

Message exchange with Finnish Customs: Technical guidebook

Contents:

Introduction	5
1 Message exchange via an operator.....	6
1.1 Introduction	6
1.2 Opening the connection	6
1.3 Transmission data.....	6
1.4 PDF files	6
1.5 Technical message sender.....	7
2 Direct message exchange	8
2.1 Introduction	8
2.2 Server certificate	8
2.2.1 Server certificate application.....	9
2.2.2 Certificate request	10
2.3 Opening the connection	11
2.4 Transmission data.....	12
2.4.1 URL address of the service.....	12
2.4.2 RequestHeader	12
2.4.3 ApplicationRequest	12
2.4.4 Application message	13
2.4.5 Interchange identifier.....	14
2.5 PDF files	14
3 The direct message exchange web service	16
3.1 SOAP message and its structure.....	16
3.2 Operations provided by the service.....	17
3.3 The phases of building and transmitting the message.....	17
3.4 Technical customer parties and roles	18
3.5 Authentication and authorisation	19
3.5.1 Authentication of the intermediary.....	19
3.5.2 Authentication of the builder	20
3.5.3 Authentication of the message declarant.....	20
3.5.4 Authorisation of the message declarant and the service provider	20
3.6 Restrictions on direct message exchange.....	20
3.6.1 Protocol versions	20
3.6.2 Encryption algorithms	21
3.6.3 Restrictions related to the message.....	21
3.6.4 Restrictions related to the number of service requests	21
3.7 Response in normal and error situations.....	22
3.7.1 ResponseHeader codes.....	22

3.7.2	SOAP fault	25
4	XML structures in direct message exchange.....	26
4.1	Requirements for XML messages	26
4.1.1	General requirements for XML messages.....	26
4.1.2	XML schema	26
4.1.3	Declaring namespaces in XML messages	27
4.1.4	Attaching an element to a namespace in XML messages.....	27
4.1.5	Default namespace as a special case of the above	28
4.2	WDSL and XSD files	29
4.3	Operations provided by the service.....	29
4.3.1	Upload	29
4.3.2	DownloadList	30
4.3.3	Download	31
4.3.4	CheckConnectivity	31
4.4	Operation-specific XML elements	33
4.4.1	Upload, request.....	33
4.4.2	Upload, response.....	34
4.4.3	DownloadList, request.....	35
4.4.4	DownloadList, response.....	36
4.4.5	Download, request.....	37
4.4.6	Download, response.....	38
4.5	Descriptions of the data of the XML elements	40
4.5.1	Descriptions of the data of the XML elements used by the intermediary	40
4.5.2	Descriptions of the data of the XML elements used by the builder	44
4.6	Example message of direct message exchange.....	47
4.6.1	Application message	47
4.6.2	ApplicationRequest document.....	47
4.6.3	Signed ApplicationRequest document	48
4.6.4	The final SOAP message (UploadRequest)	48
Appendix 1.	Technical standards utilised in direct message exchange.....	50
Appendix 2.	Further information	51

Tables:

Table 2:	Mandatory fields in the certificate request	10
Table 3:	Serial Number field in the certificate request.....	11
Table 4:	ResponseCode and ResponseText.....	25
Table 5:	Upload	30
Table 6:	DownloadList	30

Table 7: Download	31
Table 8: CheckConnectivity	32
Table 9: RequestHeader	40
Table 10: ApplicationRequestMessage	40
Table 11: ResponseHeader	41
Table 12: MessageInformation	42
Table 13: DownloadMessageListFilteringCriteria	43
Table 14: DownloadMessageFilteringCriteria	43
Table 15: ApplicationResponseMessage	43
Table 16: EchoContent	43
Table 17: ApplicationRequest	45
Table 18: ApplicationContent	45
Table 19: ApplicationResponse	46
Table 20: Technical standards utilised in direct message exchange	50
Table 21: Further information	51

Figures:

Figure 1: Acquiring a certificate	9
Figure 2: SOAP request message structure	16
Figure 3: SOAP response message structure	17
Figure 4: Building the message and transmitting the message to Customs	18
Figure 5: Upload request	33
Figure 6: Description of ApplicationRequest	33
Figure 7: Upload response	34
Figure 8: DownloadList request, using elements StartTimestamp and EndTimestamp	35
Figure 9: DownloadList request, using elements StartDate and EndDate	35
Figure 10: DownloadList response	36
Figure 11: Download request	37
Figure 12: Download response	38
Figure 13: Description of ApplicationResponse	39

Introduction

This guidebook deals with the technical matters of the message exchange with Finnish Customs at a detailed level. The guide contains both instructions for message exchange via an operator and instructions for direct message exchange.

In the guidebook, matters concerning direct message exchange include directions on acquiring a server certificate and a description of the direct message exchange web service.

As a basis for decision making that concerns choosing a service channel, we refer to the guide *Introduction to message exchange with Finnish Customs*.

1 Message exchange via an operator

1.1 Introduction

The Customs systems address the response message to the customer company and route the message to the customer's EDI operator. The EDI operator transmits the message to the customer of Customs.

X.400 and ftp/VPN data transfer protocols are used for data transfer between the EDI operator and Customs. It is not required to inform Customs of the data transfer mode used between the company and the EDI operator.

1.2 Opening the connection

In message exchange via an operator, the company and Customs have no direct communications connections. The message exchange is carried out via an operator. The company submits the name and contact information of the operator in its EDI sender application form.

The Customs testing official communicates the contact information to the operator of Customs. At the same time, the testing official gives the date when the connection can be opened. The Customs operator informs the company's operator of this in advance. The company's operator creates the connection to the Customs testing address.

1.3 Transmission data

In message exchange via an operator, the sender's EDI code (country code 0037 + Finnish Business ID without punctuation marks) is used in accordance with the ISO standard 6523 to identify the EDI sender.

For the AREX and ELEX systems, the standard Customs EDI identification number 003702454428 will be used as the code for Customs *both during testing and in production*.

When testing against the AREX and ELEX systems, in the Message block of the application message, "1" is entered as the value of the test indicator (XML element 'test').

In production, "0" must be entered as the value of the test indicator (XML element 'test').

For other systems (e.g. ITU and NCTS), the suffix '-TESTI' is appended to the Customs EDI identification number 003702454428 during testing.

1.4 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. If the PDF contains a decision made by Customs, it will include appeal instructions, which are not included in the corresponding EDI messages.

In message exchange via an operator, the files will be MIME encoded for the duration of the transfer, in order to manage the transfer of the PDF files. The process which the material is related to is identified by the subject of the message or the name of the file. In X.400 message exchange, the PDF file is transmitted as an attachment file of the X.400 message.

1.5 Technical message sender

In message exchange via an operator, the EDI customer may use a technical message sender for sending messages. The technical message sender converts the declaration message of the EDI customer into XML/EDIFACT format and sends it to Customs. The technical EDI message sender may not change any data in the messages. For Customs, receiving data from the technical sender via messages means that when making trader checks, the application used by Customs accepts as the message sender the trader that is not the trader in the declaration section of the message.

The technical message sender goes through a technical testing carried out in order to ensure functional data transfer and EDI messages aligned to the Message Implementing Guidelines. For the time being, using a technical message sender is only possible when sending EDI messages related to import, transit and summary declarations.

2 Direct message exchange

2.1 Introduction

This guide covers technical details which are required for integrating a customer system to Customs' service. Direct message exchange operations and roles are introduced in the guide *Introduction to message exchange with Finnish Customs*. It is recommended to use both guides in parallel.

2.2 Server certificate

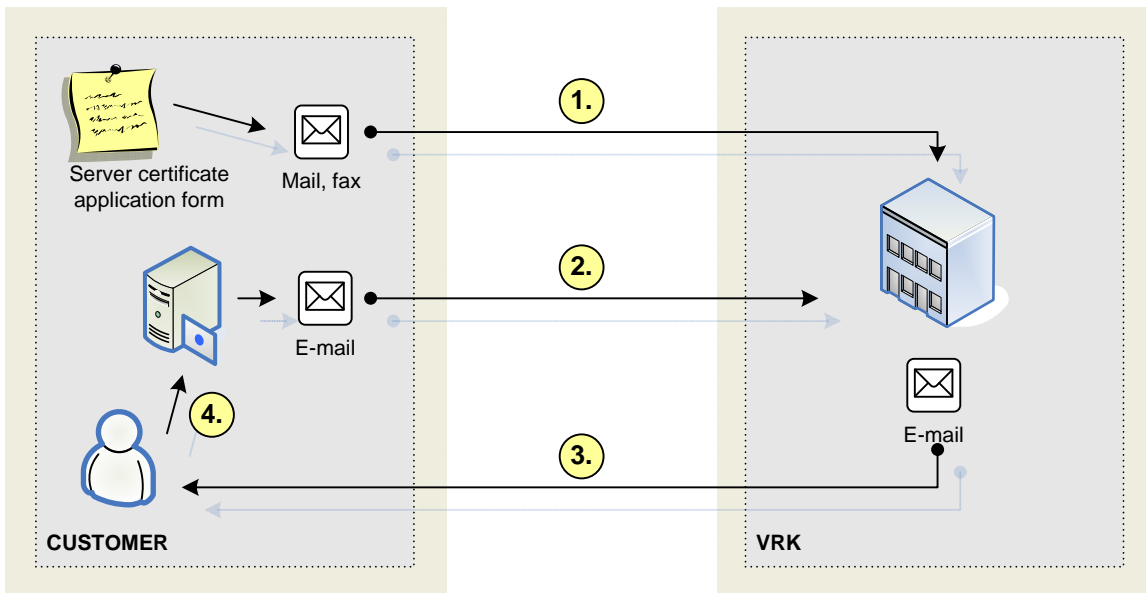
For direct message exchange, the customer must obtain a server certificate from a certificate authority accepted by Customs. Customs will only accept server certificates by the Finnish Population Register Centre (VRK).

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

Acquiring the certificate from VRK and taking it into use includes four phases:

1. For obtaining the server certificate, the company must fill in **the server certificate application form** (administrative application form) and deliver it to the registry office of VRK.
2. In addition, a technical **certificate request** (a file) is to be created on the company's server and delivered to the certificate production of VRK.
3. VRK will deliver the server certificate to the company.
4. The company needs to install **the server certificate** on its server, to be used within its software.

These four phases are illustrated in the figure Figure 1: Acquiring a certificate.



ACQUIRING A CERTIFICATE:

1. Fill in and send the server certificate application form to VRK <http://www.fineid.fi/default.aspx?id=587>
2. Create the certificate request file on server and send it to VRK
3. VRK sends back the server certificate
4. Install the VRK delivered server certificate on the server

Figure 1: Acquiring a certificate

The following sections contain instructions are for the server certificate application form, the certificate request and installation of the server certificate. For further information, the customer can send e-mail to: **varmennemynti@vrk.fi** (certificate sales).

2.2.1 Server certificate application

For obtaining the certificate, the VRK server certificate application form must be filled in. The application is available in PDF format on the VRK website for technical information at http://www.fineid.fi/vrk/fineid/home.nsf/pages/index_eng, under "Acquiring a Certificate" in the document Application form for a server certificate (pdf).

Detailed instructions for completing the application form can be found on the VRK website for technical information at http://www.fineid.fi/vrk/fineid/home.nsf/pages/index_eng, under "Acquiring a Certificate" in the document Instructions for acquisition of a server certificate (pdf).

The application form is to be filled in according to the instructions given by VRK and the specific instructions given by Finnish Customs.

2.2.1.1 Purpose of use of the service

The customer of Finnish Customs fills in here "Customs direct message exchange" as the purpose of use.

2.2.1.2 URL address or name of server

The name of the customer's server that is to be saved in the server certificate.

Finnish Customs does not have specific demands regarding the name that is chosen, as the direct message exchange of Customs does not utilize this information. (The web service for direct message exchange checks other data within the certificate.)

A customer of Customs can enter any server name as the content of the field, as long as it ends with the DNS domain name of the customer company. The chosen server name does not need to be found in the DNS service and does not need to be utilized in the direct message exchange.

The name entered in the server certificate application form must be the same as the Common Name (CN) entered in certificate request file.

2.2.1.3 Validity

The server certificates for customers of Finnish Customs direct message exchange are always applied for a two-year period of validity.

2.2.1.4 Purpose of use of the certificate

For the direct message exchange with Customs, both "client authentication" and "server authentication" must be selected as the server role.

2.2.1.5 Delivery method and address of server certificate application form

The application form that has been filled in and signed will be delivered, with appendices, to the registry office of the Population Register Centre. Accurate address details have been given in the instructions for acquisition of a server certificate.

2.2.2 Certificate request

The certificate request is a file in a prescribed format that is created with the software used by the customer in the direct message exchange.

The instructions for acquisition include information on the certificate request, under "Creation of a certificate request file."

The instructions for acquisition are available on the VRK website for technical information at http://www.fineid.fi/vrk/fineid/home.nsf/pages/index_eng, under "Acquiring a Certificate" in the document "Instructions for acquisition of a server certificate (pdf)."

2.2.2.1 Content of the certificate request file

When creating the certificate request on the server, the following specific requirements of Customs must be observed:

- The length of the Public Key for the customer's server certificate must be **2048 bytes** in order to be used in the direct message exchange with Customs.
- The following fields must be used when creating a certificate request:

Field	Data content	Example:
(CN) Common Name	Name of server (FQDN)	server.company.com
(O)Organization	Name of organisation	Company Ltd
(C) Country	Country code	FI

Table 1: Mandatory fields in the certificate request

A customer of Customs can enter any server name as the content of the field Common Name (CN), as long as it ends with the DNS domain name of the customer company. The chosen server name does not need to be found in the DNS service and does not need to be utilized in the direct message exchange.

The server name entered in the server certificate application form must be the same as the Common Name (CN) entered in certificate request file.

In addition to the above mandatory fields, the optional Serial Number field can be entered in the certificate request.

Field	Data content	Example:
(SERIALNUMBER) Serial Number	VAT number	FI12345678

Table 2: Serial Number field in the certificate request

The field contains the customer's VAT number. The VAT number of a company registered in Finland is obtained from the business ID so that the country code FI is added in front of the ID and the dash between the last two digits is removed.

It is not necessary to include the field Serial Number in the certificate request. VRK adds the customer's VAT number in the server certificate without separate request, on the basis of the information in the server certificate application form.

In the certificate request it is also possible to use the other optional fields mentioned in the VRK document "Instructions for acquisition of a server certificate". The direct message exchange with Customs does not require the use of the optional fields.

2.2.2.2 Creation of certificate request file

The creation of a certificate request file varies according to the software. It is not possible to give universally applicable specific guidance on the issue, but detailed instructions can be found in the technical documentation of the software used by the customer. If necessary, the customer should contact the software supplier.

2.2.2.3 Delivery method and address of certificate request

The certificate request file created on a server should be sent by e-mail to certificate production of the Population Register Centre. The e-mail address is given in the VRK document "Instructions for acquisition of a server certificate".

2.2.2.4 Delivery method and time of server certificate

VRK will send the server certificate created on the basis of the certificate request by e-mail to the applicant or the contact person responsible for technical matters as designated by the applicant.

Time of delivery for the server certificate is five working days after the properly completed application form, appendices and certificate request have been received by the Population Register Centre.

2.2.2.5 Installing the server certificate

The installation of the server certificate varies according to the software. It is not possible to give specific guidance on the issue that is universally applicable. Detailed instructions should be found in the technical documentation of the software used by the customer. If necessary, the customer should contact the software supplier.

2.3 Opening the connection

In direct message exchange, the messaging client is connects to the direct message exchange web service of Customs over the internet. In the customer application form, the company operating as a message declarant provides its own contact information and possibly the contact information of the service provider operating as an intermediary.

On the basis of the information in the customer application form, the customer is granted the necessary authorisations by Finnish Customs. The Customs testing official informs the applicant of the date when the direct message exchange connection is opened.

2.4 Transmission data

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).

2.4.1 URL address of the service

During testing, SOAP requests are sent over the Internet to the following URL address:

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

In production, the SOAP message is sent over the internet to the following URL address:

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

2.4.2 RequestHeader

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

An 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>
```

2.4.3 ApplicationRequest

When transmitting the message to Customs, the SOAP message contains the XML document ApplicationRequest. The parameters contained in the ApplicationRequest have the following purpose:

- The XML element 'DeclarantBusinessID' contains the identification of the message declarant.
- The XML element 'MessageBuilderBusinessID' contains the identification of either the message declarant or the service provider.
- The XML element 'Application' contains the short title of the target system of Customs. The names in use are 'AREX', 'ELEX', 'EMCS'.
- **In testing**, 'TEST' is placed as the content of The XML element 'Environment'.
- **In production**, 'PRODUCTION' is placed as the content of The XML element 'Environment'.

An 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>
```

2.4.4 Application message

The XML document ApplicationRequest contains the XML application message as the value of XML element Content. In the test situation, the country code FI and the Business ID of Customs, FI0245442-8, will be used as the Customs identification in the XML application message.

When testing against the AREX and ELEX systems, in the Message block of the application message, "1" is selected under the test indicator (XML element 'test').

An example of the Message block in the AREX message:

```
<arex:Message>
  <wco:function>FI007A</wco:function>
  <wco:sender>FI2340001-5</wco:sender>
  <wco:recipient>FI0245442-8</wco:recipient>
  <wco:issue>2010-03-19T11:15:04.515+02:00</wco:issue>
  <wco:reference>FIRMA000004671</wco:reference>
  <wco:test>1</wco:test>
</arex:Message>
```

An example of the Message block in the ELEX message:

```
<Message>
  <sender>FI2340001-5</sender>
  <recipient>FI0245442-8</recipient>
  <issue>2010-03-19T11:15:04.515+02:00</issue>
  <reference>FIRMA000004672</reference>
  <version>2.0</version>
  <test>1</test>
</Message>
```

In production, when sending declarations to the AREX or ELEX system, "0" is the value of the test indicator (XML element 'test') in the Message block of the application message.

An example of the Message block in the AREX message:

```
<arex:Message>
  <wco:function>FI007A</wco:function>
  <wco:sender>FI2340001-5</wco:sender>
  <wco:recipient>FI0245442-8</wco:recipient>
  <wco:issue>2010-03-19T11:15:04.515+02:00</wco:issue>
  <wco:reference>FIRMA000004671</wco:reference>
  <wco:test>0</wco:test>
</arex:Message>
```

An example of the Message block in the ELEX message:

```
<Message>
  <sender>FI2340001-5</sender>
  <recipient>FI0245442-8</recipient>
  <issue>2010-03-19T11:15:04.515+02:00</issue>
  <reference>FIRMA000004672</reference>
  <version>2.0</version>
  <test>0</test>
</Message>
```

2.4.5 Interchange identifier

When the data content is transmitted to Customs, the SOAP Upload request contains the XML document ApplicationRequest. It contains the XML element 'Reference', which is an interchange identifier.

The application message being transmitted as XML data content can also contain an interchange identifier. In this case, the value of 'Reference' in the ApplicationRequest document and the interchange identifier in the application message must be identical.

For example, the Message block in the AREX message contains the following XML element:

```
<wco:reference>FIRMA000004671</wco:reference>
```

In this case, the content of the XML element 'Reference' in the XML document ApplicationRequest must also be set as:

```
<req:Reference>FIRMA000004671</req:Reference>
```

The interchange identifier starts with a five-letter abbreviated name of the company. The abbreviated name is formed from the name of the declarant company, and allocated by Customs. The latter part of the interchange identifier is chosen by the customer. The maximum length of the XML element is 14 characters, thus it is possible to use e.g. a nine-character consecutive number.

The value of 'Reference' in ApplicationRequest must be unique for the combination of Customs target system and declarant. This means that a company can use the same interchange identifier once for each Customs target system (e.g. AREX and ELEX).

The checks of the incoming messages carried out by Customs lead to rejection of the message if:

- The value of XML element 'Reference' of the ApplicationRequest document and the interchange identifier in the application message are not identical.
- The same interchange identifier has been received for the combination of target application and declarant previously, i.e. the interchange identifier is not **unique**.

Customs stores the interchange identifier immediately upon receiving the Upload request. The interchange identifier is **not** 'released' if an error condition occurs and Customs rejects the Upload request. This includes the situation, where the Upload request is rejected due to a temporary technical problem at Customs.

If the customer wants to send the same data content again, the customer must always change the interchange identifier prior to resending.

The production and customer testing environments of Customs are completely separated. Customer testing does not 'consume' production interchange identifiers.

2.5 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. If the PDF contains a decision made by Customs, it will include appeal instructions, which are not included in the corresponding EDI messages.

In direct message exchange, the PDF files are attached to the application response message sent by Customs. The PDF files are packed into a ZIP archive. As the data in the ZIP archive is compressed, this reduces the size of the attachment data.

The customer retrieves the application response message in XML format and the PDF files in the same transaction, by sending a Download request to the direct message exchange web service. Both the application response message and the ZIP archive containing the PDF files are included in the the same web service Download response.

The ZIP archive can be at maximum 2 megabytes (2048 kilobytes) in size. Both the application response message in XML format and the ZIP archive are base64-encoded and inserted into the XML document ApplicationResponse. The ApplicationRequest document is base64-encoded and inserted into the Download response, which the customer receives from the web service.

If the size of the ZIP archive exceeds 2 megabytes, then Customs system (e.g. ELEX) cannot deliver the PDF files to the customer through the direct message exchange web service. The Customs system informs the customer about the situation by sending the customer an application message in XML format. The customer can agree case by case with the Electronic Customs Clearance Centre concerning the delivery of files exceeding the maximum size limit.

3 The direct message exchange web service

The direct message exchange web service is described in this chapter. The principles of operation and the data content are presented in this chapter.

3.1 SOAP message and its structure

Application messages are specific to the Customs system for which they are intended. The XML structure of each set of application messages is uniquely defined to support a specific Customs procedure. The documentation released for a Customs system will also define the application-specific messages, including the XML schemas.

The direct message exchange web service uses SOAP request and response messages for transmitting application messages. The structure the SOAP requests and responses are described with a WSDL and XML schema description. The structure is standardised and independent of application messages.

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

The high level model below describes the structure of the SOAP message (request message).

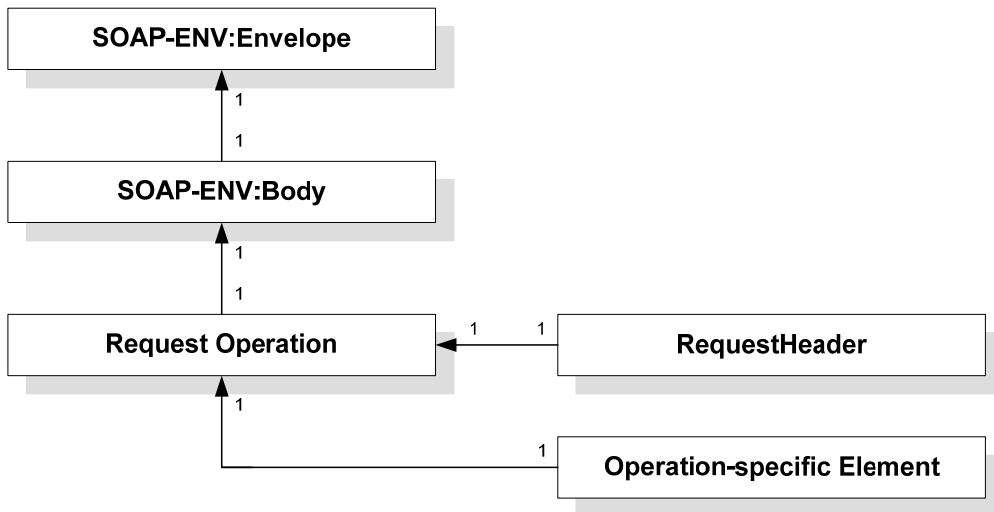


Figure 2: SOAP request message structure

SOAP messages are composed of an outer envelope, *SOAP Envelope*. The SOAP Envelope contains a *SOAP Header* and a *SOAP Body*, which both consist of different elements. The SOAP Body is a mandatory part of SOAP. The Header, on the other hand, is missing in the SOAP messages sent to Customs, as the SOAP Message Security features are not used.

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 high level model below describes the structure of the SOAP response message.

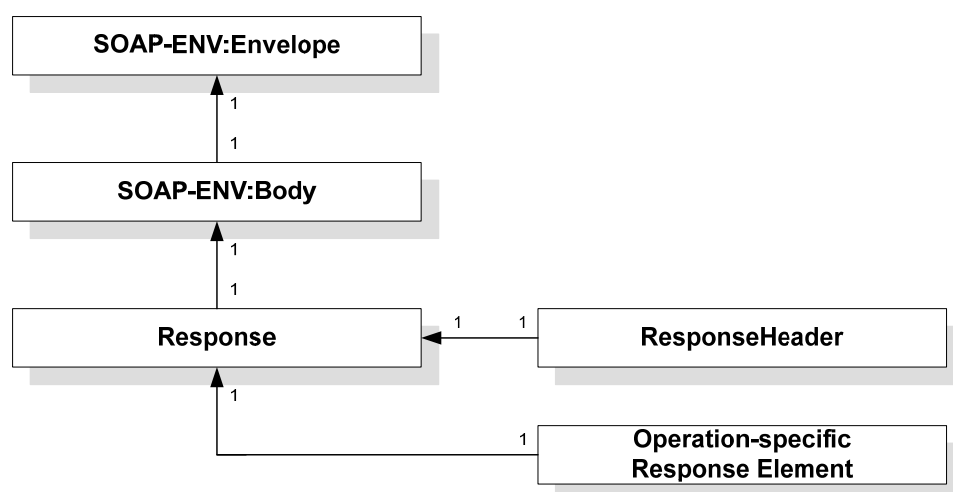


Figure 3: SOAP response message structure

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

3.2 Operations provided by the service

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

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

DownloadList = Returns a list of messages that are waiting to be retrieved. The data obtained by using this operation is used for the download operation.

Download = Used for retrieving a set of data from Customs. For example, the operation can be used for retrieving the response from Customs to a declaration which the customer has sent earlier.

3.3 The phases of building and transmitting the message

The XML application messages of Finnish Customs are uniquely defined for the application. The XML structures of Customs SOAP messages are standardised and independent of the application. The following model describes the phases of building and transmitting messages.

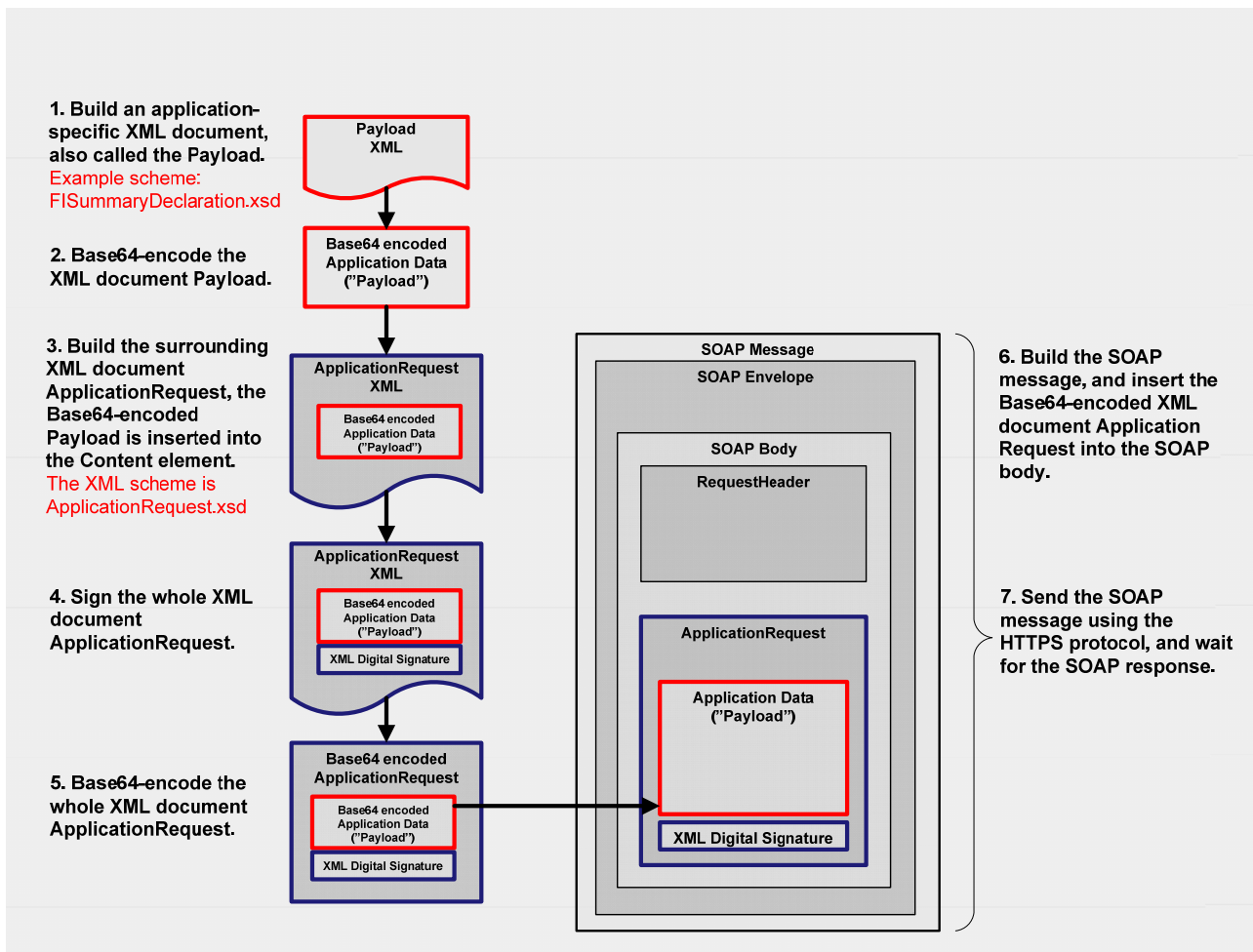


Figure 4: Building the message and transmitting the message to Customs

The **red box** contains the actual application message. The **blue box**, i.e. the ApplicationRequest block contains both the application message and the message structure around it. The signature of the builder covers the whole content of the blue box.

The Body of the SOAP request always contains a standard XML element, RequestHeader. In addition, the request contains an operation-specific XML element.

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

The intermediary embeds a complete, signed message structure in a SOAP Envelope, which is the outermost element of the SOAP message. The Soap Envelope contains a SOAP Header and a SOAP Body, which both consist of different elements. The SOAP Body is a mandatory part of SOAP. The Header, on the other hand, is missing in the SOAP messages sent to Customs, as the SOAP Message Security features are not in use.

3.4 Technical customer parties and roles

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

- DeclarantBusinessId (message declarant)
- MessageBuilderBusinessId (builder)
- IntermediaryBusinessId (intermediary)

The above terms refer to the customer parties in a purely technical sense. The terms do not refer to the economic operators of the application messages. For example, intermediary does not refer to the intermediary of goods or transport.

Two **technical parties** can engage in direct message exchange with Customs: the message declarant and optionally the service provider of the declarant.

1. The **message declarant** is the party who has the responsibility to provide declaration data or similar data to Customs and who does this by using message exchange. Depending on the procedure, the message declarant can be a principal, a representative (for example a forwarding agency) or other.

If it is a question of, for example, customs declarations, the decisions made by Customs apply to the message declarant. If the message declarant is a representative, the decisions also apply to the principal of the message declarant.

2. The **service provider** is a party who can take over certain technical roles related to direct message exchange.

The **technical roles** are related to building messages, and conveying the messages to Customs.

- The company whose data system constructs messages in the format prescribed by Customs and signs the messages with an electronic XML signature has the role of the **message builder**.
- The messages are transferred to Customs over a HTTPS connection, using SOAP messages that are specific to the direct message exchange web service. The company whose data system connects to the direct message exchange web service over the internet has the role of the **intermediary**.

Both technical roles are related to the direct message exchange web service only. They are not related to any Customs procedure.

3.5 Authentication and authorisation

When the message is processed by Customs, a number of security checks are done. Security checks related to data transfer and business level security checks have been separated. This makes it possible for the customer to use a service provider as an intermediary between the customer and Customs.

3.5.1 Authentication of the intermediary

Intermediary is a technical role, which is related to the direct message exchange web service. Either the message declarant has this role, or the message declarant has outsourced the role to a service provider.

For the HTTPS connection, the intermediary needs a server certificate. During connection the service checks that the certificate is not on the certificate revocation list.

The service authenticates the HTTPS connection of the intermediary against the intermediary's server certificate. Customs authenticates the intermediary by checking that the VAT number, which can be derived from the value of the XML element `IntermediaryBusinessId`, matches the VAT number in the server certificate.

3.5.2 Authentication of the builder

Message builder is a technical role, which is related to the direct message exchange web service. It is not related to any Customs procedure. Either the message declarant has this role, or the message declarant has outsourced the role to a service provider. The service provider can have the message builder role only in case the service provider also has the role of the intermediary of the message declarant in question.

For the XML signature, the builder needs a server certificate. During the processing of the XML signature, the service checks that the certificate is not on the certificate revocation list.

The service authenticates the message builder against the server certificate in the XML signature. Customs authenticates the builder by checking that the VAT number, which can be derived from the value of the XML element MessageBuilderId, matches the VAT number in the server certificate.

3.5.3 Authentication of the message declarant

The message declarant is authenticated with the country code and the business ID in the XML element DeclarantBusinessID. The 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.

3.5.4 Authorisation of the message declarant and the service provider

In terms of technical operation, the message declarant has always authorised a party to function as the intermediary:

- If the message declarant does not use a service provider, the message declarant has authorised himself or herself to be the intermediary.
- If the message declarant used a service provider, the message declarant has authorised the service provider to be the intermediary.

In the service, the values of XML elements DeclarantBusinessId (message declarant) and IntermediaryBusinessId (intermediary) are compared with data in the Customs' authorisation register. The intermediary can send or retrieve the message declarant's messages only in case the intermediary has been authorised by the message declarant.

The authorisation information is collected through the applications for authorisation for customers of the direct message exchange.

3.6 Restrictions on direct message exchange

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

3.6.1 Protocol versions

The interface supports the following protocol versions:

- SOAP 1.1
- HTTP 1.1
- TLS version 1 (recommended) and SSL version 3

3.6.2 Encryption algorithms

The allowed cipher suites for the HTTPS connection, in case TLS connection is in use, are:

- 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 suites for the HTTPS connection, in case SSL connection is in use, are:

- 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 is:

- RSAwithSHA256

The allowed algorithm for the DigestMethod of the XML signature is:

- SHA256

3.6.3 Restrictions related to the message

Following restrictions related to the HTTP transfer are imposed on the 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 elements
- 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 application message (element Content of the ApplicationRequest) is 512 kilobytes. The size limit refers to the application message 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

3.6.4 Restrictions related to the number of service requests

The processing of service requests is restricted as follows:

- **Upload:** The service processes not more than **one (1)** Upload request **per second** per client (IntermediaryBusinessID) connected to the service.
- **DownloadList:** The service processes not more **than one (1)** DownloadList request **per five minutes** per client (IntermediaryBusinessID) connected to the service.

- **Download:** The service processes not more than **five (5)** Download requests **per second** per client (IntermediaryBusinessID) connected to the service.
- **CheckConnectivity:** The service processes not more than **one (1)** CheckConnectivity request **per second** per client (IntermediaryBusinessID) connected to the service.

DownloadList and CheckConnectivity requests that exceed the above thresholds are rejected.

Upload and Download requests are handled more gracefully: Upon reaching the thresholds, new requests from the client are "buffered." This means that the TCP session, through which Customs received the new SOAP request, is queued by leaving it idle. The request is dequeued by resuming processing. An queued Upload or Download request will be terminated only when it stays idle for such a long time that a timeout value is reached.

3.7 Response in normal and error situations

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'.

3.7.1 ResponseHeader codes

In an error situation, the message interface strives to return the normal 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 and the content of the Response Text XML element are as presented in the table below:

ResponseCode	ResponseText	Description
000	OK	The operation has been completed successfully.
450	Invalid HTTP connection parameters	A limit imposed for the HTTP connection has been exceeded.
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.
455	Rejected by policy	A generic response code for an error that have occurred during 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. It is returned when a more specific response code has not been defined.
457	Allowed message frequency exceeded	A limit imposed on the frequency of service requests has been exceeded.

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.
460	Intermediary id not valid	In the SOAP request, the length of the value of XML element IntermediaryBusinessId is not 9-17 characters, or 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 authorized by any declarant.
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.
465	Declarant authorization failed	The declarant is not authorized to use the target system (value of XML element Application of ApplicationRequest).
466	Builder authorization failed	The message builder is not authorized 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 authorized by the message declarant.
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 application message (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.
473	Content exceeds size limit for application	The application message (i.e. the value of XML element Content in the ApplicationRequest document) exceeds the size limit for the target application.
474	Uploads to application temporarily disabled	No 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.
476	XML signature not valid	The XML signature of the ApplicationRequest document 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 of the ApplicationRequest document uses a signature algorithm which is not among the permitted signature algorithms.
478	DigestMethod algorithm in signature not allowed	The XML signature of the ApplicationRequest document 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.
490	Backend connection error	No connection could be established from the Direct Message Exchange frontend service to the backend service.
499	Unknown Error	The Direct Message Exchange frontend service has encountered an error, for which a no specific error code has been defined.
500	A message with the same MessageBuilderBusinessId and Reference already exists.	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 application message (Content).

502	DeclarantBusinessId in ApplicationRequest and sender in content do not match.	The value of XML element DeclarantBusinessId in the ApplicationRequest does not match the identification of the sender in the application message (Content).
600	Start time too far away in the past.	An attempt was made to retrieve a list of 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 retrieve the message.
999	Backend technical error	The Direct Message Exchange backend service has encountered an error, for which no specific error code has been defined.

Table 3: ResponseCode and ResponseText

3.7.2 SOAP fault

As a rule, Customs strives to return information to the customer's system about the 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, in the SOAP fault, in which the HTTP return value is '500 Internal Error', following data is returned:

- XML element code, into which 999 is entered in cases of unexpected errors. There may be situations, where a non-numeric value such as
- XML element text, into which the text *Unexpected error* is entered in cases of unexpected errors.

Customs reserves the right to insert other than the above described data into the fault element. There exist also known error situations, where a SOAP fault containing only the generic SOAP fault elements faultcode and faultstring are returned to the client.

It is essential that the client software is prepared to also handle error situations, including the processing of SOAP faults.

4 XML structures in direct message exchange

4.1 Requirements for XML messages

The XML message requirements presented in this chapter concern both application messages, as well as the SOAP request and response messages that are used to transport the application messages.

4.1.1 General requirements for XML messages

In the XML messages of Customs, version 1.0 of the XML standard is used and the character encoding used in the messages is UTF-8. 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 encoding attribute. The UTF-8 encoding is expressed with the BOM marker. Its hexadecimal value is EF BB BF. 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.

In the messages, elements with no data should be left out. In some of the Customs systems, the elements with optional data can be marked as empty in the message, but often they cause problems. In the export system this causes a rejection.

In the messages, CDATA sections of the XML should not be used in the values of the elements.

4.1.2 XML schema

An XML schema is a description of the structure of XML messages (DTD document type definition is not used for new message specifications.) The XML schemas of Customs are released in order to make it easier to process messages in the XML format. The correctness of the formed XML message should be checked by validation: checking if it conforms to a schema. A message that has not been validated should not be sent to Customs, as such messages will not pass the validation process that occurs in the beginning of the message reception.

The schemas of the new 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 tags 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.

The XML message structure consists of elements. The schema of the message describes the elements used in the structure, their location in the message and determines what type of

content of the elements is acceptable. An element can contain either sub-elements or data. Usually a part of the elements used in the message structure have been defined in the schema of the message structure itself. However, some of the commonly used structure elements have been inserted directly from external contexts, i.e. component schemas. When using namespaces, the elements defined in the message schema belong to the namespace of the schema. The structures of elements inserted from external contexts, however, belong to the namespace of the component schema.

In addition to these, the namespace "http://www.w3.org/2001/XMLSchema-instance", which contains general specifications for XML structures, may need to be presented in the XML message. The namespace can be presented, for example, as follows:

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

The namespace needs to be presented when attributes that belong to the message are used (http://www.w3.org/TR/xmlschema-1/#Instance_Document_Constructions).

4.1.3 Declaring namespaces in XML messages

When using namespaces, the name spaces to which the elements contained in XML messages belong should be declared in the XML message. The namespaces are usually declared in the start tag of the root element of the message. There can be no elements without a namespace in a message, as the schema requires namespaces to be used. When declared, the namespaces are also given an abbreviation. 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 declared for a namespace can be chosen freely. The chosen abbreviation is only used within that specific 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 usually formed automatically by using consecutive 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">
```

4.1.4 Attaching an element to a namespace in XML messages

When using namespaces, each of the elements in the messages belongs to a specific namespace. Elements without a namespace are not allowed. The namespace of an element in an XML message is determined by attaching the abbreviation of the namespace as a prefix to the element name. The prefix is made up of the abbreviation of the namespace to which the element belongs. Each namespace has been given its own abbreviation in the declaration of the namespaces used in the message. The prefix and the element are separated by a colon. In the name, a prefix 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 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 with a similar message structure, 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.

4.1.5 Default namespace as a special case of the above

There is no need to include a default namespace in a message, but it is possible to define one of the namespaces used in the message 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 and shall not use an abbreviation as a prefix for the elements that belong to it. When a default namespace has been declared in a message, the assumption is that all elements without a prefix in a tag belong to the 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. This abbreviation must be used as a prefix of the elements. Below is an example of how to use a default namespace in the FISummaryDeclaration message (the beginning of the 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>003701289133-</wco:sender>
    <wco:recipient>003702454428</wco:recipient>
    <wco:issue>2010-04-13T09:37:16</wco:issue>
    <wco:reference>AUGLJ1000004-01</wco:reference>
    <wco:reference>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>
```

4.2 WSDL and XSD files

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

- CustomsCorporateService.wsdl

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

- ApplicationRequest.xsd
- ApplicationResponse.xsd
- ApplicationMessageTypes.xsd
- EchoContent.xsd
- WsdTypes.xsd
- xmldsig-core-schema.xsd

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

http://www.tulli.fi/en/businesses/eServices/message/direct_message_exchange/index.jsp

The package is located under the heading “Direct message exchange, XML schemas”.

The above address is the only place from where to obtain the WSDL. The WSDL **cannot** be retrieved by sending a HTTP GET ‘?wsdl’ request to the URL of the web service.

4.3 Operations provided by the service

In WSDL, the functionality provided by the service is grouped into logical operations. An operation is made up of a set of data elements that are interrelated with each other, usually of two (main) data elements: request and response. The message exchange interface provides operations similar to the ones shown in the tables below:

4.3.1 Upload

Operation	Description
Upload	Used for sending a data entity, e.g. a customs declaration to Customs systems. As a result, a technical acknowledgement of reception of the declaration (= being processed) is received. After the technical acknowledgement, the declaration is processed in the Customs system. Further processing (usually) provides a response message, which has to be retrieved separately with a separate download operation. A list of messages that can be retrieved can be obtained with the DownList operation (see the following operations).

Request

UploadRequest

RequestHeader

ApplicationRequestMessage

Response

UploadResponse

ResponseHeader

MessageInformation

Table 4: Upload4.3.2 DownloadList

Operation	Description
DownloadList	Used for retrieving a data list from the Customs systems. Contains a list of basic information of the response messages that are to be downloaded. The information that is 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 5: DownloadList

4.3.3 Download

Operation	Description
Download	Used for retrieving one data entity from the Customs systems. For example, a response to an earlier sent Customs declaration can be retrieved with the operation.
	Request
	DownloadRequest
	RequestHeader
	DownloadMessageFilteringCriteria
	Response
	DownloadResponse
	ResponseHeader
	ApplicationResponseMessageInformation
	ApplicationResponseMessage

Table 6: Download

4.3.4 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 7: CheckConnectivity

4.4 Operation-specific XML elements

4.4.1 Upload_request

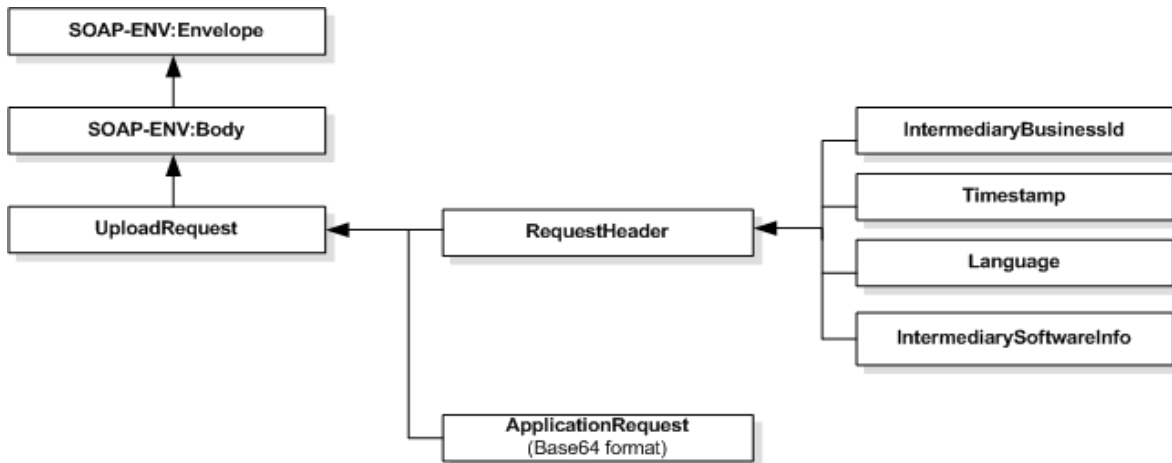


Figure 5: Upload request

Description of ApplicationRequest element:

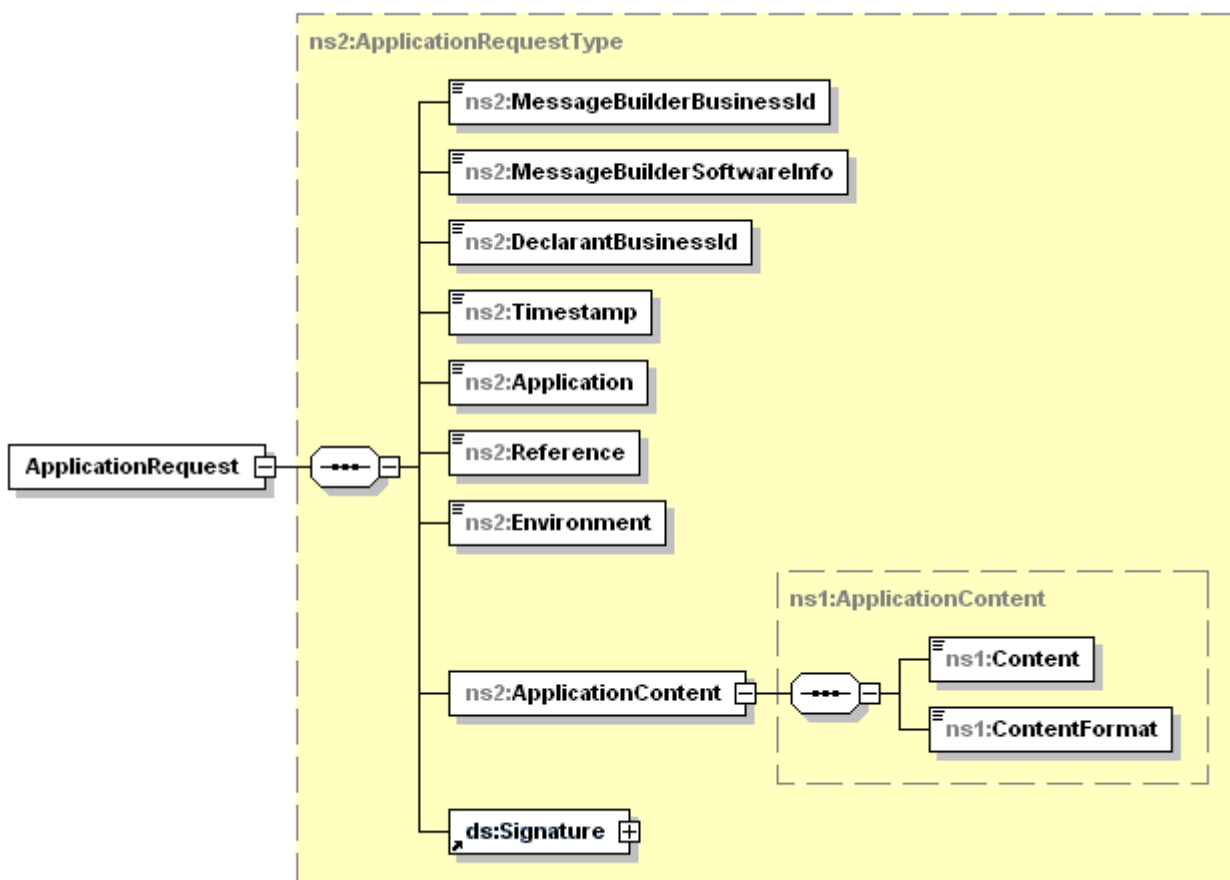


Figure 6: Description of ApplicationRequest

4.4.2 Upload, response

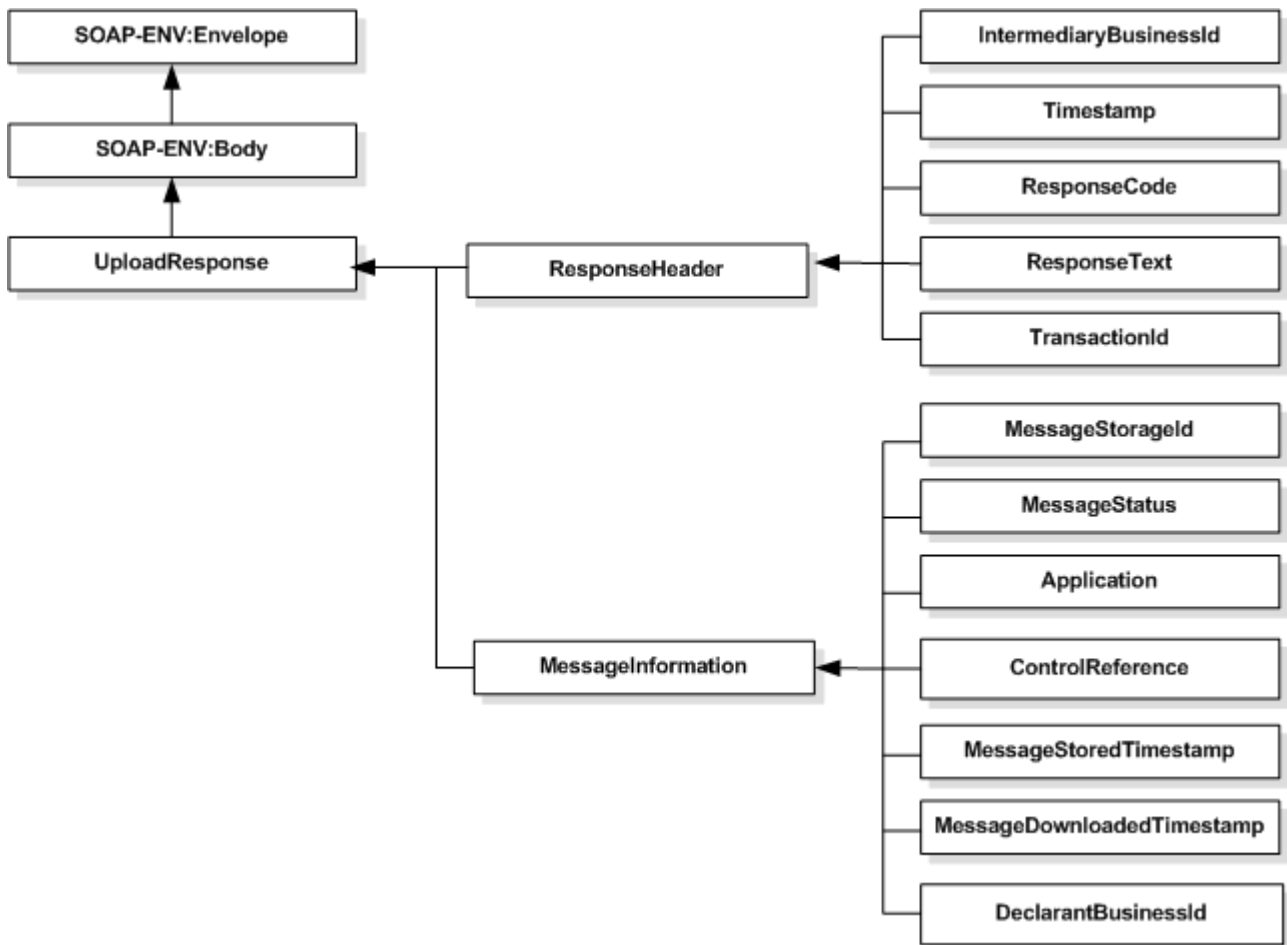


Figure 7: Upload response

4.4.3 DownloadList, request

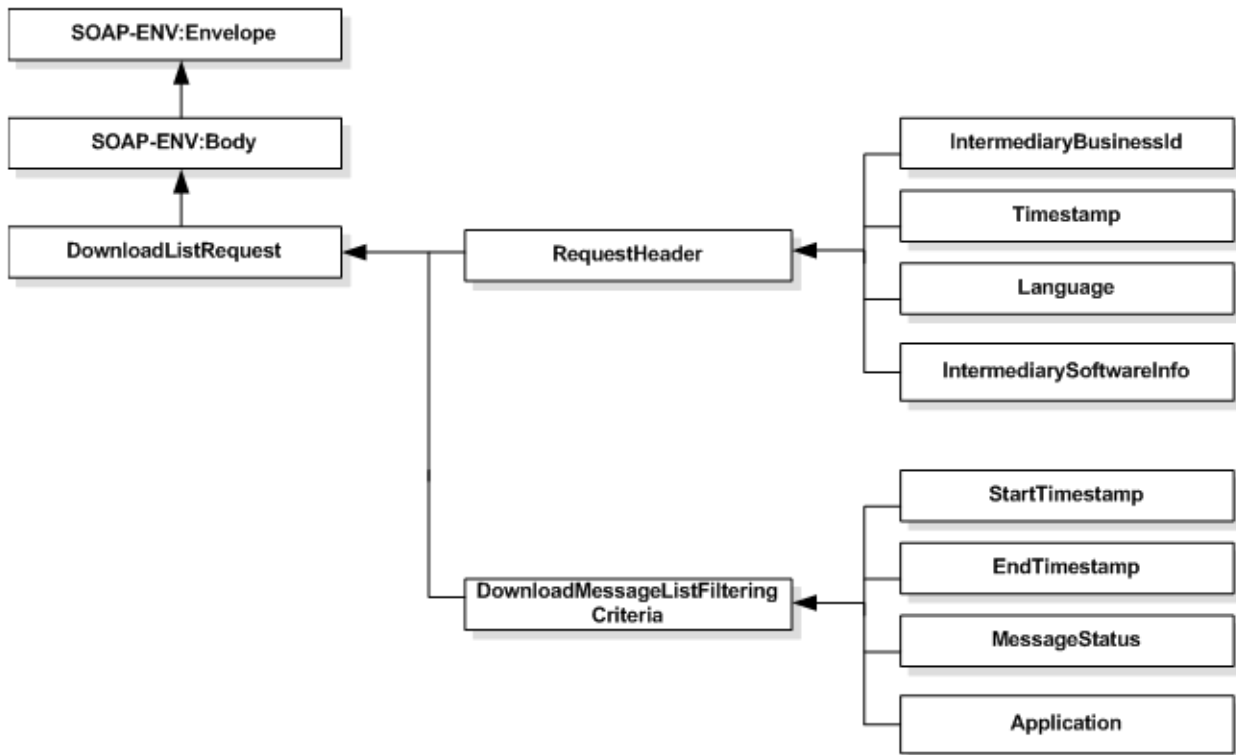


Figure 8: DownloadList request, using elements StartTimestamp and EndTimestamp

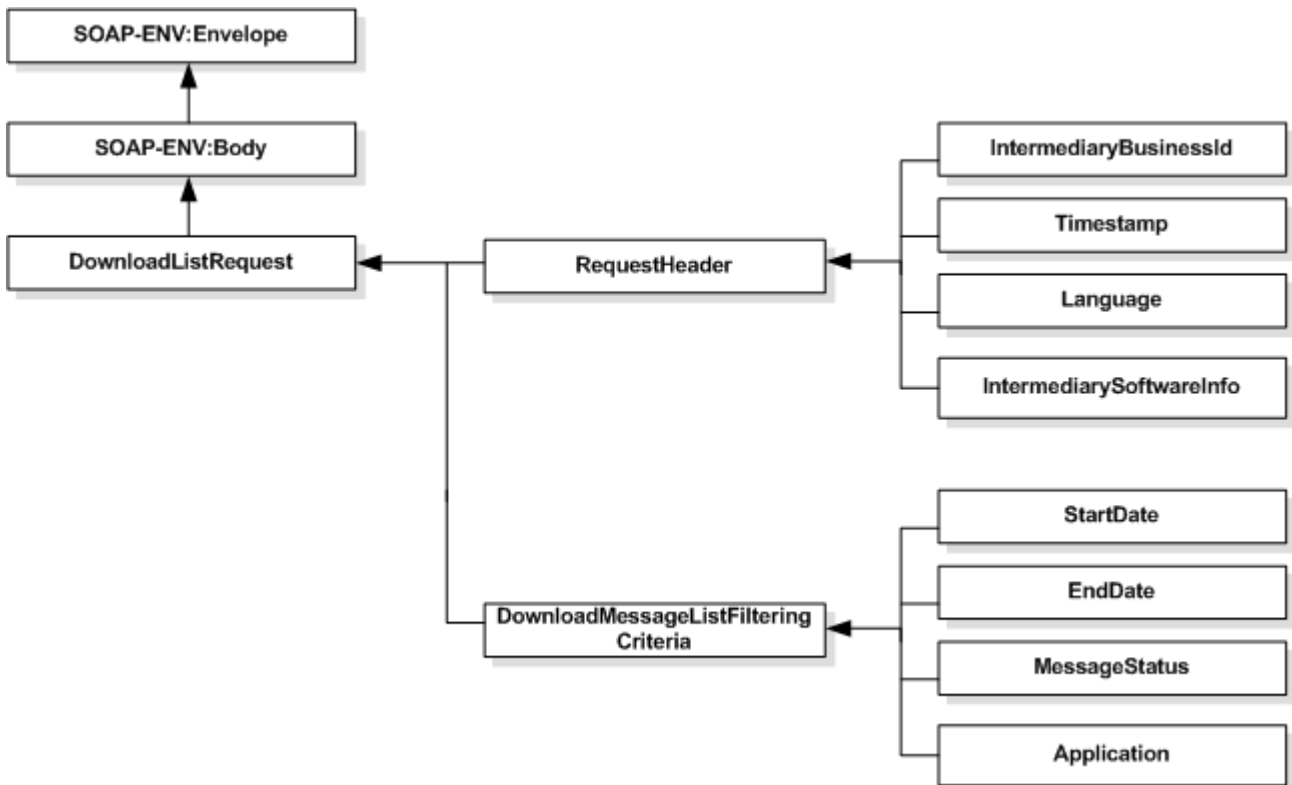


Figure 9: DownloadList request, using elements StartDate and EndDate

4.4.4 DownloadList, response

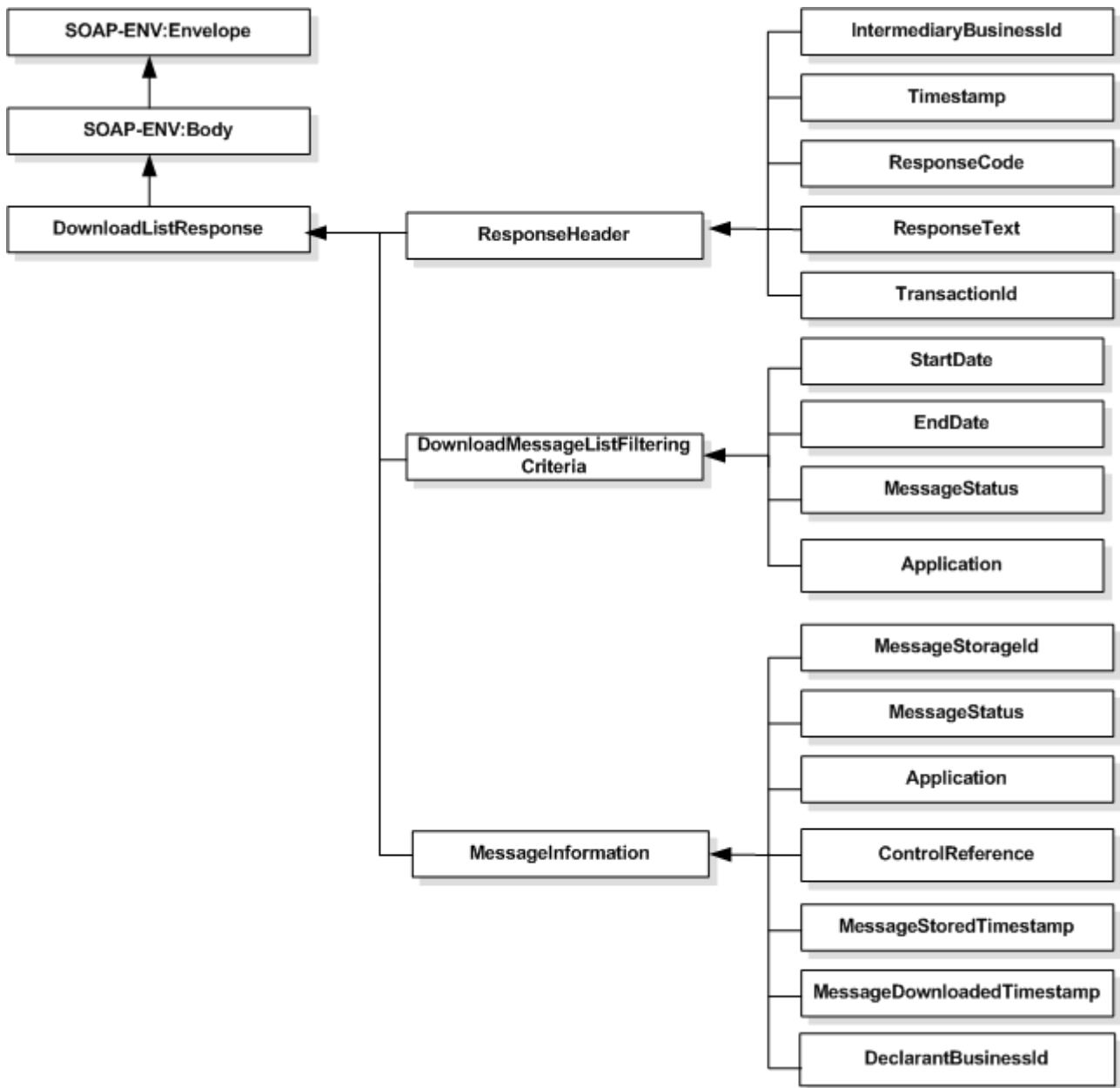


Figure 10: DownloadList response

4.4.5 Download, request

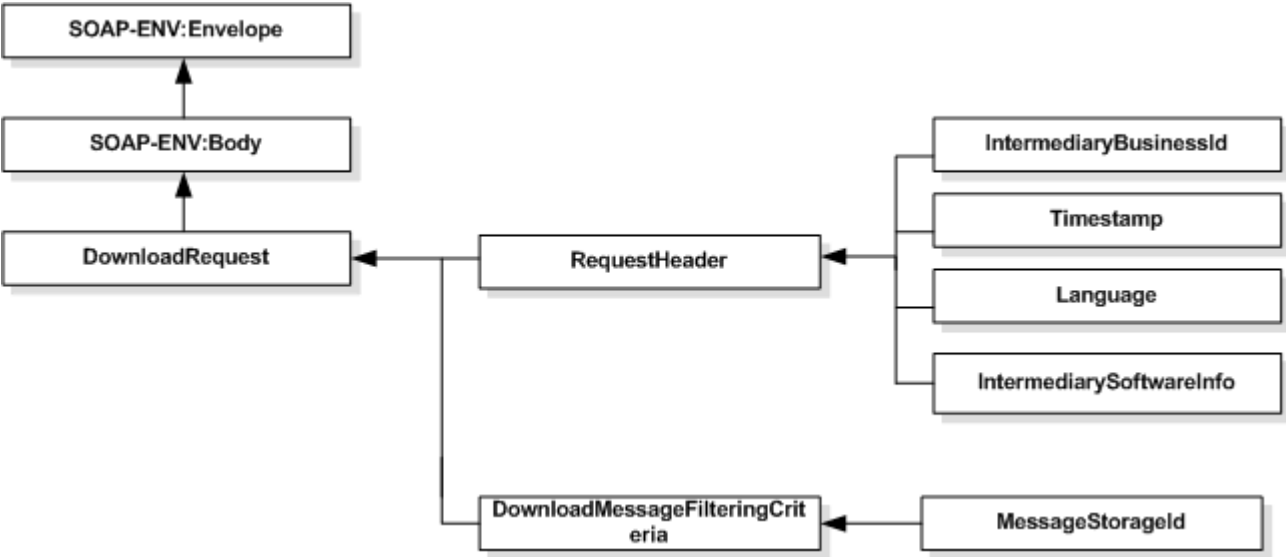


Figure 11: Download request

4.4.6 Download, response

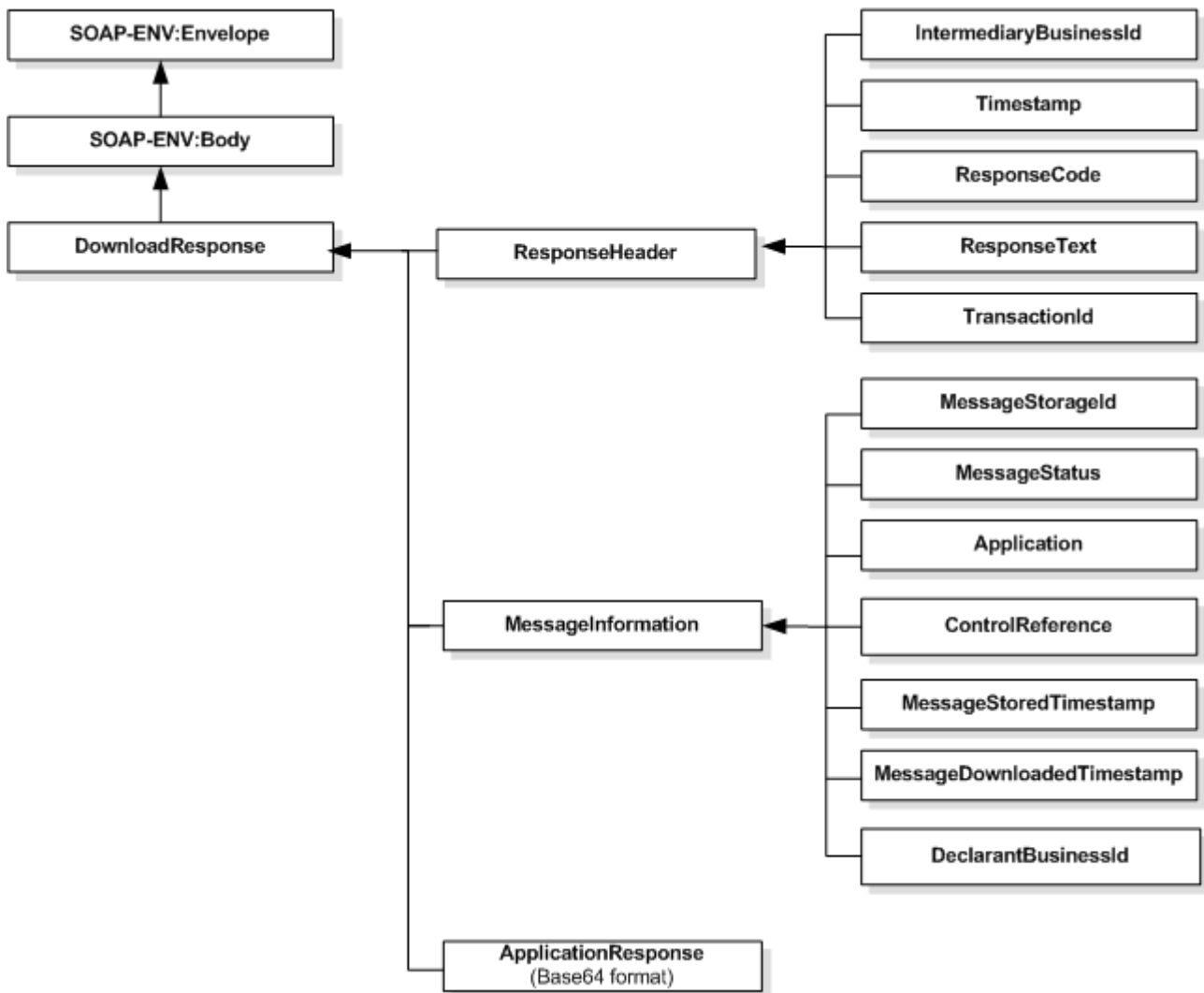


Figure 12: Download response

Description of ApplicationResponse element:

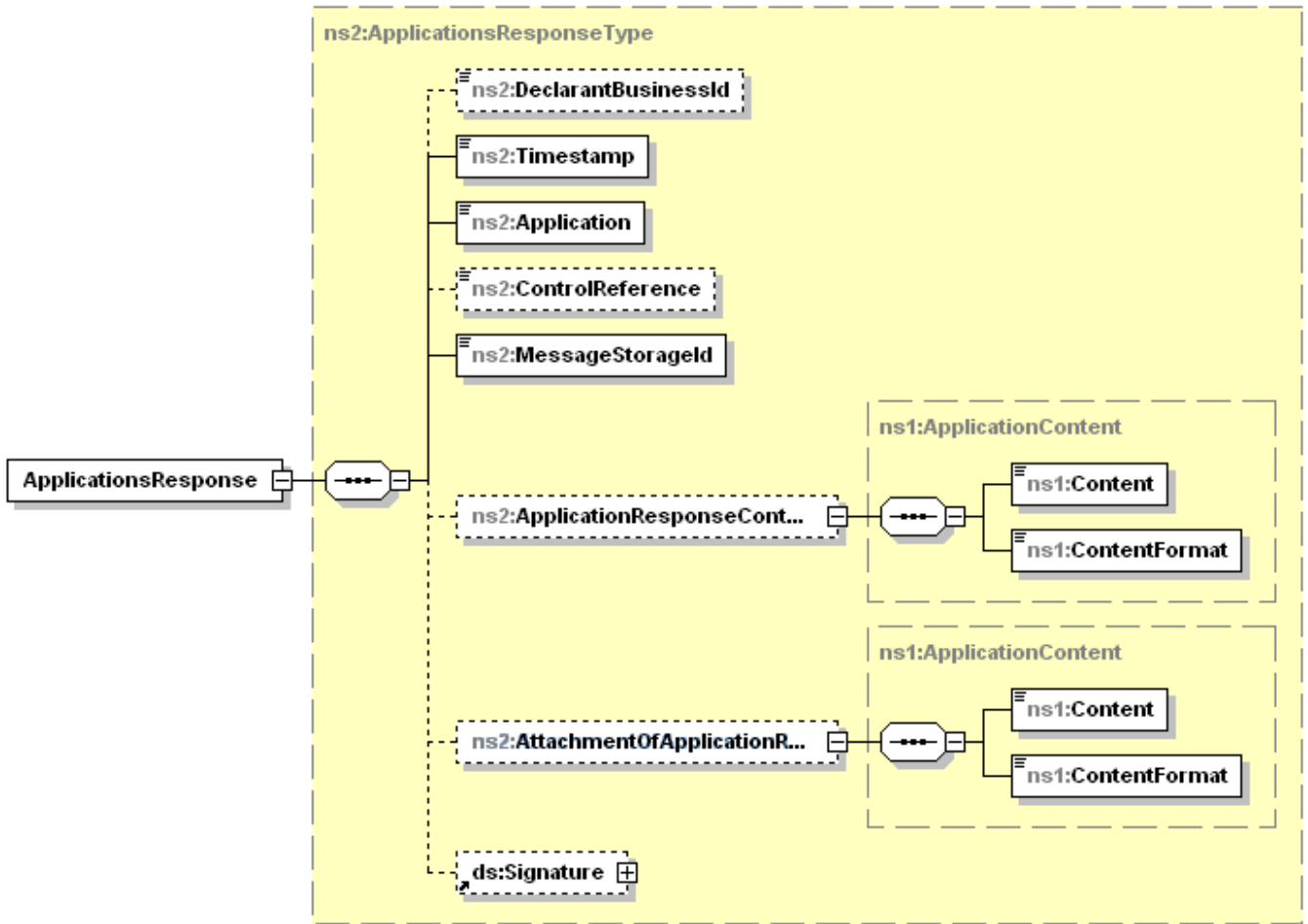


Figure 13: Description of ApplicationResponse

4.5 Descriptions of the data of the XML elements

4.5.1 Descriptions of the data of the XML elements used by the intermediary

The data elements that are of interest to the intermediary and defined in WSDL are described in the following tables.

Element name	Description	Type	Required (Y/N)	Occurrence
RequestHeader	This element is the header for each request sent to the WS interface. This header is constructed by the intermediary.			
IntermediaryBusinessId	Country code and Business ID of the intermediary	string	Y	[1..1]
Timestamp	Time and date of sending the request Data type is f ISODatetime. If the time zone has not been defined, the Finnish time zone is used.	dateTime	Y	[1..1]
Language	Language code, which indicates what language is used in the informative data elements. At the moment 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 8: RequestHeader

Element name	Description	Type	Required (Y/N)	Occurrence
ApplicationRequestMessage	Contains the XML message by the builder. XML is base64-encoded.	base64Binary	Y	[1..1]

Table 9: ApplicationRequestMessage

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 f ISODatetime. If the time zone has not been defined, UTC time zone is default.	dateTime	Y	[1..1]
ResponseCode	Response code which indicates the success or failure of the operations followed by the request.	string	Y	[1..1]
ResponseText	Verbal description of the response code (in English).	string	Y	[1..1]
TransactionId	Identification of the transaction, with which the request and the response can be bundled together.	string	Y	[1..1]

Table 10: ResponseHeader

Element name	Description	Type	Required (Y/N)	Occurrence
MessageInformation	This element contains basic information about the message.			
MessageStorageId	Unique ID of the message, through which the response message can be retrieved from the Customs system.	string	Y	[1..1]
MessageStatus	Message status: <ul style="list-style-type: none"> NEW – Message has not yet been retrieved DLD – The message has already been retrieved 	string	Y	[1..1]
Application	Short name of the Customs application, to which the message is connected.	string	Y	[1..1]
ControlReference	Originally the identifier of the interchange of the customer (the builder of the XML message), with which the customer can bundle together all the messages in the transaction. Additional information in the description of the element <i>Reference</i> of the element <i>ApplicationRequest</i> .	string	N	[0..1]
MessageStoredTimestamp	Date and time (time stamp) when the message was saved in the Customs	dateTime	Y	[1..1]

MessageDownloadedTimestamp	system. Date and time (time stamp) when the message was retrieved from the Customs system (if the download operation has been performed).	dateTime	N	[0..1]
DeclarantBusinessId	Country code and Business ID of the declarant	string	Y	[1..1]

Table 11: MessageInformation

Element name	Description	Type	Required (Y/N)	Occurrence
DownloadMessageListFilteringCriteria	This element data is used as filtering criteria when requesting (downloadList) which response messages can be retrieved one by one (download).			
StartDate	The start date, during which or after which the message has been saved.	date	Y	[1..1]
EndDate	The end date, during which or before which the message has been saved.	date	Y	[1..1]
StartTimestamp	The start timestamp, during which or after which the message has been saved.	dateTime	Y	[1..1]
EndTimestamp	The end timestamp, during which or before which the message has been saved.	dateTime	Y	[1..1]
MessageStatus	It is possible to use only either the combination StartDate and EndDate, or the combination StartTimestamp and EndTimestamp. The request can be filtered by message status. The following status data can be used: <ul style="list-style-type: none"> NEW – A list of of the response messages that have not yet been retrieved (downloaded) is requested. DLD - A list of the response messages that have been retrieved is requested. ALL - A list of all the response messages (whether they have been retrieved or not) is requested. 	string	Y	[1..1]
Application	The short name of the Customs application, of which the message data is retrieved. If the element is left out, all the	string	N	[0..*]

message data of the Customs applications are requested.

Table 12: DownloadMessageListFilteringCriteria

Element name	Description	Type	Required (Y/N)	Occurrence
DownloadMessageFilteringCriteria	This element is needed as key data, when retrieving one (response) message from the Customs system.			
MessageStorageId	A unique ID of the message, through which the response message can be retrieved from the Customs message storage.	string	Y	[1..1]

Table 13: DownloadMessageFilteringCriteria

Element name	Description	Type	Required (Y/N)	Occurrence
ApplicationResponseMessage	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.	base64Binary	Y	[1..1]

Table 14: ApplicationResponseMessage

Element name	Description	Type	Required (Y/N)	Occurrence
EchoContent	Is used for testing purposes: input and output data for the echo operation.			
Text	Freeform text	string	Y	[1..1]
Signature	XML signature. EchoContent is signed.	string	N	[0..1]

Table 15: EchoContent

4.5.2 Descriptions of the data of the XML elements used by the builder

The data elements that are of interest to **the XML message builder** and defined in the separate schemas are described in the following tables.

Element name	Description	Type	Required (Y/N)	Occurrence
ApplicationRequest	XML constructed by the message builder. Contains application-specific data (the actual application message and the parameters associated with the application message). The XML signed. Signature by the message builder.			
MessageBuilderBusinessId	Country code and Business ID of the XML message builder.	string	Y	[1..1]
MessageBuilderSoftwareInfo	Name of the software and version number used by the message builder.	string	Y	[1..1]
DeclarantBusinessId	Country code and Business ID of the declarant	string	Y	[1..1]
Timestamp	Timestamp of when the message was built. Data type is f ISODatetime. If the time zone has not been defined, UTC time zone is default.	dateTime	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	<p>Interchange identifier of the customer. With this identifier, the customer can bundle together the message sent by the customer, and the response messages received from Customs.</p> <p>The interchange identifier is unique for each application and declarant.</p> <p>It is formed as follows: the five-letter abbreviated name of the company combined with the nine-character consecutive number. The minimum length of the element is 6 characters and the maximum length is 14 characters.</p> <p>The abbreviated name is formed from the name of the declarant company. Usually the builder of the XML message allocates the interchange identifier.</p> <p>An example of the first interchange identifier of a company, for which Customs has allocated the abbreviated</p>	string	Y	[1..1]

Environment:	name FIRMA: FIRMA000000001. The environment of Customs, at which the action is targeted. Possible values: <ul style="list-style-type: none"> • PRODUCTION • TEST 	string	Y	[1..1]
ApplicationContent	Contains the actual application message, which is directed to the Customs application. See description of the ApplicationContent element.	element	Y	[1..1]
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.	Signature	Y	[0-1]

Table 16: ApplicationRequest

Element name	Description	Type	Required (Y/N)	Occurrence
ApplicationContent	The application message with additional data.			
Content	Contains the application message base64-encoded (e.g. AREX or ELEX message).	string	Y	[1..1]
ContentFormat	Format of the Content element. Possible values are MIME media types. As a customer can only convey XML messages to Customs, the customer must set the value to 'application/xml'. Previously, the only allowed value was 'XML'. The continued use of this value is allowed.	string	Y	[1..1]

Table 17: ApplicationContent

Element name	Description	Type	Required (Y/N)	Occurrence
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 application message and the parameters associated with the application message). The XML signed. Signature by the message builder.			
DeclarantBusinessId	Country code and Business ID of the declarant	string	Y	[1..1]
Timestamp	Timestamp of when the message was built. Data type is ISODateTime. If the	dateTime	Y	[1..1]

	time zone has not been defined, UTC time zone is default.			
Application	The short name of the Customs application, from which the actual application message placed in the ApplicationResponseContent element is directed.	string	Y	[1..1]
ControlReference	Originally the interchange identifier of the customer. It is unique for each application and declarant. With this identifier, the customer can bundle together the message sent by the customer, and the response messages received from Customs.	string	N	[0..1]
	For additional information, see the description of the subelement <i>Reference</i> of the element <i>ApplicationRequest</i> .			
MessageStorageId	The unique Id of the message.	string	Y	[1..1]
ApplicationResponseContent	Contains the actual response message created by the Customs application. See description of the ApplicationContent element.	Element	N	[0-1]
AttachOfApplicationResponseContent	Attachment. The message from Customs to the customer may contain as an attachment one ZIP archive. Customs declares as the value of the element ContentFormat 'application/zip'. See description of the ApplicationResponseContent element.	Element	N	[0..1]
Signature	XML signature To be used only upon specific agreement.	Signature	N	[0..1]

Table 18: ApplicationResponse

4.6 Example message of direct message exchange

This section illustrates how an application message is packaged into an ApplicationRequest document and further into an Upload request.

Further examples can be downloaded from the following address on the Customs website:

http://www.tulli.fi/en/businesses/eServices/message/direct_message_exchange/index.jsp

A package containing example messages is located under the heading "Direct message exchange, examples".

4.6.1 Application message

```
<?xml version="1.0" encoding="UTF-8"?>
<FIExportDeclaration xmlns="http://tulli.fi/schema/elex/v2">
  <Message>
    <sender>FI2340001-5</sender>
    <recipient>FI0245442-8</recipient>
    <issue>2010-12-03T09:34:00.000+02:00</issue>
    <reference>FIRMA000000001</reference>
    <version>2.0</version>
    <test>1</test>
  </Message>
  <Declaration>
    <Exporter>
      <Party>
        <identity>FI2340001-5</identity>
        <identityExtension>T0001</identityExtension>
        <name!>Sample payload, not a real message!</name!>
      </Party>
    </Exporter>
  </Declaration>
  <GoodsShipment>
    <UCR>
      <traderReference>Sample payload, not a real message!</traderReference>
    </UCR>
  </GoodsShipment>
</FIExportDeclaration>
```

4.6.2 ApplicationRequest document

The application message is Base64-encoded in the element Content.

```
<?xml version="1.0" encoding="UTF-8"?>
<req:ApplicationRequest
  xmlns:req="http://tulli.fi/schema/corporateservice/appl/v1"
  xmlns:mess="http://tulli.fi/schema/corporateservice/v1">
  <req:MessageBuilderBusinessId>FI2340001-5</req:MessageBuilderBusinessId>
  <req:MessageBuilderSoftwareInfo>Examples 1.5</req:MessageBuilderSoftwareInfo>
  <req:DeclarantBusinessId>FI2340001-5</req:DeclarantBusinessId>
  <req:Timestamp>2010-12-03T09:41:12.152+02:00</req:Timestamp>
  <req:Application>ELEX</req:Application>
  <req:Reference>FIRMA000000001</req:Reference>
  <req:Environment>TEST</req:Environment>
  <req:ApplicationContent>
    <mess:Content>base64-encoded application message</mess:Content>
    <mess:ContentFormat>application/xml</mess:ContentFormat>
  </req:ApplicationContent>
</req:ApplicationRequest>
```

4.6.3 Signed ApplicationRequest document

The XML signature adds a Base64 encoded certificate, a signature and a digest to the ApplicationRequest document.

```
<?xml version="1.0" encoding="UTF-8"?>
<req:ApplicationRequest
  xmlns:req="http://tulli.fi/schema/corporateservice/appl/v1"
  xmlns:mess="http://tulli.fi/schema/corporateservice/v1">
  <req:MessageBuilderBusinessId>FI2340001-5</req:MessageBuilderBusinessId>
  <req:MessageBuilderSoftwareInfo>Examples 1.5</req:MessageBuilderSoftwareInfo>
  <req:DeclarantBusinessId>FI2340001-5</req:DeclarantBusinessId>
  <req:Timestamp>2010-12-03T09:41:12.152+02:00</req:Timestamp>
  <req:Application>ELEX</req:Application>
  <req:Reference>FIRMA000000001</req:Reference>
  <req:Environment>TEST</req:Environment>
  <req:ApplicationContent>
    <mess:Content>base64-encoded application message</mess:Content>
    <mess:ContentFormat>application/xml</mess:ContentFormat>
  </req:ApplicationContent>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
  <ds:CanonicalizationMethod
    Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
  <ds:SignatureMethod
    Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
  <ds:Reference URI="">
  <ds:Transforms>
  <ds:Transform
    Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
  <ds:Transform
    Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments"/>
  </ds:Transforms>
  <ds:DigestMethod
    Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
  <ds:DigestValue>base64-encoded content</ds:DigestValue>
  </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>base64-encoded content</ds:SignatureValue>
  <ds:KeyInfo>
  <ds:X509Data>
  <ds:X509Certificate>base64-encoded content</ds:X509Certificate>
  </ds:X509Data>
  <ds:KeyValue>
  <ds:RSAKeyValue>
  <ds:Modulus>base64-encoded content</ds:Modulus>
  <ds:Exponent>AQAB</ds:Exponent>
  </ds:RSAKeyValue>
  </ds:KeyValue>
  </ds:KeyInfo>
  </ds:Signature></req:ApplicationRequest>
```

4.6.4 The final SOAP message (UploadRequest)

The XML signature adds a Base64 encoded certificate to the ApplicationRequest document.

The RequestHeader of the example contains intermediary data, the rest of the message is the Base64 encoded ApplicationRequestMessage.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:cs="http://tulli.fi/ws/corporateservice/v1"
  xmlns:cst="http://tulli.fi/ws/corporateservicetypes/v1">
  <soapenv:Body>
```

```
<cs:UploadRequest>
  <cs:RequestHeader>
    <cs:IntermediaryBusinessId>FI2340001-5</cs:IntermediaryBusinessId>
    <cs:Timestamp>2010-12-03T08:19:00.376Z</cs:Timestamp>
    <cs:Language>EN</cs:Language>
    <cs:IntermediarySoftwareInfo>Examples 1.5</cs:IntermediarySoftwareInfo>
  </cs:RequestHeader>
  <cs:ApplicationRequestMessage>base64-encoded
content</cs:ApplicationRequestMessage>
</cs:UploadRequest>
</soapenv:Body>
</soapenv:Envelope>
```

Appendix 1. Technical standards utilised in direct message exchange

Standard	Version
WS-I Basic Profile	1.1
SOAP	1.1
WSDL	1.1
HTTP	1.1
SSL	3.0
TLS	1.0
XMLDSIG	1.0 http://www.w3.org/TR/2002/REC-xmlsig-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 19: Technical standards utilised in direct message exchange

Appendix 2. Further information

(Only available in Finnish)

The Finnish Customs:	http://www.tulli.fi/en
Summary declarations	http://www.tulli.fi/fi/yrityksille/tuonti/saapuminen/index.jsp http://www.tulli.fi/fi/yrityksille/vienti/poistuva/index.jsp
Export to third countries	http://www.tulli.fi/fi/yrityksille/vienti/index.jsp elex@tulli.fi
Import from third countries	http://www.tulli.fi/fi/yrityksille/tuonti/index.jsp
Transit	http://www.tulli.fi/fi/yrityksille/muut_tullimenettelyt/passitus/index.jsp
Intra-Community import and export	http://www.tulli.fi/fi/suomen_tulli/ulkomaankauppatilastot/intrastat/index.jsp
EDI in general	http://www.tieke.fi/in_english/
EDIFACT	http://www.unece.org/trade/untdid/welcome.htm
XML	http://www.w3.org/XML/
EU Commission	http://ec.europa.eu/taxation_customs/customs/policy_issues/customs_security/index_en.htm
The World Customs Organization WCO	http://www.wcoomd.org/sw_overview_wco.htm
the Finnish Customs RSS service	http://www.tulli.fi/fi/suomen_tulli/julkaisut_ja_esitteet/RSS_palvelu/index.jsp

Table 20: Further information

Terminology used in procedures and message exchange can be found in a separate Excel file:

>> http://www.tulli.fi/fi/yrityksille/sahkoinenasiointi/edi/yhteiset_tiedostot/tiedostot/Sanasto-Terminologi-Terminology.xls