted to Customs, the message declarant provides the URL to which Customs is to send the message notifications. Customs will not in any way check who administers the url or comment on it, as long as the service provided there presents a service certificate approved by Customs when the connection is being created. It is entirely at the responsibility of the message declarant that the url provided in the application for authorisation is correct. If the message notifications are received by the same party that builds and transmit the messages, one server certificate is enough.

10 Server certificate

For direct message exchange, the business customer needs a certificate.

For direct message exchange with Customs, the business customer has to acquire a server certificate from a certificate authority approved by Customs. Customs will only accept server certificates by the Digital and Population Data Services Agency (DVV). Business customer's EU VAT identifier have to be stored as "SerialNumber" attribute for "Subject" field.

The server certificate is subject to a charge. Customs does not distribute certificates granted by DVV. The customer acquires the needed certificate directly from DVV.

The server certificate is acquired by the party that builds and transmits messages to Customs' direct message interface. If the message declarant carries out these phases, the server certificate is to be acquired for the message declarant. If a service provider is used for building and transmitting messages on behalf of the message declarant, the server certificate is to be acquired by the service provider.

The same certificate may be used in the test and production environments.

For exceptional cases, see FAQ.

The business customer needs DVV's CA certificates when the customer's software checks the accuracy of the service certificate of Customs.

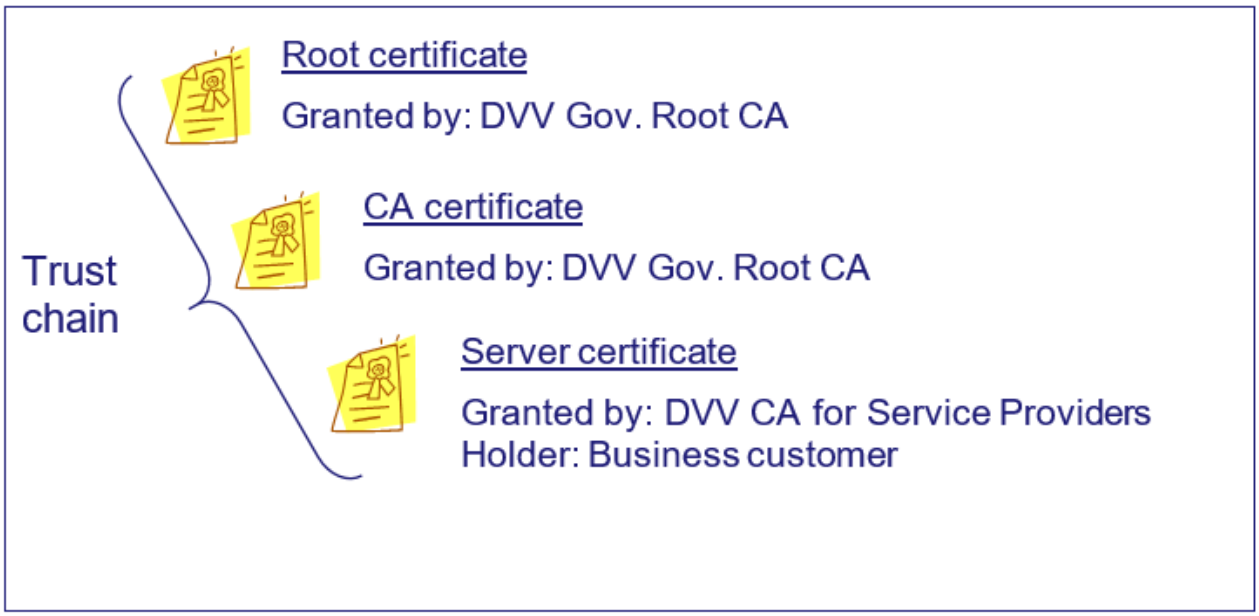

Figure 26: Certificate trust chain

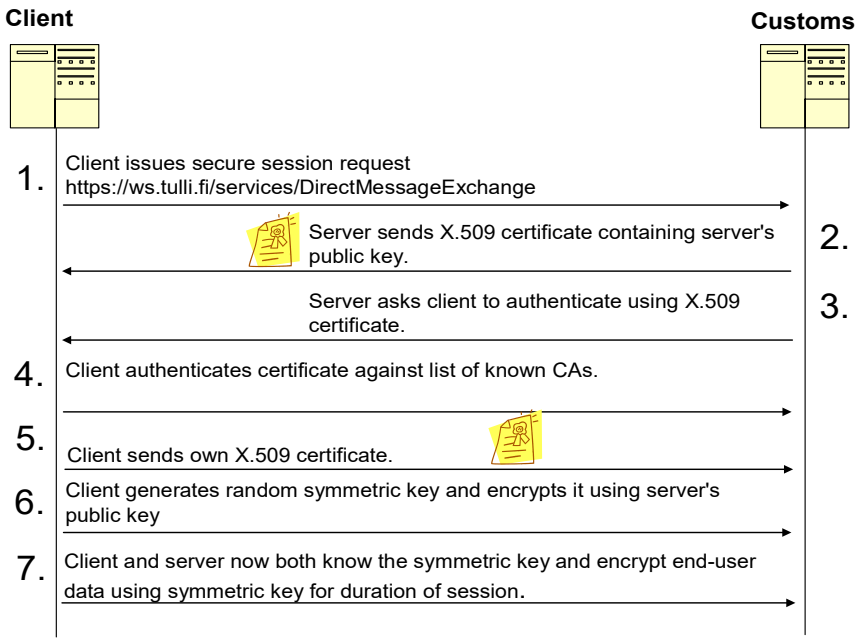

Figure 27: Creating a HTTPS connection using a certificate (simplified).

The trust chain in the previous figure concerns phases 2 and 4.

10.1 Acquisition and implementation of a server certificate

Acquiring a DVV server certificate and its implementation includes three stages:

1. The company must apply via DVV's electronic service (https://asiointi.dvv.fi). The DVV webpage gives instructions that are more precise on necessary measures to be taken.
2. DVV sends the server certificate to the business.
3. The business has to install the **server certificate** on its server, to be used by its own software

10.2 Renewing the server certificate

The server certificate granted by the Digital and Population Data Services Agency (DVV), which is used in direct message exchange, is valid for one year. Messages cannot be sent to Customs using an expired server certificate.

DVV reminds Customs' customers when their certificates are about to expire, but it is the customers who are responsible for renewing their certificates.

Businesses do not need to notify Customs of the renewal of their server certificate.

Customs also has to renew its certificate every two years. Customers will not be notified of this, because it is assumed that they check the certificate of Customs against the list of certification authorities. If the check is performed by comparing it statically with the Customs certificate saved in the customer's system, message exchange may be interrupted when Customs changes its certificate. In this case it is the customer's responsibility to know when the certificate is changed. The validity of Customs' certificates can be checked in DVV's Certificate Directory.

11 Customer message transmission data

11.1 Service provider's identification

In direct message exchange, the message declarant and the service provider, if any, are identified on the basis of the country code FI and the Business ID (for example FI1234567-8) – foreign customers are identified on the basis of the VAT number.

11.2 URL of the service

The SOAP message is sent to the following URLs:

Testing environment:

https://ws-customertest.tulli.fi/services/DirectMessageExchange

Production:

https://ws.tulli.fi/services/DirectMessageExchange

The **calls to** the customer's **Message Notification Service** come from Customs from the address 157.129.127.247. Customs recommends the customers to use a standard https port in their own Message Notification Service, but the use of other ports is also possible.

11.3 Interchange identifier

The interchange identifier begins with the five-letter abbreviated name of the business. The abbreviation is formed by Customs from the name of the business operating as a message declarant. The end of the interchange identifier is chosen by the customer. The maximum length of the element is 14 characters, so the end can consist e.g. of a running number with 9 characters.

The interchange identifier must be unique to the combination of the target system of Customs and the message declarant. This means that the business can use the same interchange identifier once for each target system of Customs (e.g. AREX and ELEX).

The checks carried out by Customs of the messages sent by the customer lead to rejection of the message, if:

- the value of the element Reference in the XML document ApplicationRequest and the interchange identifier in the application message do not match.
- the same interchange identifier has been received from the same target application–message declarant combination before.
- Customs saves the interchange identifier immediately after receiving the Upload request. The interchange identifier will **not** be 'freed' if an error is detected and Customs rejects the Upload request. This also applies when the rejection of the Upload request is due to a temporary technical problem in the Customs system.

The production and customer testing environments of Customs are entirely separate from each other. Customer testing will not 'consume' any production interchange identifiers.

The customer is responsible for seeing to that the interchange identifiers remain unique.

11.4 Transmission data in the customer message

10.4.1 RequestHeader

The message declarant's or the service provider's identification is the content of the XML element 'IntermediaryBusinessID' in the SOAP message RequestHeader.

Example of the RequestHeader XML element in the SOAP message:

```
<cst:RequestHeader
 xmlns:cst="http://tulli.fi/ws/corporateservicetypes/v1">
 <cst:IntermediaryBusinessId>FI2340001-5</cst:IntermediaryBusinessId>
 <cst:Timestamp>2010-03-19T09:15:02.765Z</cst:Timestamp>
 <cst:Language>EN</cst:Language>
 <cst:IntermediarySoftwareInfo>Examples 1.5</cst:IntermediarySoftwareInfo>
</cst:RequestHeader>
```

10.4.2 ApplicationRequest

The SOAP message contains the XML document ApplicationRequest. The XML elements it contains have the following purpose:

- **'DeclarantBusinessID'** contains the identification of the message declarant.
- **'MessageBuilderBusinessID'** contains the identification of the message declarant or, if the declarant uses a service provider, the identification of the service provider.
- **'Application'** contains the short name of the target system of Customs. The names in use are 'AREX', 'ELEX', 'EMCS', 'ALA', 'NCTS', 'ITU', 'CWAR', 'IMP', 'INSTAT' and 'GUARANTEE'.
- The value set for the element **'Environment'**
  - in testing: 'TEST'
  - in production: 'PRODUCTION'.
- **'Reference'** contains the interchange identifier.

Example of the parameters in the XML document 'ApplicationRequest':

```
<req:ApplicationRequest
 xmlns:req="http://tulli.fi/schema/corporateservice/appl/v1">
 <req:MessageBuilderBusinessId>FI2340001-5</req:MessageBuilderBusinessId>
 <req:MessageBuilderSoftwareInfo>Examples 1.5</req:MessageBuilderSoftwareInfo>
 <req:DeclarantBusinessId>FI2340001-5</req:DeclarantBusinessId>
 <req:Timestamp>2010-03-19T11:15:05.234+02:00</req:Timestamp>
 <req:Application>AREX</req:Application>
 <req:Reference>FIRMA000004671</req:Reference>
 <req:Environment>TEST</req:Environment>
```

An application message transmitted as XML data content can also contain the interchange identifier. In this case, the same the interchange identifier as in the application message must be set as the content of the XML element 'Reference' in the XML document ApplicationRequest.

For example, the Message block in the AREX message contains the following XML element:
`<wco:reference>FIRMA000004671</wco:reference>`

ApplicationRequest sisältämän XML-elementin 'Reference' sisältää
`<req:Reference>FIRMA000004671</req:Reference>`

11.4.3 Payload

The XML element Content in the XML document ApplicationRequest contains the XML payload. In the XML payload, the country code FI and the Business ID of Customs, FI0245442-8, are used as the Customs identification in the element **'recipient'**.

The application-specific transmission data should be checked in the message implementing guidelines for each system, available on the Finnish Customs website at

http://tulli.fi/en/e-services/services/message-exchange/message-descriptions

12 XML signature

In direct message exchange between Customs and business customers, XML signatures containing the certificate obtained by the business customer are used for authenticating payloads and for ensuring integrity and non-repudiation.

The application that **builds** the payload signs it using an XML signature.

All customer messages sent to Customs using Upload request messages in direct message exchange are signed using an XML signature.

DownloadList and Download request messages do not contain a customer message or an XML signature. The CheckConnectivity request message used for testing the connection can be signed using an XML signature.

12.1 Structure of the XML signature

The easiest way to find out the structure of the XML signature is to study the example messages in the distribution package "Direct message exchange, examples" on the Finnish Customs website at

http://tulli.fi/en/e-services/services/direct-message-exchange

For more general information about the recommendation relating to the XML signature, see http://www.w3.org/2002/02/xmlsignature-pressrelease

12.2 Implementation of the XML signature

For an XML signature, the customer must have a key store with the certificate granted to the customer by DVV.

An XML signature is always created using program tools. There are a large number of these third-party implementations, such as Java and Microsoft.NET implementations.

12.3 Validation of the XML signature and possible problems

The customer has to validate the XML signature.

Also in connection with the technical ws testing (before the actual customer testing), the business customer can check that the XML signature is working by sending a signed CheckConnectivity request message to Customs.

If the signature does not get validated, it is possible that it will break in the XML processing carried out by the customer's application after signing. That is why it is best to base64 encode the ApplicationRequest immediately after signing.

13 Restrictions on direct message exchange

Direct message exchange imposes restrictions on the customer systems in order to guarantee accessibility of the service. The restrictions are described in this section.

13.1 Protocol versions

The interface supports the following protocol versions:

- SOAP 1.1
- SOAP 1.2 (required if the customer wants to use the Message Notification Service)
- HTTP 1.1
- TLS version 1.2

13.2 Encryption algorithms

The allowed cipher suits for the HTTPS connection, if the TLS connection is used:

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA

The allowed cipher suits for the HTTPS connection, if the SSL connection is used:

- SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA

The allowed algorithm for the SignatureMethod of the XML signature:

- RSAwithSHA256

The allowed algorithm for the DigestMethod of the XML signature:

- SHA256

13.3 Restrictions related to the message

Following restrictions related to the HTTP transfer are imposed on arriving messages:

- Only requests sent via http POST method are allowed
- The overall length of the URL can be max. 4 kilobytes
- The overall length of the header data of the message can be max. 64 kilobytes
- The maximum number of header elements is 128
- The length of the name of the header element can be max. 256 bytes
- The value of the header element can be max. 32 kilobytes
- The length of the URL parameter can be max. 3 kilobytes

Following limits have been set for the structure of the XML message:

- The maximum size of the XML payload (XML element Content of the ApplicationRequest) is 512 kilobytes. The size limit refers to the payload before it is base64 encoded
- The maximum number of nested XML elements is 128
- The maximum number of attributes of the XML element is 64
- The maximum size of the attachment file of an UploadAttachment message is 5 megabytes. The size limit of the attachment file refers to the payload before it is base64 encoded
- The message cannot contain any prohibited character strings that might be interpreted as SQL injection attacks.

13.4 Restrictions related to the number of service requests

The processing of service requests per customer connected to the service is restricted as follows (IntermediaryBusinessID):

- **Upload**: The service processes, at one-second intervals, no more than three Upload requests.
- **UploadAttachment**: The service processes, at one-second intervals, no more than three UploadAttachment requests.
- **DownloadList**: The service processes, at five-minute intervals, no more than one DownloadList request.
- **Download**: The service processes, at one-second intervals, no more than five Download requests.
- **CheckConnectivity**: The service processes, at one-second intervals, no more than one CheckConnectivity request.

When the customer starts to use the Message Notification Service, the customer can send DownloadList service requests once an hour at the most.

Furthermore, the total number of joint message exchange messages for all customers is restricted as follows:

- **Upload**: The service processes no more than 20 customs declarations per second.
- **UploadAttachment**: The service processes no more than 20 attachments per second.
- **DownloadList**: The service processes no more than 10 requests for decision lists ready for retrieval per second.
- **Download**: The service processes no more than 100 clearance decision retrievals per second.

DownloadList and CheckConnectivity requests that exceed the above thresholds are rejected. However, enquiries regarding Upload, UploadAttachment and DownLoad will not be rejected outright, but rather they are “buffered”, when the customer has sent requests that exceed the threshold. In that case, the SOAP request that Customs received through the TCP session, is left queuing for processing. An Upload request or Download request that has been placed in a queue is only rejected if it has not been processed before the https connection timeout.

13.5 Timeouts for service requests

The customers have to set the timeout for the SOAP requests from their software calling Customs' direct message exchange interface to at least 120 seconds. Normally, Customs’ interface returns a response within a couple of seconds, but during certain peak loads the response time may be considerably longer. If the customer software breaks the SOAP connection too quickly, Customs’

interface cannot return a response message to the customer, which means that the response code will not be returned to the customer software. This way the generation of so-called double messages in Customs' systems can also be avoided.

The timeout for Customs' software sending message notifications to customers has also been set to 120 seconds. If the customer's Message Notification Service does not reply before timeout, Customs' systems will not attempt resend, but the attempt to send the message notification will be registered in Customs' systems as an error.

13.6 Availability of decisions by Customs and other documents

Customs stores decision on customs declarations sent by customers and related documents (response messages) - for retrieval by the customer's software - for one year from the time when the customs declaration was received. In time-consuming customs clearances some of the response messages may not be available for a whole year, but at least for three months from the time the response message was generated.

Appendix 1: ResponseCode and ResponseText

Response Code	ResponseText	Description
Message sent successfully		
000	OK	The operation has been completed successfully.
Message sending failed **Authorisation error, please contact the EDI Support of Customs**		
460	Intermediary id not valid	In the SOAP request, the length of value of XML element IntermediaryBusinessId is not 9–17 characters, or the IntermediaryBusinessId and the server certificate used for authenticating the HTTPS client do not refer to the same corporate identity.
461	Intermediary authorization failed	The intermediary is not authorised by any declarant.
465	Declarant authorization failed	The declarant is not authorised to use the target system (value of XML element Application of ApplicationRequest).
466	Builder authorization failed	The message builder is not authorised to use the target system (value of XML element Application of ApplicationRequest), or the message builder is not also the declarant or the intermediary.
467	Intermediary authorization failed	The intermediary is not authorised by any declarant.
Message sending failed **Error in the message, please, correct and resend**		
450	Invalid HTTP connection parameters	A limit imposed on the HTTP connection has been exceeded.

Response Code	ResponseText	Description
451	Schema validation error in SOAP request	A WSDL or schema validation error has occurred while validating the SOAP request message.
452	Schema validation error in ApplicationRequest	A schema validation error has occurred while validating the ApplicationRequest document.
453	Wrong target environment for DownloadRequest	An attempt was made to download a payload destined for the testing environment from the production environment.
455	Rejected by policy	A generic response code for an error that occurred while checking the incoming service request. It is returned when a more specific response code has not been defined.
456	Rejected by filter	A generic response code that is returned when the service request is rejected by filtering functionality.
458	ApplicationRequest with duplicate reference received	The interchange identifier of the ApplicationRequest document (XML element Reference) has been used previously.
459	Encountered character not allowed by XML encoding	The service request contains a character, which is not allowed by the used XML encoding.
463	Builder id not valid	The length of the value of XML element MessageBuilderBusinessId is not 9–17 characters.
464	Declarant id not valid	The length of the value of XML element DeclarantBusinessId is not 9–17 characters.
468	Application request environment not valid	An ApplicationRequest document destined for production has been sent to the test environment, or an ApplicationRequest document destined for the test environment has been sent to production.
469	Content format not XML	The value of XML element ContentFormat in an ApplicationRequest is not ‘application/xml’ or ‘XML’.
470	ApplicationRequestMessage validation failed	XML validation of the ApplicationRequest document failed.
471	Content validation failed	XML validation of the payload (i.e. the value of XML element Content in the ApplicationRequest document) failed.
472	Invalid Application specified	The XML element Application of the DownloadList request contains an application identifier that is not among the allowed application identifiers.

Response Code	ResponseText	Description
473	Content exceeds size limit for application	The payload (i.e. the value of XML element Content in the ApplicationRequest document) exceeds the size limit for the target application.
476	XML signature not valid	The XML signature is not valid. For example, the error code is returned when the digest of the XML signature and the digest computed during validation of the signature differ.
477	SignatureMethod algorithm in signature not allowed	The XML signature uses a signature algorithm, which is not among the permitted signature algorithms.
478	DigestMethod algorithm in signature not allowed	The XML signature uses a digest algorithm, which is not among the permitted digest algorithms.
479	Reference URI in signature invalid	The XML signature of the ApplicationRequest document contains a Reference element, the value of which is not empty (""). The Reference value must always be empty in an enveloped XML signature.
480	SOAP request exceeds size limit	The SOAP request exceeds the allowed maximum size.
482	Invalid RelatedMessageStorageId in AttachmentRequest	The message identifier has the wrong format.
500	ApplicationRequest with duplicate reference received	The message builder must submit only one ApplicationRequest document with any specific interchange identifier (value of XML element Reference). Any further ApplicationRequest documents that contain the same value of XML element Reference are rejected as duplicates.
501	Reference values in ApplicationRequest and Content do not match.	The value of XML element Reference in the ApplicationRequest document does not match the interchange identifier in the payload (Content).
502	DeclarantBusinessId in ApplicationRequest and sender in content do not match.	The value of XML element DeclarantBusinessId in the ApplicationRequest document does not match the identification of the sender in the payload (Content).
503	Referenced declaration not found	A declaration corresponding to the value provided in the message for the RelatedMessageStorageId element cannot be found.
504	Identical attachment for the referenced declaration already exists	An attachment file for the declaration (RelatedMessageStorageId) referred to in the message has already been sent.

Response Code	ResponseText	Description
505	Referenced declaration not accepted	The processing of the original declaration referred to in the message (RelatedMessageStorageId) has been rejected in Customs' systems
506	Referenced declaration and function do not match	The original declaration referred to in the message (RelatedMessageStorageId) and the application identifier (Application), do not match.
600	Start time too far away in the past.	The customer is trying to send a DownloadList request for messages that are too old.
601	Start time greater than end time	The start time of the search criteria is after the end time.
700	Invalid request	The message cannot be found, or the customer is not authorized to download the message.
Message sending failed: Communication or system failure, please resend the message after a few minutes		
457	Allowed message frequency exceeded	The limit imposed on the frequency of service requests has been exceeded.
474	Uploads to application temporarily disabled	No new messages are accepted for delivery to the target application (i.e. the value of XML element Application in the ApplicationRequest document) for the time being. Send a new Upload request message later.
490	Backend connection error	No connection could be established from the Direct Message Exchange frontend service to the backend service.
491	Backend connection error	No connection could be established from the Direct Message Exchange frontend service to the backend service.
492	Backend connection error	Connection error in Customs direct message exchange integration layer
499	Unknown Error	The Direct Message Exchange frontend service has encountered an error, for which no specific error code has been defined.
999	Unexpected Error	The Direct Message Exchange frontend service has encountered an error, for which no specific error code has been defined.
999	Backend technical error	The Direct Message Exchange frontend service has encountered an error, for which no specific error code has been defined.

Table 36: ResponseCode and ResponseText

Appendix 2: Error codes used in the NotifyResponse message

Response Code	ResponseText	Description
Successful message transmission		
000	OK	Successful message notification
Message transmission that resulted in an error		
899	Unexpected Error	Error situation for which there is no specific error code
Optional message errors		
851	Schema validation error in SOAP request	Schema error in the notification message
860	Intermediary id not valid	Intermediary ID is not valid
864	Declarant id not valid	Declarant is not an authoriser of the intermediary
868	Environment not valid	The declaration message refers to a wrong environment
872	Invalid application specified	Application unidentified or not in use
841	Control reference not valid	Unidentified reference ID
842	MessageStorageId not valid	Unidentified message ID

Table 37: Error codes used in the NotifyResponse message

Appendix 3: Requirements for the XML messages of Customs

General requirements for XML messages

The XML message requirements presented here concern the payloads as well as the SOAP request and response messages used for transporting the payloads.

In the XML messages of Customs, version 1.0 of the XML standard is used. In the messages, the Unicode characters and the character encoding UTF-8 are used. These features should be given in the XML prolog occurring before the root element of the message. Below is an example of a prolog. The XML version of the prolog used in the XML process instructions is expressed using the version attribute and the character encoding of the message using the encoding attribute.

<?xml version="1.0" encoding="UTF-8"?>

If the prolog has a standalone attribute that defines the existence of an externally defined DTD (Document Type Definition), the value of the standalone attribute is to be “no”. The attribute is not necessary, as the default status of the feature (standalone = “no”) complies with the Customs practice of checking the accuracy of the messages in terms of the external DTD. The prolog should not contain any other processing instructions than the ones mentioned above, i.e. the XML processing instructions that include the XML version and the character encoding (possibly including the standalone attribute=“no”).

If a Byte-Order-Mark (or BOM) is entered in the beginning of the prolog of the XML message, it has to have the same value as the above-mentioned encoding attribute. The UTF-8 encoding is expressed with the BOM marker. Its hexadecimal value for the three primary bytes is EF BB BF. When creating test messages as text files, it should be noted that several of the XML and word processing tools add the BOM marker at the beginning of files, even though the tools do not display the marker. The settings of the tool used should be checked to ensure that the potential BOM marker to be inserted in a file is UTF-8.

In the messages, elements with no data should be left out. In the export system, for example, elements with no data cause a rejection.

In the messages, CDATA sections of the XML should not be used in the values of the elements. They are not even needed in the messages of Customs’ applications.

The structure of the XML messages of Customs is described using XML schemas. The XML schemas are released in order make it easier to process messages in the XML format. The customer has to make sure that the message contains correctly formed data. The correctness of the XML message can be checked by validating it against the message schema.

The schemas of the XML messages specified by Customs have implemented **namespaces** and a requirement of using explicit names for elements (elementFormDefault="qualified"). In the messages of these types of schemas, only elements that belong to the namespaces are used, and the tag names used for marking the elements must be added to the namespace of the element. The fact that the elements are derived from specific namespaces ensures that each element can be explicitly identified without name conflicts.

For more information about the use of namespaces in XML messages, see Appendix 4.

Requirements related to application messages

Restrictions regarding the format of each field as well as allowable values in application messages are described in the message implementing guidelines. The general requirements that have been set for message data are presented here as well as some application-specific general changes.

The Unicode character set cannot be used in its entirety for presenting data to be processed in the Customs applications. The Unicode standard defines character subsets based on language areas. The usual subset characters of ‘BasicLatin’ (hexadecimal code values x00 – x7F) and ‘Latin-I Supplement’ (hexadecimal code values x80 – xFF) can generally be used in all applications, except in the special cases and exceptions described below. The subset BasicLatin is in fact the old ASCII character set and as such, it includes numerous control characters that should not be used in XML-messages. The detailed content of the Unicode subsets is available at http://www.unicode.org/charts/index.html.

The corresponding subsets are defined in the appendix to the W3C XML-schema-standard in accordance with the Unicode subsets. The appendix is available at http://www.w3.org/TR/xmlschema-2/#charcter-classes. A table of language area character subsets can be found under the heading 'F.1 Character Classes', section 'Block Escape'. The control characters are also defined in the previously mentioned link in the table under 'Category Escape'. The XML-character set category mentioned there; 'Cc' for control characters, comprises the control characters of the Unicode blocks 'C0' and'C1', but some characters have been left out: SP (space) NBSP (Non-Breaking Space) and SHY (Soft Hyphen).

The only control characters in the Unicode BasicLatin subset, i.e. the so-called ’C0’-area characters (Hexadecimal code values x00 – xIF) that can be used are the form characters HT (Horizontal Tab), LF (Line Feed), VT (Vertical Tab), FF (Form Feed) and CR (Carriage Return). The characters VT and FF should be avoided, because they do not serve any purpose in XML. The special character DEL (Delete, hiding the previous character), which is much like the control characters, is also included in the subset. Even though it is not included in the actual ’C0’ group of control characters, it does not serve any purpose in XML either, and should not be used.

The only control character in the Unicode BasicLatin subset (i.e. the so-called ’CI’-area characters with Hexadecimal values x80 – x9F) that can be used is the formatting character NEL, but it does not serve any purpose in XML and should also be avoided. The special characters NBSP and SHY, which are much like the control characters, are also included in the subset. Even though they are not included in the actual control character group ’C1’ they should not be used in XML since they are also void of meaning.

The permissible control characters mentioned earlier, the special characters, which are much like them and the space-character SP (x20) all constitute a group pf formatting characters frequently used in a text. They are used sparingly in XML text contents, as described below.

HT ’Horizontal Tabulation’ (Hexadecimal code value x09) is a horizontal tabulator that often is used for indentation in XML messages.

LF ’Line Feed’ (Hexadecimal code value x0A) is a character mainly used for changing lines in XML messages.

VT ’Vertical Tabulation’ (Hexadecimal code value x0B) is a vertical tabulator which serves no purpose in XML messages and should not be used.

FF ’Form Feed’ (Hexadecimal code value 0C) is a page-break which serves no purpose in XML messages and should not be used.

CR ’Carriage Return’ (Hexadecimal code value x0D) resets the position to the beginning of a text and is used in XML messages as one of the line changing characters.

SP ’Space’ (Hexadecimal code value x20) is an empty space character which is also used in XML messages for line indentation.

DEL ’Delete’ (Hexadecimal code value x7F) has been used for hiding previously written characters. It serves no purpose in XML messages and should not be used.

NEL ’Next Line’ (Hexadecimal code value x85) is a line change which originates from the EBCDIC character set. It serves no purpose in XML messages and should not be used.

NBSP ’Non-Breaking Space’ (Hexadecimal code value xA0) prevents an automatic line break in HTML-contents. It serves no purpose in XML messages and should not be used.

SHY ’Soft Hyphen’ (Hexadecimal code value xAD) is a page-break, which serves no purpose in XML messages and should not be used.

Some of the formatting characters, i.e. HT, LF, CR and SP, form a ‘White Space’ group in XML. This is the character set-‘s’ for which XML defines special rules of handling (http://www.w3.org/TR/2008/REC-xml-20081126/#sec-white-space). In addition to formatting the XML structure, they can also be used in the XML text content, but of the normalising values they are compensated with a single space character in the middle of the text and disappear from the beginning and the end of the values. The characters LF and CR and a combination of them are used for line changing in XML, but when handled in XML all line changes convert into LF characters.

XML schemas for application messages

The data content in XML messages always appear as text in character format and is referred to as the text content of a message. The text interpretation of the value of structure components – elements and attributes in XML messages, is governed by datatypes used in the XML schemas. The values are interpreted numerically or in a comparable way, when the structure component is based on primitive datatypes that describe the numeric value, truth value (Boolean), time point or a time interval. Empty values for these values are naturally not allowable, since there is no meaningful interpretation for them. Empty values can by nature be meaningful for datatypes interpreted as text, but they should be used advisedly. When a text value of a structure component is empty, it should not be considered as missing, rather as empty. Interpreting data as missing only applies when the structure component is missing from the XML structure.

There are often primitive datatypes (date, dateTime and time) that are used as a basis for **datatypes that refer to time** in the applications for XML schemas regarding direct message exchange. The formats have been restricted, though, so that fractions of seconds (milliseconds) or parts expressing the time zone are not permitted.

For datatypes that are like texts, only a limited number out of the vast amount of characters of the Unicode character set are used in XML schemas of the applications for direct message exchange. Limiting the use of control characters and special characters aims to control their usage in messages in the way described earlier. There are some differences in the schemas of different applications. These differences are clarified in the following system-specific descriptions.

Requirements for the import system (ITU) and the transiting system (NCTS)

The datatype **BaseTextType** has been created in the schemas of the interfaces as a basis for all data in text format. The datatype only allows the Unicode characters that are part of the subsets '**BasicLatin**' or '**Latin-1Supplement**'. This BaseTextType is only a basis from which the actual datatypes for text data occurring in messages have been derived, by restricting the choice of characters even further. This has been done by deriving four datatypes for data of different nature: TextType, LineType, CodeType and IdentifierType. These are described further below. They all have in common that the datatype is based on BaseTextType, but it prohibits the use of control characters that do not serve the right purpose.

TextType is used for data that contains free text. An exception is the limitations in the XML control character category '**Cc**'. It does, however, allow the character category '**s**', i.e. '**White Space**' characters.

LineType is used for text data on one line**, CodeType** is used for code abbreviations and **IdentifierType** for identification data. An exception is the limitations in the XML control character category '**Cc**'. It does, however, allow the character category '**t**', which includes the character **HT** (horizontal tabulator).

Requirements related to messages in the Intrastat system

The datatype **BaseTextType** has been created in the schemas of the interfaces as a basis for all data in text format. The datatype only allows the Unicode characters that are part of the subsets '**BasicLatin**' or '**Latin-1Supplement**' as well as **the euro symbol**. This BaseTextType is only a basis from which the actual datatypes for text data occurring in messages have been derived by restricting the choice of characters even further. This has been done by deriving four datatypes for data of different nature: TextType, LineType, CodeType and IdentifierType. These are described further below. They all have in common that the datatype is based on BaseTextType, but it prohibits the use of control characters that do not serve the right purpose.

TextType is used for data that contains free text. An exception is the limitations in the XML control character category '**Cc**'. It does, however, allow the character category '**s**', i.e. '**White Space**' characters.

LineType is used for text data on one line, **CodeType** is used for code abbreviations and **IdentifierType** for identification data. An exception is the limitations in the XML control character category '**Cc**'. It does, however, allow the character category '**t**', which includes the character **HT** (horizontal tabulator).

Appendix 4: XML messages and the use of namespaces

For the sake of simplicity, this appendix describes only the most typical ways of using namespaces in XML messages. It does not include any exceptions or more complex ways of using namespaces.

XML schemas

The XML message structure consists of elements and of attributes that may be attached to the elements. The schema of the message defines the elements used in the structure, their location in the message and determines what type of element content is allowable. The structures of specific messages are typically defined in the schema of the message structure itself, but commonly used structure definitions are also imported as ready-made components from outside the message schema from so-called component schemas.

When using namespaces, all the namespaces used in the schema should be declared in it. The structures defined in the schema itself belong to the message schema's targetNamespace. This also applies to the once that are based on imported datatypes. The child elements in imported structures, though, belong to the component schema of the imported namespace.

XML messages

The standard practice is to use namespaces for message elements but not for attributes. When the main schema of a message requires the use of namespaces, the message should at least include a declaration of the target namespace defined in the schema. The namespaces of component schemas need to be declared only if the message includes structures that belong to them.

In addition, the namespace "http://www.w3.org/2001/XMLSchema-instance", which contains general specifications for structures that refer to an XML schema, may need to be declared in the XML message. The namespace is usually declared by using the abbreviation xsi as follows:

xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance

This namespace needs to be declared in the message when attributes belonging to it are used in the message (http://www.w3.org/TR/xmlschema-1/#Instance_Document_Constructions).

Structures of the namespace can occur in XML messages "http://www.w3.org/XML/1998/namespace" and the namespace is always referred to with the abbreviations xml. This namespace should not be declared as an exception, though, because it is always used automatically with XML parsers.

Declaring namespaces in XML messages

The namespaces are usually declared in the start tag of the root element of the message. The namespaces may, however, be declared in elements located anywhere in the message, in which case they only influence that specific element. There can be no elements without a namespace when the schema requires that namespaces be used. When declared, the namespaces are given an abbreviation, except for any default namespace used. Below is an example of how to declare the namespaces at the beginning of the FISummaryDeclaration message in AREX.

```
<?xml version="1.0" encoding="UTF-8"?>
<decl:FISummaryDeclaration xmlns:decl="http://tulli.fi/schema/arex/declaration/v3"
 xmlns:wco="http://tulli.fi/schema/common/wco/v12_0">
```

Below is a corresponding example of how to declare the namespaces of the elements occurring in the FIEntryExitResponse message in AREX.

```
<?xml version="1.0" encoding="UTF-8"?>
<resp:FIEntryExitResponse xmlns:arex="resp://tulli.fi/schema/arex/response/v3"
 xmlns:wco="http://tulli.fi/schema/common/wco/v12_0">
```

It should be noted that the abbreviation given to a namespace when declaring it can be chosen freely. The chosen abbreviation is only shown within an occurrence of that particular structure, typically within a particular message. The abbreviations "decl" and "resp" used in the examples above can just as well be replaced with other abbreviations. The abbreviations for the namespaces used in a message are very often formed automatically by using e.g. running numbers, as in the example below.

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:FIEntryExitResponse xmlns:ns1="http://tulli.fi/schema/arex/response/v3"
 xmlns:ns2="http://tulli.fi/schema/common/wco/v12_0">
```

Attaching an element to a namespace in XML messages

When the schema requires that namespaces be used for elements, each of the elements in the messages belong to a namespace, since elements without namespaces are not allowed. The namespace of an element in an XML message is expressed by attaching the abbreviation of the namespace to which the element belongs as a prefix to the element name. Each namespace has been given its own abbreviation when declaring the namespaces used in the message. The prefix and the element are separated by a colon. The prefix to the name is given both in the start tag and the end tag of the element. Below is an example of some elements in the FISummaryDeclaration message where the abbreviations of the above example have been used.

```
 <decl:Agent>
 <wco:Party>
 <wco:identity>FI5342687-3</wco:identity>
 <wco:identityExtension>T0001</wco:identityExtension>
 <wco:name1>Huolintatesti Oy</wco:name1>
 <wco:Address>
 <wco:line>Tullihallitus 2</wco:line>
 <wco:city>Helsinki</wco:city>
 <wco:postCode>00110</wco:postCode>
 <wco:country>FI</wco:country>
 </wco:Address>
 </wco:Party>
 </decl:Agent>
```

The abbreviation chosen to represent the namespace is always determined for each message and is only valid within that message. Consequently, there can be variations between the abbreviations and prefixes used in separate messages whose message structures are defined in a specific schema, even though the abbreviations refer to the same namespace. What is crucial for ensuring the correctness of the messages is that the elements have been attached to the correct namespaces, not which abbreviations have been used to refer to them.

Default namespace as a special case of the above

There is no need to include a default namespace in a message, but if desired, one of the namespaces used in the message can appear as a default namespace. The namespace, which is not given an abbreviation when it is declared, becomes the default namespace.

If no abbreviation has been defined for a namespace, one cannot use an abbreviation as a prefix for the elements that belong to it. After declaring a default namespace all elements without name prefixes are interpreted as belonging to that default namespace.

Only one namespace at a time can be the default namespace, so if several namespaces are in use, the rest of them should be given an abbreviation when declared. This abbreviation must be used as a prefix to the element names. Below is an example of how to use a default namespace in the FISummaryDeclaration message.

```
<?xml version="1.0" encoding="UTF-8"?>
<FISummaryDeclaration xmlns="http://tulli.fi/schema/arex/declaration/v3"
 xmlns:wco="http://tulli.fi/schema/common/wco/v12_0">
 <Message>
 <wco:function>FI547A</wco:function>
 <wco:sender>003753426873</wco:sender>
 <wco:recipient>003702454428</wco:recipient>
 <wco:issue>2010-04-13T09:37:16</wco:issue>
 <wco:reference>TESTI1000004-01</wco:reference>
 <wco:test>1</wco:test>
 </Message>
 <Declaration>
 <Document>
 <wco:reference>Testi 3 meri</wco:reference>
```

```
<wco:issue>2010-04-13T09:37:16</wco:issue>
</Document>
<Agent>
<wco:Party>
<wco:identity>FI5342687-3</wco:identity>
<wco:identityExtension>T0001</wco:identityExtension>
<wco:name1>Huolintatesti Oy</wco:name1>
<wco:Address>
<wco:line>Tullihallitus 2</wco:line>
<wco:city>Helsinki</wco:city>
<wco:postCode>00110</wco:postCode>
<wco:country>FI</wco:country>
</wco:Address>
</wco:Party>
</Agent>
```

A default namespace may also be declared in an element located anywhere in the message, in which case it is effective only within that specific element.

Appendix 5: The technical standards of direct message exchange

Standard	Version
WS-I Basic Profile	1.1
SOAP	1.1 and 1.2
WSDL	1.1
HTTP	1.1
SSL	3.0
TLS	1.0, 1.1 and 1.2
XMLDSIG	1.0 http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/
SHA256 DigestMethod	http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/
RSA-SHA256 SignatureMethod	http://www.ietf.org/rfc/rfc4051.txt

Table 38: The technical standards of direct message exchange