**Cookbook  
eJustBox Consultation Web Service  
Version 1.0**

This document is provided to you free of charge by  
**Smals**

**Fonsnylaan 20 – Av. Fonsny 20**

**1000 BRUSSELS**

All are free to circulate this document with reference to the URL source.

# Table of contents

Table of contents 2

1 Document management 5

1.1 Document history 5

2 Introduction 6

2.1 Goal of the service 6

2.2 Various highlights 6

2.3 Goal of the document 6

2.4 eJustice document references 6

2.5 Service history 6

3 Business and privacy requirements 8

4 Global overview 9

5 Step-by-step 10

5.1 Technical requirements 10

5.2 Process overview 11

5.2.1 Web Service WSDL (Web Service Definition Language) 11

5.3 eJustBox Consultation Web Service 13

5.3.1 Lifetime of a message 13

5.3.1 Out-of-Office system 13

5.3.1.1 Introduction 13

5.3.1.2 Out-of-Office restrictions 13

5.3.2 getBoxInfo Method 13

5.3.2.1 getBoxInfo Request 14

5.3.2.2 getBoxInfo Response 14

5.3.2.3 Example 15

5.3.3 getMessagesList Method 16

5.3.3.1 getMessagesList Request 16

5.3.3.2 getMessagesList Response 17

5.3.3.3 Example 19

5.3.4 getFullMessage Method 20

5.3.4.1 getFullMessage Request 20

5.3.4.2 getFullMessage Response 21

5.3.4.3 Example 21

5.3.5 MoveMessage Method 23

5.3.5.1 MoveMessage Request 23

5.3.5.2 MoveMessage Response 24

5.3.5.3 Example 24

5.3.6 DeleteMessage Method 25

5.3.6.1 DeleteMessage Request 25

5.3.6.2 DeleteMessage Response 26

5.3.6.3 Example 27

5.3.7 getHistory Method 27

5.3.7.1 getHistory Request 27

5.3.7.2 getHistory Response 28

5.3.7.3 Example 29

5.3.8 GetMessageAcknowledgmentsStatus Method 29

5.3.8.1 GetMessageAcknowledgmentsStatus Request 29

5.3.8.2 GetMessageAcknowledgmentsStatus Response 30

5.3.8.3 Example 30

5.3.9 InsertOoO Method 31

5.3.9.1 InsertOoO Request 31

5.3.9.2 InsertOoO Response 32

5.3.9.3 Example 32

5.3.10 DeleteOoO Method 33

5.3.10.1 DeleteOoO Request 33

5.3.10.2 DeleteOoO Response 34

5.3.10.3 Example 34

5.3.11 GetOoOList Method 35

5.3.11.1 GetOoOList Request 35

5.3.11.2 GetOoOList Response 35

5.3.11.3 Example 36

5.3.12 Used Types 37

5.3.12.1 Acknowledgment 37

5.3.12.2 Annex 38

5.3.12.3 Actor 38

5.3.12.4 Content 39

5.3.12.5 ContentContext 40

5.3.12.6 ContentInfo 41

5.3.12.7 ContentSpecification 41

5.3.12.8 CustomMeta 42

5.3.12.9 EJustboxIdentifier 42

5.3.12.10 Document 43

5.3.12.11 Error 43

5.3.12.12 Message 44

5.3.12.13 MessageInfo 45

5.3.12.14 Meta 45

5.3.12.15 Sender 46

5.3.12.16 Substitute 46

5.3.12.17 Status 47

6 Risks and security 49

6.1 SOAP with Attachments 49

6.2 Security 49

6.2.1 Business security 49

6.2.2 Web Service Security Policy 49

6.2.3 Security policies to apply 49

7 Test and release procedure 50

7.1 Procedure 50

7.1.1 Initiation 50

7.1.2 Development and test procedure 50

7.1.2.1 Create test cases 50

7.1.3 Release procedure 50

7.1.4 Operational follow-up 50

7.2 Test cases 50

8 Error and failure messages 52

8.1 Error Element Status Codes 52

8.2 Consultation Response Status Codes 52

8.3 Soap Fault Error Codes 54

8.3.1 Schema Validation Errors 55

8.3.2 Technical Errors 56

9 List of sources 58

To the attention of: “IT expert” willing to integrate this web service.

# Document management

## Document history

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description of changes / remarks** |
| 1.0 | 28/04/2016 | Malik Alimoekhamedov | Initial draft |

# Introduction

## Goal of the service

The eJustBox Consultation Web Service allows an authenticated user to consult information about the content associated with his eJustBox.

A user can get general information on his eJustBox, a list of messages for a specific folder and the content of a specific message. He can also move a message to his inbox and handle his out-of-offices.

The size of a message and of an eJustBox is currently limited to 10MB, **on inbox and trash bin folder**. Note that an encrypted message weighs more due to the encryption overhead.

## Various highlights

* An “Out-of-Office” system enables the sender to know if one of the recipients is absent and to send his message to a substitute or substitutes, so the sender’s message can be treated. This mechanism is comprised of three: 1. *InsertOoO* used to add an Out-of-Office, 2. *DeleteOoO* used to delete an inserted OoO and 3. *GetOoOList* used to get the list of inserted OoO.
* SOAP with Attachments (SwA) standard is used.

## Goal of the document

This document provides functional and technical information about calling .

In this service specification document, we will explain the structure and content aspects of the possible requests, as well as the replies of the eJustice web service. An example illustrates each of those messages. A list of possible errors can also be found in this document.

This information should allow (the IT department of) an organization to integrate and use the web service call.

Some technical and legal requirements must be met in order for eJustice web services to be integrated within client applications; this document is meant to provide you with an overview of these requirements.

This document is neither a development nor a programming guide for internal applications; eJustice partners always have a total freedom within those fields. Nevertheless, in order to interact in a smooth, homogeneous and risk controlled way with a maximum of partners, eJustice partners must commit to comply with specifications, data format, and release processes described in this document.

In addition, our partners in the judiciary sector must also comply with the business rules of validation and integration of data within their own applications in order to minimize errors and incidents.

## eJustice document references

All the document references can be found in the technical library on the eJustice portal. These versions or any following versions can be used for the eJustice service.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Title** | **Version** | **Date** | **Author** |
| 1 | Cookbook eJustBox Publication | 1.0 | 15/04/2016 | Malik Alimoekhamedov |

## Service history

This chapter contains the list of changes applied to the service with respect to the previous version.

|  |  |  |
| --- | --- | --- |
| **Previous version** | **Previous release date** | **changes** |
| 1.0 | 12/04/2016 | Initial version. |

Remark: = “None” when the major version = 1

# Business and privacy requirements

For problems with applications working in the acceptance / integration environment, please contact:

Integration Support: contactcenter@eranova.fgov.be

Available from 9:00 am till 16:30 pm

For problems with applications in the production environment, please contact:

Production Support: contactcenter@eranova.fgov.be

Available from 7:00 am till 20:00 pm

# Global overview

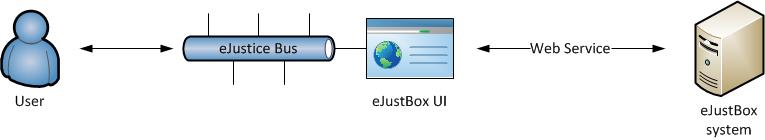


Figure 1: eJustBox Concept Diagram

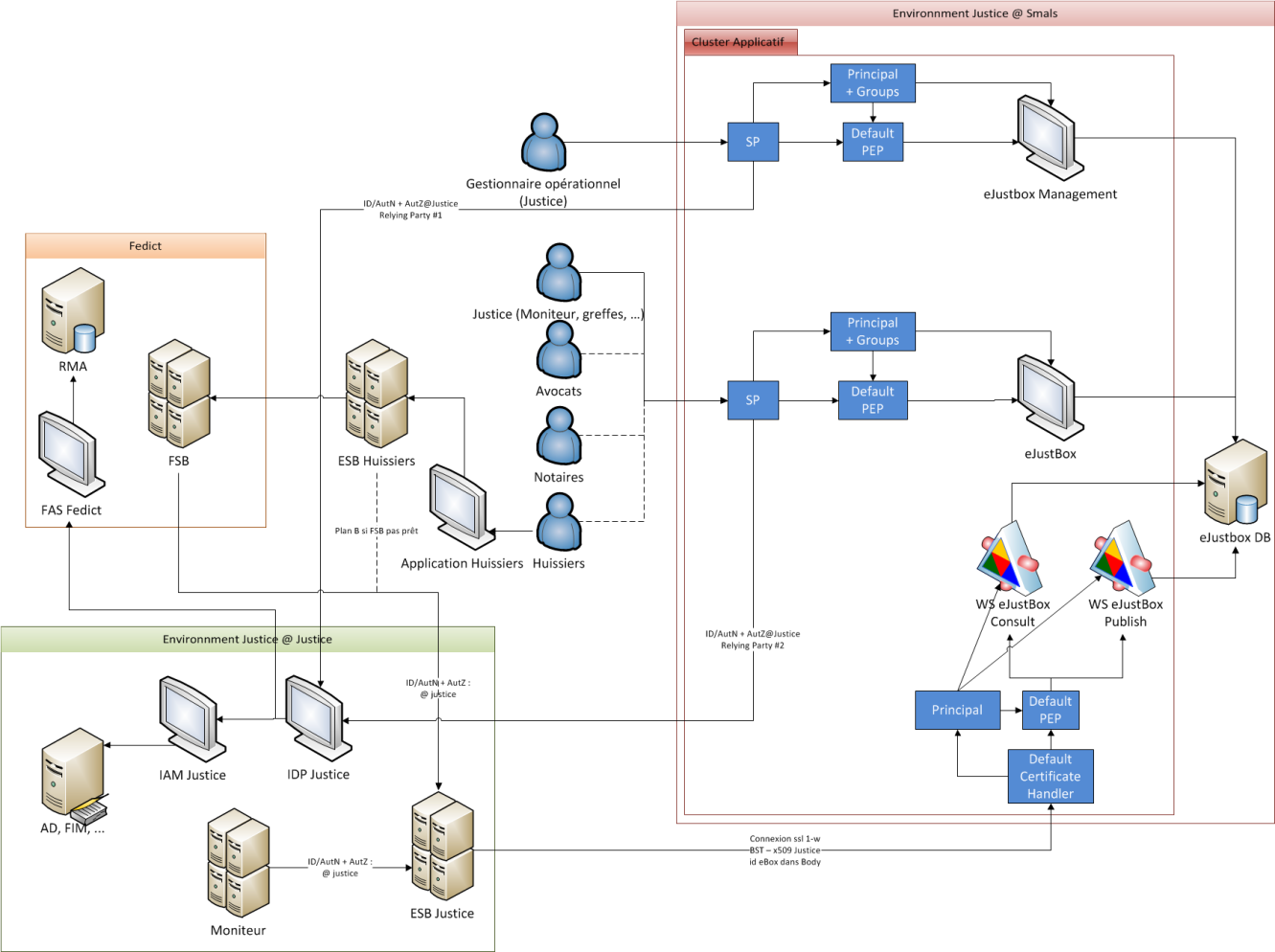


Figure 2: eJustBox interactions diagram

There will be no STS authentication process implemented (at least in the initial phase) for users of eJustBox. eJustice Bus will act as a trusted filter allowing only authorized users to access eJustBox’s user interface. This user interface will be wired to consultation (this document) and publication Web services. No message encryption will be provided by the system in first phase.

# Step-by-step

## Technical requirements

All the xml requests that are submitted to the web service must be encoded in the UTF-8 format.

## Process overview

### Web Service WSDL (Web Service Definition Language)

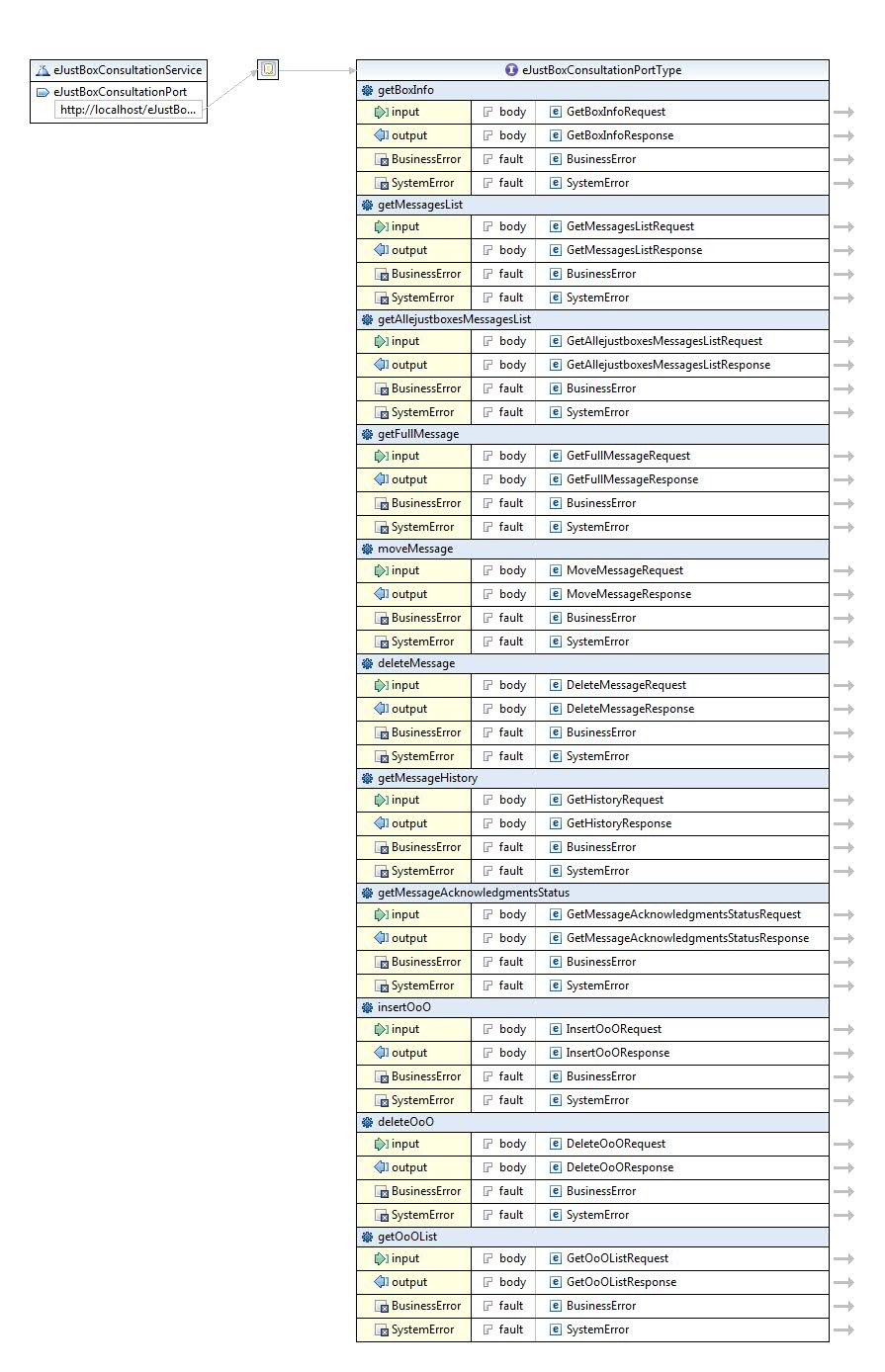


Figure 3: eJustBox Consultation WSDL

The important sections of the WSDL (Web Service Definition Language) of the Consultation Web Service are:

* The different methods: **getBoxInfo**, **getMessagesList**, **GetAllejustboxesMessagesList**, **getFullMessage**, **moveMessage**, **getMessageHistory, getMessageAcknowledgmentsStatus, InsertOoO, DeleteOoO, GetOoOList**.
* The types that are used by the methods.
* The fault messages are also defined for each method.

## eJustBox Consultation Web Service

### Lifetime of a message (To be further defined)

* When the expiration date of a message is reached AND it has already been placed in the recycle bin AND it has been read, the message is definitely removed from the application.
* When a message is older than 1 year (counted from publication date), it is definitely removed from the application, even if it hasn’t been read.

### Out-of-Office system

#### Introduction

This system enables the sender to know if one of the recipients is absent and to send his message to a substitute or substitutes, so the sender’s message can be treated. For example, justice of peace on holiday may want to ensure continuity of services for their clients. Users can automatically transfer their messages to another colleague responsible during their holidays thanks to the “Out-of-Office” system.

#### Out-of-Office restrictions

1. **A person cannot be defined as a substitute if he is already absent for this period (or part of the period).**
2. It is however possible to introduce an OoO while being a substitute for someone else.
3. A person may be substitute for several other persons.
4. The absent person and his substitutes are persons, not organizations.
5. A person is identified by its ID + quality.
6. There may be maximum 5 substitutes per OoO.
7. A maximum of 10 periods OoO may exist per eJustBox.
8. **OoO periods may not overlap.**
9. A period may not end later than J + 1Year.
10. A period can last up to 1 Year.
11. The end date is mandatory.

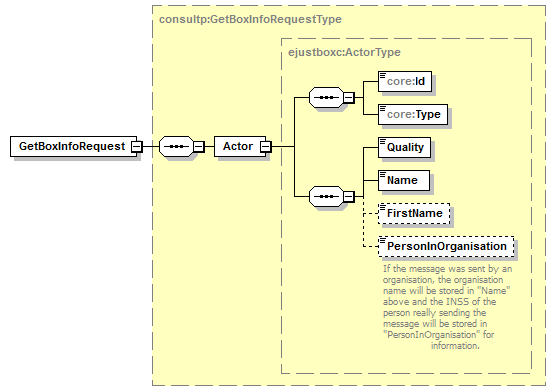
A WebService request contains one period and a maximum of 5 substitutes may be specified for that period.

The OoO activates automatically when the time period is reached.

### getBoxInfo Method

The *getBoxInfo* method allows a user to receive general information about his mailbox: the current used size of your mailbox, the maximum allowed size of your mailbox, and the number of message which couldn’t be received because the mailbox was full. These messages are still waiting to be placed in your mailbox. You need to clean your mailbox until the current size is lower than the max size. The messages will then enter into your mailbox.

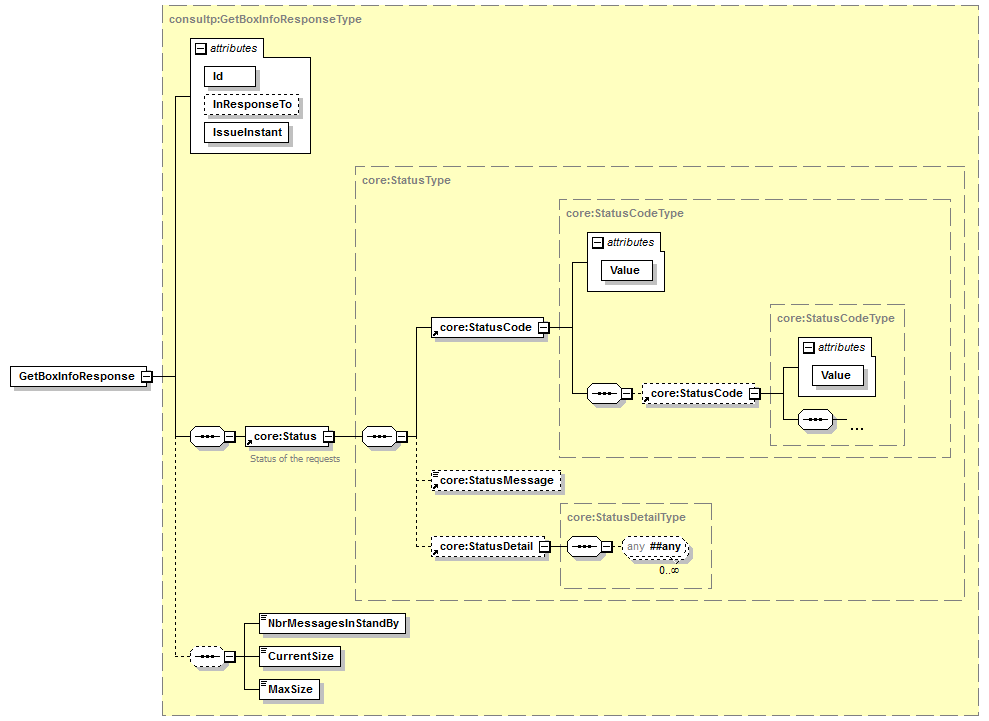
#### getBoxInfo Request



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| Id | The recipient’s identification number. This is a digital number representing an organization. String. |
| Type | The recipient’s ID type. String. |
| Quality | A *Quality* defines the recipient’s eJustBox. String. |

#### getBoxInfo Response

The response contains a success status code and general information on the eJustBox as explained below. The *Actor* element enables you to discover information on the eJustBox. Attention should be paid to *NbrMessagesInStandBy* larger than zero and if *CurrentSize* > *MaxSize*.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| NbrMessagesInStandBy | The number of messages that are in standby because the eJustBox is full. To consult these messages, the user has to delete others. |
| CurrentSize | The current size of the eJustBox expressed in bytes. |
| MaxSize | The maximum size of the eJustBox expressed in bytes. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:GetBoxInfoRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

</urn:GetBoxInfoRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns3:GetBoxInfoResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns3="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>100</Code>

<Message Lang="EN">SUCCESS</Message>

</Status>

<NbrMessagesInStandBy>0</NbrMessagesInStandBy>

<CurrentSize>58</CurrentSize>

<MaxSize>10485760</MaxSize>

</ns3:GetBoxInfoResponse>

</S:Body>

</S:Envelope>

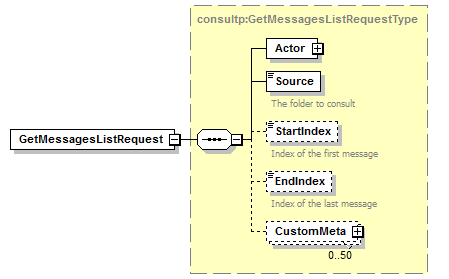
### getMessagesList Method

The *getMessagesList* method provides a list of messages for a specific folder of your eJustBox listed in order by date (most recent first, index “1”) A consequence could be that if a new message arrives between two consecutive queries, a message will be shown two times (message “1” becomes “2” , “2” becomes “3”, etc.). Ex. if you requested the messages between “1” and “100” and then the messages between “101” and “200”, then the message “100” would be the same as message “101”.

The content of the message is not yet returned by this method, but all the information needed is there in order to treat, filter, sort the messages. The sender, recipient, title message, publication date, message size, custom metas are all displayed for example.

#### getMessagesList Request

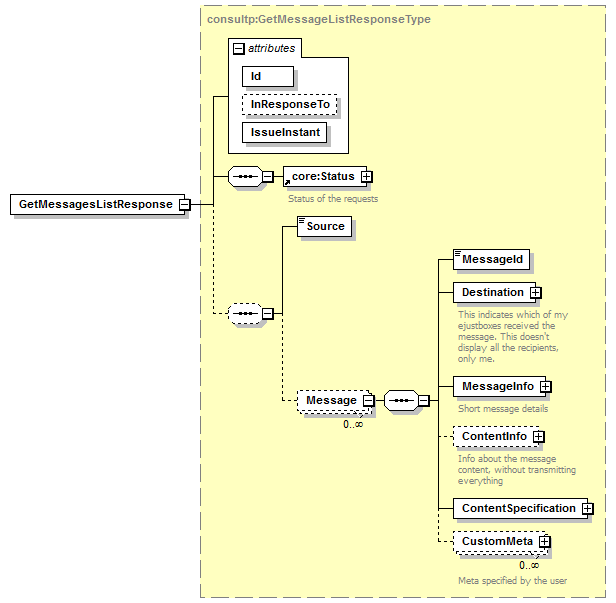
This method can only return 100 messages at a time; consequently, you must use it multiple times if necessary.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| Source | You can specify the folder specific to your request via the *Source* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. * “BININBOX” for messages moved from the inbox folder. * “BINSENTBOX” for messages moved from the sentbox folder. |
| StartIndex | Index of the first message (minimum 1). |
| EndIndex | Index of the last message (minimum 1). A maximum of 100 messages can be returned at once. EndIndex < StartIndex + 100 |
| CustomMeta | See section 5.3.12.8 |

#### getMessagesList Response

The response contains a success status code and as many *Message* elements as there are messages in the considered eJustBox. Each element contains all necessary information to treat the message without downloading each individual message.



| **Field name** | **Description** |
| --- | --- |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| Source | You can specify the folder specific to your request via the *Source* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. * “BININBOX” for messages moved from the inbox folder. * “BINSENTBOX” for messages moved from the sentbox folder. |
| Message | 0-to-more Message tag(s) describe(s) the eJustBox Message(s). Each Message is defined by the following:   * The MessageId that represents a unique message identification generated by the system and returned during publication and when calling the getMessagesList. String of 13 digits. * The *Destination* of the Message in terms of mailbox. * The *MessageInfo* that contains additional information about the Message such as publication date, size … (see section 5.3.12.13). * The *ContentInfo* of the Message such as title, mime type … (see section 5.3.12.6). * The *ContentSpecification* of the Message that contains information such as importance … (see section 5.3.12.7). * The *CustomMeta* of the Message that contains free Meta data specified by the user (see section 5.3.12.8 ) |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:GetMessagesListRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<Source>INBOX</Source>

<StartIndex>1</StartIndex>

<EndIndex>100</EndIndex>

</urn:GetMessagesListRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4:GetMessagesListResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>100</Code>

<Message Lang="EN">SUCCESS</Message>

</Status>

<Source>INBOX</Source>

<Message>

<MessageId>9Y0002LKM100K</MessageId>

<Destination>

<Id>77012824158</Id>

<Type>Type A</Type>

</Destination>

<MessageInfo>

<PublicationDate>2011-06-28+02:00</PublicationDate>

<ExpirationDate>2011-12-31+01:00</ExpirationDate>

<Size>46</Size>

</MessageInfo>

<ContentInfo>

<EncryptableINSSPerson>OTgwNTMwNDU3NDYyMQ0K</EncryptableINSSPerson>

<Title>News in eJustBox</Title>

<MimeType>text/plain</MimeType>

<HasAnnex>false</HasAnnex>

</ContentInfo>

<ContentSpecification>

<ContentType>NEWS</ContentType>

<IsImportant>false</IsImportant>

<IsEncrypted>true</IsEncrypted>

</ContentSpecification>

</Message>

</ns4:GetMessagesListResponse>

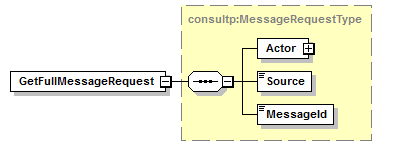
</S:Body>

</S:Envelope>

### getFullMessage Method

The *GetFullMessage* method is used to get the corresponding complete message and its content to a provided *MessageId*.

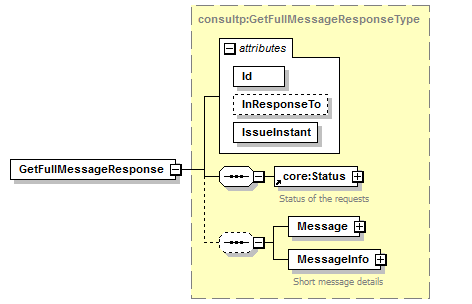
#### getFullMessage Request



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| Source | You can specify the folder specific to your request via the *Source* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. |
| MessageId | The *MessageId* is a unique message identification generated by the system and returned during the publication and when calling upon the getMessagesList. String of 13 digits. |

#### getFullMessage Response

The response contains the same information as already returned by GetMessagesList plus the Message content in the element *Message*.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| Sender | The *Sender* of the Message (see section 5.3.12.15). |
| Message | The *Message* itself (see section 5.3.12.12). |
| MessageInfo | Additional information about the Message (see section 5.3.12.13). |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:GetFullMessageRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<Source>INBOX</Source>

<MessageId>9Y0002LKLP004</MessageId>

</urn:GetFullMessageRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4:GetFullMessageResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>100</Code>

<Message Lang="EN">SUCCESS</Message>

</Status>

<Message MessageId="9Y0002LKLP004">

<PublicationId>InitialDoc</PublicationId>

<DestinationContext>

<Id>99999999964</Id>

<Type>Type B</Type>

<Quality>Quality B</Quality>

</DestinationContext>

<ContentContext>

<Content>

<Document>

<Title>Document in eJustBox</Title>

<EncryptableContent>

Some content…

</EncryptableContent>

</Document>

</Content>

<ContentSpecification>

<ContentType>DOCUMENT</ContentType>

<IsImportant>false</IsImportant>

<IsEncrypted>false</IsEncrypted>

</ContentSpecification>

<CustomMeta>

<Key>CategoryID</Key>

<Value>2</Value>

</CustomMeta>

<CustomMeta>

<Key>DocumentType</Key>

<Value>Scan</Value>

</CustomMeta>

</ContentContext>

</Message>

<MessageInfo>

<PublicationDate>2011-06-28+02:00</PublicationDate>

<ExpirationDate>2011-12-31+01:00</ExpirationDate>

<Size>12</Size>

</MessageInfo>

</ns4:GetFullMessageResponse>

</S:Body>

</S:Envelope>

### MoveMessage Method

The *MoveMessage* method enables the user to move a message from a *Source* (“INBOX”, “SENTBOX”, “BININBOX”, “BINSENTBOX”) to a *Destination* (“INBOX”, “SENTBOX”, “BININBOX”, “BINSENTBOX”). Only some combinations are allowed as explained below. You need to indicate if a message was received or sent by the concerned eJustBox. You can do this by looking in which folder the message is situated or thanks to a comparison between the currently connected user and the *Sender* and/or *Destination* element.

Allowed combinations:

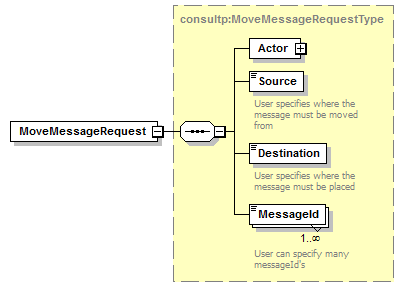
|  |  |
| --- | --- |
| **Source** | **Destination** |
| INBOX | BININBOX |
| SENTBOX | BINSENTBOX |
| BININBOX | INBOX |
| BINSENTBOX | SENTBOX |

Limitation:

Max 100 message / request

#### MoveMessage Request

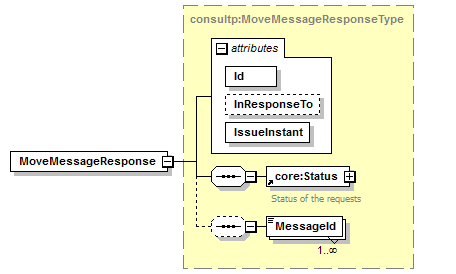
In Source specify where the message is currently situated and in Destination where the message must be moved to. In *MessageId* specify as many elements as there are messages to be moved. This method can only move 100 messages at a time; consequently, you must use it multiple times if necessary.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| Source | You can specify the folder specific to your request via the *Source* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. * “BININBOX” for messages moved from the inbox folder. * “BINSENTBOX” for messages moved from the sentbox folder. |
| Destination | You can specify the folder specific to your request via the *Destination* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. * “BININBOX” for messages moved from the inbox folder. * “BINSENTBOX” for messages moved from the sentbox folder. |
| MessageId | The *MessageId’s* corresponding to the message(s) to move. |

#### MoveMessage Response

The response contains a success status code or a Business Error as defined in chapter 8. The Business Error enables you to identify which messages where not successfully moved, even though all other have been moved successfully.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| MessageId | List of MessageId which could not be moved. Though, the other messages were successfully moved. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:MoveMessageRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<Source>INBOX</Source>

<Destination>BININBOX</Destination>

<MessageId>9Y0002LKM3006</MessageId>

<MessageId>9Y0002LKM3007</MessageId>

</urn:MoveMessageRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4:MoveMessageResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>813</Code>

<Message Lang="EN">Not all messages were moved successfully.

Please verify for each message that the Source and the MessageID are correct.

Also pay attention that a message in the recycle bin which was moved from the Inbox cannot be restored back to the Sentbox and vice versa.</Message>

</Status>

<MessageId>9Y0002LKM3006</MessageId>

</ns4:MoveMessageResponse>

</S:Body>

</S:Envelope>

### DeleteMessage Method

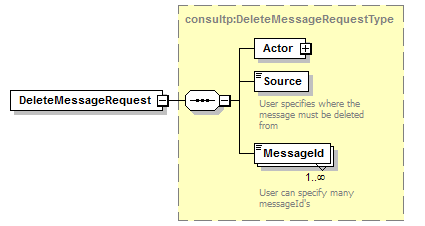
The DeleteMessage method enables the user to physically and definitely delete one or more messages from the inbox, sentbox or bin. This can be used to clean up the eJustBox when reaching the size limit. Be cautious when using this method. Best is to show a warning message to the user before deleting the messages.

Limitation:

Max 100 message / request

#### DeleteMessage Request

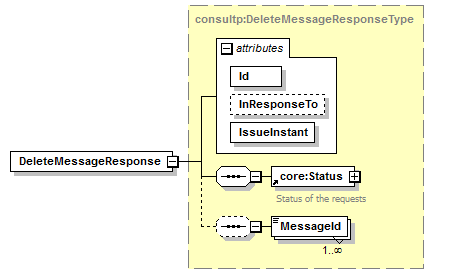
In MessageId you specify where the messages to delete from the bin or directly from the inbox or sentbox. This method can only delete 100 messages at a time; consequently, you must use it multiple times if necessary.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| Source | You can specify the folder specific to your request via the *Source* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. * “BININBOX” for messages moved from the inbox folder.   “BINSENTBOX” for messages moved from the sentbox folder. |
| MessageId | The *MessageId’s* corresponding to the message(s) to delete. |

#### DeleteMessage Response

The response contains a success status code or a Business Error as defined in chapter 8. The Business Error enables you to identify which messages where not successfully deleted, even though all other have been deleted successfully.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| MessageId | List of MessageId which could not be deleted. Though, the other messages were successfully deleted. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:DeleteMessageRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<Source>INBIN</Source>

<MessageId>9Y0002LKM3006</MessageId>

<MessageId>9Y0002LKM3007</MessageId>

</urn:DeleteMessageRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4:DeleteMessageResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>815</Code>

<Message Lang="EN"> One or more messages couldn’t be deleted. All other messages were successfully deleted. Please verify for each message that the MessageId is correct, and it is in the Inbox or in the recycle bin.</Message>

</Status>

<MessageId>9Y0002LKM3006</MessageId>

</ns4:DeleteMessageResponse>

</S:Body>

</S:Envelope>

### getHistory Method

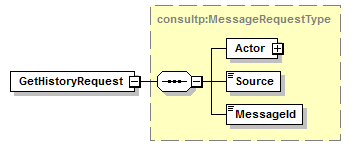
When a new message is sent and updates an old News item (by using the same PublicationId), the old news item is archived and the newer news item replaces the old one. This method enables you to request a list of the old version of that news item by using the *MessageId* attributed to the newer news item.

The *getHistory* method is used to get the older message versions of a “news” type message. The method returns a list of *MessageId’s* corresponding to the previous version of a “news item”, which enables the user to enter a *getFullMessage* on those *MessageId’s*.

The *getHistory* method can’t retrieve a history of a document.

#### getHistory Request

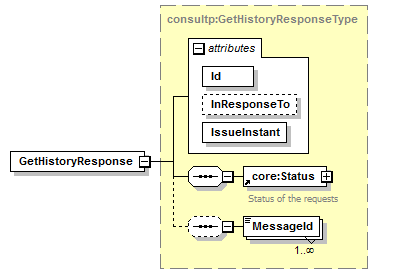
You can request the list of *MessageId* of a news item from you Inbox or from your SentBox. In *MessageId*, you can place the *MessageId* of the newer News item, or the *MessageId* of an old version of the same news item.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| Source | You can specify the folder specific to your request via the *Source* parameter.  The possible values are:   * “INBOX” for the inbox folder. * “SENTBOX” for the sentbox folder. |
| MessageId | The *MessageId* of the message to consult. The *MessageId* is a unique message identification generated by the system and returned during the publication and when calling upon getMessagesList. String of 13 digits. |

#### getHistory Response

The response gives you a group of *MessageId*’s which concern the same news item. You can then enter a GetFullMessage in order to retrieve the old news item if necessary.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| MessageId | List of MessageId’s from the older message versions. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:GetHistoryRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<Source>INBOX</Source>

<MessageId>9Y0002LKLP004</MessageId>

</urn:GetHistoryRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4:GetHistoryResponse xmlns:ns2="urn:be:fgov:eJustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>100</Code>

<Message Lang="EN">SUCCESS</Message>

</Status>

<MessageId>9Y0002LKLN001</MessageId>

</ns4:GetHistoryResponse>

</S:Body>

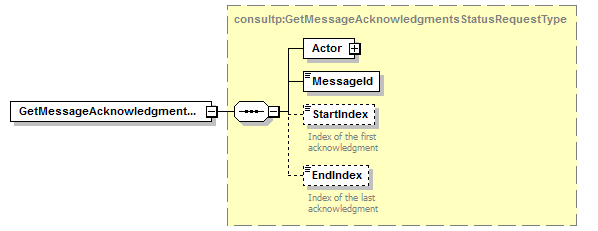
</S:Envelope>

### GetMessageAcknowledgmentsStatus Method

The *GetMessageAcknowledgmentsStatus* method is used to find out for a message that the user has sent which recipients have received, viewed or read the message and at what time.

#### GetMessageAcknowledgmentsStatus Request

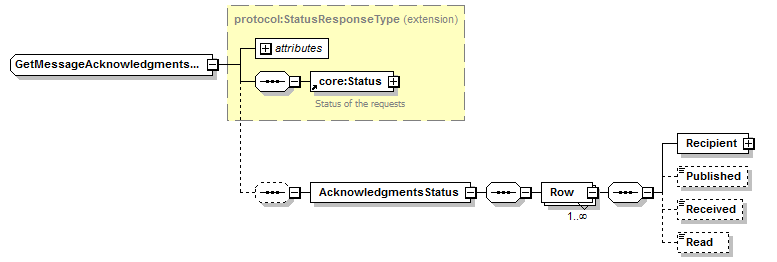
This method can only return 100 acknowledgements at a time; consequently, you must call upon it multiple times if necessary.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| MessageId | The *MessageId’s* of the message to consult. The *MessageId* is a unique message identification generated by the system and returned during the publication and when calling upon the getMessagesList. String of 13 digits. |
| StartIndex | Index of the first acknowledgment (minimum 1). |
| EndIndex | Index of the last acknowledgment (minimum 1). A maximum of 100 acknowledgments can be returned at once. EndIndex < StartIndex + 100 |

#### GetMessageAcknowledgmentsStatus Response

The response gives you information about your sent message: who received and read your message and at what time.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| AcknowledgmentsStatus | Contains a *Row* for each different *Recipient* of the message. Each *Row* contains the identification of the *Recipient* (Type *EJustboxIdentifier*, see the EJustboxIdentifier section), the time the message was published, the time the message was received (=viewed) by that *Recipient*, and the time the message was read by that *Recipient*. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:GetMessageAcknowledgmentsStatusRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<MessageId>9Y0002LKH020J</MessageId>

<StartIndex>1</StartIndex>

<EndIndex>100</EndIndex>

</urn:GetMessageAcknowledgmentsStatusRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4:GetMessageAcknowledgmentsStatusResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v3">

<Status>

<Code>100</Code>

<Message Lang="EN">SUCCESS</Message>

</Status>

<AcknowledgmentsStatus>

<Row>

<Recipient>

<Id>99999999964</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<User LastName="Simon" FirstName="Llew">99999999965</User>

</Recipient>

<Published>2011-12-17T09:30:47Z</Published>

<Received>2011-12-17T10:31:17Z </Received>

<Read>2011-12-17T09:30:47Z </Read>

</Row>

</AcknowledgmentsStatus>

</ns4:GetMessageAcknowledgmentsStatusResponse>

</S:Body>

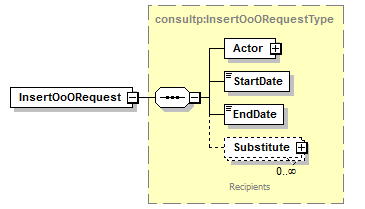
</S:Envelope>

### InsertOoO Method

The *InsertOoO* method is used to insert an Out-of-Office for the considered eJustBox. There are a number of rules and limitations to respect as described in section 5.3.1.2. If these rules are not respected a Business error is returned as detailed in section 8.2.

#### InsertOoO Request

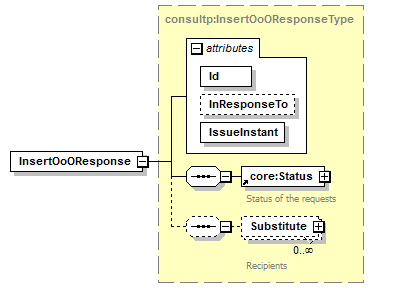
Only one OoO can be inserted by request, but multiple substitutes may be specified.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| StartDate | The *StartDate* of the Out-of-Office. This date is inclusive. Ex. 2013-07-03+02:00 |
| EndDate | The *EndDate* of the Out-of-Office. This date is inclusive. Ex. 2013-07-17+02:00 |
| Substitue | Optional. You may want to specify someone which is replacing you during your absence. The structure is as usual, detailed in the Substitute section. |

#### InsertOoO Response

The response is success without further details if everything went right. Otherwise the wrong or absent substitutes will be returned in the response as detailed below.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The Id element contains the OoOId attributed to this OoO by the system, which can later be used in DeleteOoO for example. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| Substitute | Contains a *Row* for each different *Substitute* which is incorrect, unknown, or absent (absence period of this substitute is then specified). See the Substitute section. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:InsertOoORequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<StartDate>2013-07-03+02:00</StartDate>

<EndDate>2013-07-17+02:00</EndDate>

<Substitute>

<Id>82351425106</Id>

<Type>Type B</Type>

<Quality>Quality B</Quality>

</Substitute>

</urn:InsertOoORequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4: InsertOoOResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>824</Code>

<Message Lang="EN">One or more substitutes cannot be chosen because they are absent.</Message>

</Status>

<Substitute>

<Id>82351425106</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<AbsentFrom>2013-07-03+02:00</AbsentFrom>

<AbsentTo>2013-07-17+02:00</AbsentTo>

</Substitute>

</ns4: InsertOoOResponse>

</S:Body>

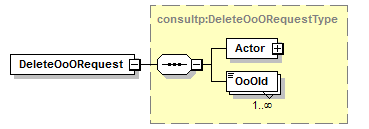
</S:Envelope>

### DeleteOoO Method

The *DeleteOoO* method is used to delete an Out-of-Office for the considered eJustBox. The Out-of-Office Id must be specified. This can be retrieved from InsertOoO, in the Id element or from GetOoOList.

#### DeleteOoO Request

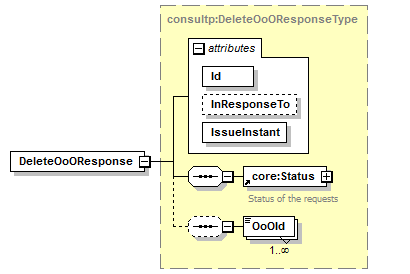
Only one OoO can be inserted by request, but multiple substitutes may be specified.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |
| OoOId | The OoOId from the Out-of-Offices that the user wants to delete. The OoOId’s can be retrieved from InsertOoO, in the Id element or from GetOoOList. |

#### DeleteOoO Response

The response is success without further details if everything went right. Otherwise the OoOId which could not be found will be returned in the response as detailed below. All other OoO which were found were deleted.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| Status | See section 5.3.12.18 |
| OoOId | Contains the OoOId’s which couldn’t be found and were not deleted. All others were deleted. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:DeleteOoORequest>

<Actor>

<Id>99999999964</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

<OoOId>1005</OoOId>

</urn:DeleteOoORequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4: DeleteOoOResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>840</Code>

<Message Lang="EN">One or more OoOId are invalid.</Message>

</Status>

<OoOId>1005</OoOId>

</ns4:DeleteOoOResponse>

</S:Body>

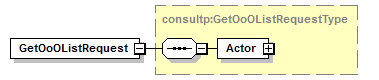
</S:Envelope>

### GetOoOList Method

The *GetOoOList* method is used to list all inserted Out-of-Office for the considered eJustBox. The list contains all OoO, past, active and future OoO. The OoOId returned can de further used to delete an OoO.

#### GetOoOList Request

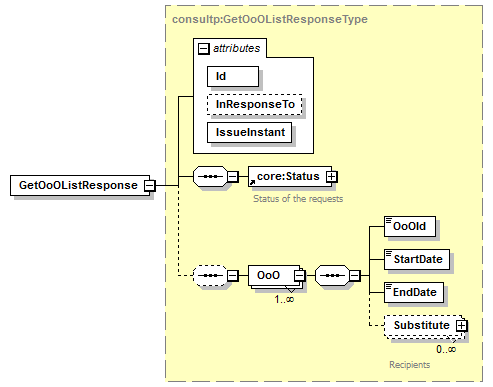
Only one OoO can be inserted by request, but multiple substitutes may be specified.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Actor | See section 5.3.12.3 |

#### GetOoOList Response

The response contains a success status code and as many *OoO* elements as there are Out-of-Office in the considered eJustBox. Each element contains all OoO data when it was inserted.



|  |  |
| --- | --- |
| **Field name** | **Description** |
| Id | The ticket number (*Id*) is attributed to the exchange request/response by the eJustice platform. This is used to identify the eJustice session. |
| InResponseTo | Id attribute of the request |
| IssueInstant | Date and time of the response. |
| OoOId | The OoOId assigned by the system to the Out-of-Office. |
| StartDate | The *StartDate* of the Out-of-Office. This date is inclusive. Ex. 2013-07-03+02:00 |
| EndDate | The *EndDate* of the Out-of-Office. This date is inclusive. Ex. 2013-07-17+02:00 |
| Substitue | Optional. The person which is replacing the absent person. The structure is as usual, detailed in the Substitute section. |

#### Example

Request:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<soapenv:Header/>

<soapenv:Body>

<urn:GetOoOListRequest>

<Actor>

<Id>982374923273</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

<Name>Name</Name>

</Actor>

</urn:GetOoOListRequest>

</soapenv:Body>

</soapenv:Envelope>

Response:

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns4: GetOoOListResponse xmlns:ns2="urn:be:fgov:ejustice:errors:service:v1" xmlns:ns4="urn:be:fgov:ejustice:ejustbox:consultation:protocol:v1">

<Status>

<Code>100</Code>

<Message Lang="EN">SUCCESS</Message>

</Status>

<OoO>

<OoOId>1000</OoOId>

<StartDate>2013-07-03+02:00</StartDate>

<EndDate>2013-07-17+02:00</EndDate>

<Substitute>

<Id>82351425106</Id>

<Type>Type A</Type>

<Quality>Quality A</Quality>

</Substitute>

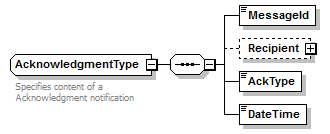
</OoO>

</ns4:GetOoOListResponse> </S:Body>

</S:Envelope>

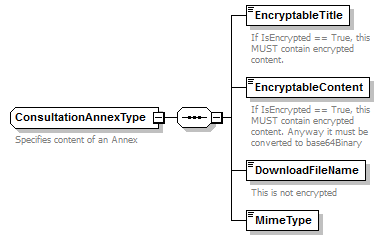
### Used Types

#### Acknowledgment



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| MessageId | The MessageId of the original message. |
| Recipient | The recipient receiving or reading the original message. This element in only present for an acknowledgment of type ‘RECEIVED or ‘READ’. |
| AckType | 3 acknowledgment types exist: ‘PUBLISHED’, ‘RECEIVED and ‘READ’:   1. An ack. ‘published’ => when the system is treating the original message 2. An ack. ‘Received’ => when the recipient has seen the original message (request getMessageList) 3. An ack. ‘Read’ => when the recipient has opened the original message (request getFullMessage) |
| DateTime | Date & Time when the acknowledgment was issued. Eg. <DateTime>2001-12-17T09:30:47Z</DateTime> |

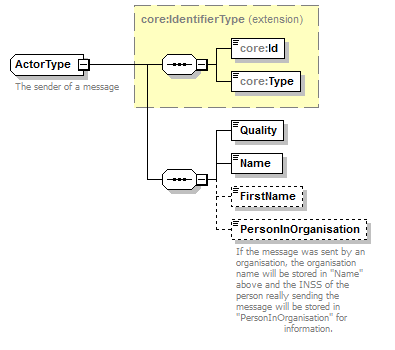
#### Annex



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| EncryptableTitle | An *Annex* has an *EncryptableTitle*, a human readable description of its intent (minimum 1, maximum 400 characters in binary form).  If *IsEncrypted* is true (see Section 5.3.12.7), the title must be encrypted. |
| EncryptableContent | Contains a SOAP reference to the attachment Eg. <EncryptableContent> [cid:4906633928</EncryptableContent](cid:4906633928%3c/EncryptableContent)>. Type is xsd:anyURI.  If *IsEncrypted* is true (see Section 5.3.12.7), the attachment must be encrypted. |
| DownloadFileName | E.g. “principal.pdf” (string minimum 1, maximum 255). |
| MimeType | Represents the mime type of the content. E.g. “application/pdf”,” text/plain”, “application/octet-stream” (string minimum 1, maximum 255). |

#### Actor

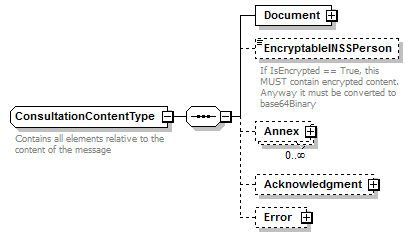
An *Actor* contains all the information on the eJustBox used for the request.



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| ID | Your eJustBox’s identification number. This is a digital number representing an organization |
| Type | Your eJustBox’s ID type. String. |
| Quality | Your eJustBox’s *Quality*. String. |
| Name | Actor’s Last Name |
| FirstName | Actor’s First Name |
| PersonInOrganisation | If the message was sent by an organization, the organization name will be stored in “Name” above and the INSS of the person really sending the message will be stored in “PersonInOrganisation” for information purposes |

#### Content

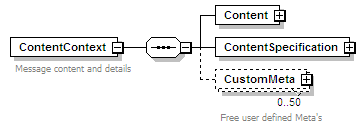
A *Content* contains the message content (a document or a news item), a concerned person’s INSS and zero-or-more annexes.



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| Document | A news item is now a special case of a Document. See section 5.3.12.10 |
| EncryptableINSSPerson | This optional field allows specifying an INSS number of a person concerned by the message content.  If *IsEncrypted* is true (see Section 5.3.12.7), the INSS number must be encrypted. |
| Annex | See section 5.3.12.2 |
| Acknowledgment | See section 5.3.12.1 |
| Error | See section 5.3.12.11 |

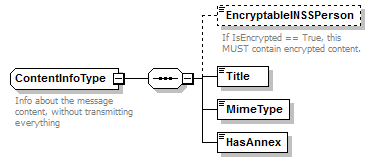
#### ContentContext

A *ContentContext* contains the message content and message details, as well as zero-or-more (50 maximum) free *CustomMeta*s. These *CustomMetas* can be freely specified by the user for internal usage. You can define a Key and a value for each *CustomMeta* (see 5.3.12.8).



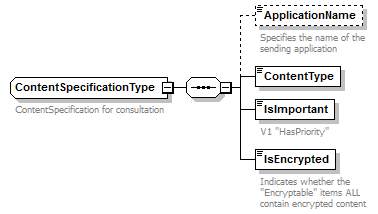
|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| Content | See section 5.3.12.4 |
| ContentSpecification | See section 5.3.12.7 |
| CustomMeta | See section 5.3.12.8 |

#### ContentInfo



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| EncryptableINSSPerson | This optional field allows to specify an INSS number of a person concerned by the message content.  If *IsEncrypted* is true (see Section 5.3.12.7), the content must be encrypted before being converted to xs:base64Binary. |
| Title | The *Title* of the message, a human readable description of its purpose (string minimum 1, maximum 400). |
| MimeType | Represents the mime type of the content. E.g. “application/pdf”,” text/plain”, “application/octet-stream” (string minimum 1, maximum 255). |
| HasAnnex | A flag (true or false) that indicates if the message has an *Annex*. |

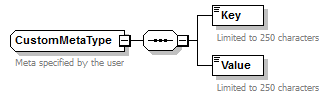
#### ContentSpecification



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| ApplicationName | The Application sending the message (optional, string minimum 1, maximum 25). |
| ContentType | The *Content Type* of the message (“DOCUMENT”, “NEWS”, “ACKNOWLEDGMENT”, “ERROR”). |
| IsImportant | Boolean (true or false) that indicates if the message is to be considered as important. |
| IsEncrypted | Boolean (true or false) that indicates if the content has been encrypted. |

#### CustomMeta

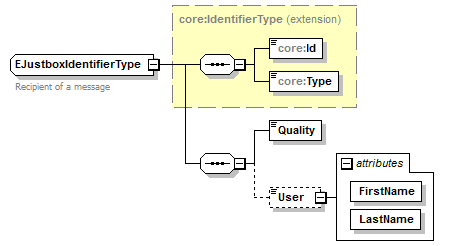
*CustomMeta* was introduced in order to enable the client to transport any Meta information relative to the message he wants. You can specify a maximum of 50 different pairs (key, value). The fields are limited each to 250 characters. Those *CustomMetas* will be transported from the sender to the recipient. You can for example add a CustomMeta for internal usage as “CategoryId, 17”, or “MessageContent, Judgment issued”.



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| Key | Alphanumeric string used as a key (string minimum 1, maximum 250). |
| Value | Alphanumeric string value corresponding to the *Key* (string minimum 1, maximum 250). |

#### EJustboxIdentifier

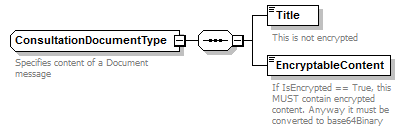
A *EJustboxIdentifierType* contains all the information on the recipient. Generally it is being referred to as *DestinationContext*.



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| ID | The recipient’s identification number. This is a digital number representing an organization. String. |
| Type | The recipient’s ID type. String. |
| Quality | A *Quality* defines the recipient’s eJustBox. String. |
| User | An optional *User* (*FirstName* and *LastName*) can be added in the destination context. In case of a publication to an organization, this field is used to specify a member of this organization (string minimum 1, maximum 100). String. |

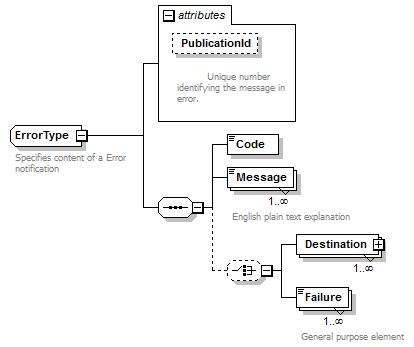
#### Document

Please note that a message will either contain a News item or a Document, not both.



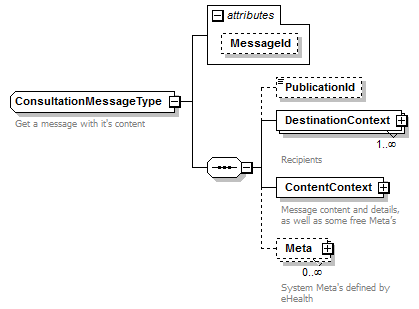
|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| Title | A Document has a *Title,* a human readable description of its intent (string minimum 1, maximum 400). |
| EncryptableContent | Contains a SOAP reference to the attachment Eg. <EncryptableContent> [cid:4906633928</EncryptableContent](cid:4906633928%3c/EncryptableContent)>. Type is xsd:anyURI.  If *IsEncrypted* is true (see Section 5.3.12.7), the attachement must be encrypted. |

#### Error



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| PublicationId | The PublicationId of the sent message (in error). |
| Code | The Error Code as defined in table 8.1. The number of possible error codes can evolve. |
| Message | A human readable English message. |
| Destination | The recipient causing an error because he could not be found for example. Check if the identifier and identifier type is correct. See the EJustboxIdentifier section for more information. |
| Failure | All-purpose field containing technical Ids as PublicationId or MessageId. Not currently used, but for future needs. |

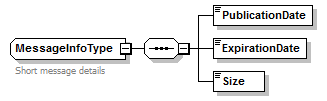
#### Message



|  |  |
| --- | --- |
| **Field name** | **Description** |
| MessageId | The *MessageId* is a unique message identification generated by the system and returned during the publication and when calling upon the getMessagesList. String of 13 digits. |
| PublicationId | The Id that the sender has used to publish the message. String, minimum 1, maximum 13. |
| DestinationContext | The *DestinationContext* is a complex type that contains information about the recipients.  This type is further detailed in the EJustboxIdentifier section.  A *Message* can have numerous *DestinationContext* (numerous recipients). |
| ContentContext | The *ContentContext* is a complex type that contains the message content.  This type is detailed in section 5.3.12.5. |
| Meta | **Currently, no meta information is defined.**  Additional system meta information can be defined by eJustice and used in convention with the client (for future needs). The type of meta information must be defined in the eJustBox system before it can be used (see section 5.3.12.14). |

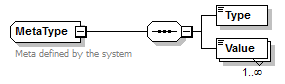
#### MessageInfo

A *MessageInfo* contains short details about the message.



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| PublicationDate | The *Date* on which the message has been published (format: yyyy-mm-dd+hh:mm). |
| ExpirationDate | The *Expiration Date* of the message (format: yyyy-mm-dd+hh:mm). Please find more information about the Expiration Date in Lifetime of a message. |
| Size | The *Size* of the message in bytes. |

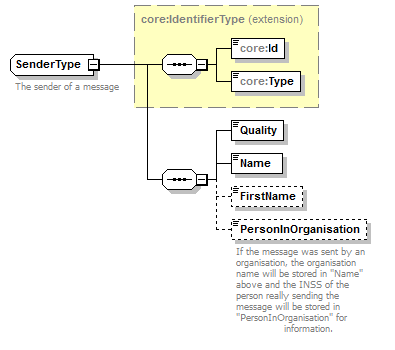
#### Meta



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| Type | The type of the meta information (string minimum 1, maximum 250). |
| Value | A list of *Values* for this *Type* (string minimum 1, maximum 250). |

#### Sender

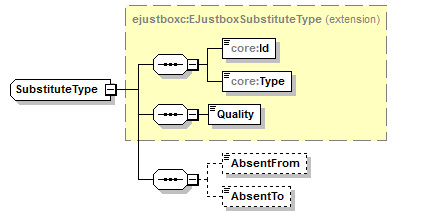
A *Sender* element contains all the information relative to the *Sender* of the message.



|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| ID | This is a digital number representing an organization. String. |
| Type | The Sender ID type. String. |
| Quality | A *Quality* defines the Sender eJustBox. String. |
| Name | Name of the Sender. String. |
| FirstName | FirstName of the Sender (optional). String. |
| PersonInOrganisation | If the message was sent by an organization, the organization name will be stored in "Name" above and the INSS of the person really sending the message will be stored in "PersonInOrganisation" for information. String. |

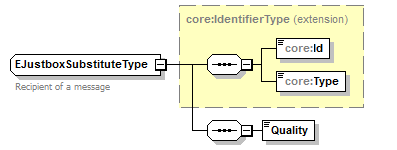
#### Substitute

A *Substitute* element contains all the information relative to the *Substitute* designated for a person.



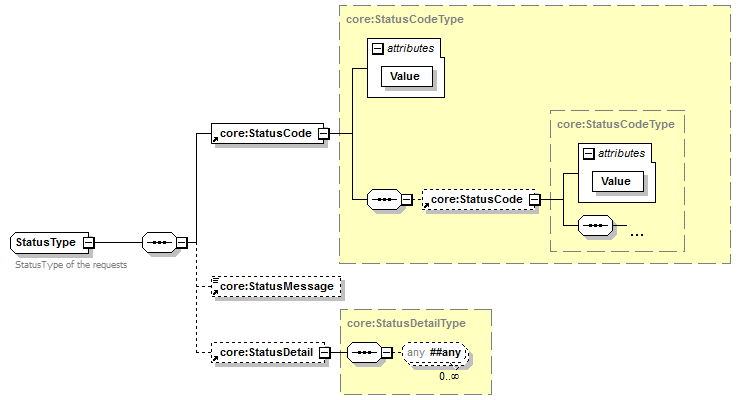
|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| ID | This is a digital number representing an organization. String. |
| Type | The Sender ID type. String. |
| Quality | A *Quality* defines the Sender eJustBox. String. |
| AbsentFrom | The *AbsentFrom* of the Out-of-Office. This date is inclusive. Ex. 2013-07-03+02:00 |
| AbsentTo | The *AbsentTo* of the Out-of-Office. This date is inclusive. Ex. 2013-07-17+02:00 |

#### EJustboxSubstitute



| **Field name** | **Descriptions** |
| --- | --- |
| Id | This is a digital number representing an organization. String. |
| Type | The Sender ID type. String. |
| Quality | A *Quality* defines the Sender eJustBox. String. |

#### Status



| **Field name** | **Descriptions** |
| --- | --- |
| Status | Status of the request |
| StatusCode | If no error has occurred during the transaction, the *Code* will be ‘100’.  In case of a business error:   * The *Code* is an error code that unequivocally identifies the problem (see chapter 7 for the possible values). |
| StatusMessage | If no error has occurred during the transaction, the *Message* ‘SUCCESS’.  In case of a business error:   * The *Message* will be a description of the error. Each Message has a *Lang* (language) characteristic :   + “FR” : French   + “NL” : Dutch   + “EN” : English   + “DE” : German   + “NA” : Not applicable   In case of technical errors, you will receive a Soap Fault message (see chapter 8). |
| StatusDetail | Detailes regarding the status |

# Risks and security

## SOAP with Attachments

The standard is used to transfer binary data via SOAP web services. SOAP with Attachments is an older alternative to MTOM. SwA is used in combination with WS-Security.

The SwA specification describes how to send binary data separated from the SOAP body using multipart MIME messages.

[*SOAP Messages with Attachments*](http://www.w3.org/TR/SOAP-attachments) is a link to the W3C specification of the standard:

<http://www.w3.org/TR/SOAP-attachments>

*Attachments Profile Version 1.0* is a link to the WS-I profile with clarifications and amendments to the SwA specification:

<http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html>

To keep a reference to the attachment from within the message payload, WS-I defined the simpleType swaRef:

<http://ws-i.org/profiles/basic/1.1/swaref.xsd>

The attachment data is no part of the message payload. So when you sign the SOAP body for secure transfer, the attachment data itself is not signed.

## Security

### Business security

In case the development adds an additional use case based on an existing integration, eJustice (contactcenter@eranova.fgov.be) must be informed at least one month in advance with a detailed estimate of the expected load. This will ensure an effective capacity management.

In case of technical issues on the web service, the partner may obtain support from the contact center that is responsible for this service.

In case eJustice finds a bug or vulnerability in its software, the partner is advised to update his application with the newest version of the software within 10 business days.

In case the partner finds a bug or vulnerability in the software or web service that eJustice has delivered, he is obliged to contact and inform eJustice immediately and he is prohibited to publish this bug or vulnerability in any case.

### Web Service Security Policy

N/A

### Security policies to apply

N/A

# Test and release procedure

## Procedure

This chapter explains the procedures for testing and releasing an application in acceptation or production.

### Initiation

If you intend to use the eJustice service, please contact [contactcenter@eranova.fgov.be](mailto:contactcenter@eranova.fgov.be). The Project department will provide you with the necessary information and mandatory documents.

### Development and test procedure

You have to develop a client (service consumer) in order to connect to our web service. Most of the required information to integrate is published in the technical library on the eJustice portal site.

In some cases eJustice provides you with a mock-up service or test cases in order for you to test your client (service consumer) before releasing it in the acceptance environment.

#### Create test cases

Rules to access the Publication services are the same in test and in production.

Access rules:

* + To use the Consultation services, the user must be part of one of the following profiles: **Quality A**

All test cases have to be configured by the eJustice development team.

### Release procedure

When the development tests are successful, you can request to access the eJustice acceptance environment.

From the moment you start the integration and acceptance tests, eJustice suggests testing during at least one month.

After the acceptance tests have been successfully completed, the partner sends his test results and performance results with a sample of the “eJustice request” and “eJustice answer” to the eJustice point of contact by email.

Then the eJustice platform and the partner agree on a release date. eJustice prepares the connection to the production environment and provides the partner with the necessary information. During the release day, the partner provides eJustice with feedback on the test and on the performance tests.

For further information and instructions, please contact: [contactcenter@eranova.fgov.be](mailto:contactcenter@eranova.fgov.be).

### Operational follow-up

Once in production, the partner using the eJustice service for one of its applications will always perform test first in the acceptance environment before releasing any adaptations of its application in production. In addition, he will inform eJustice on the progress and test period.

## Test cases

This section describes a step-by-step process to test the Consultation service.

eJustice recommends performing tests for all of the following cases:

1. Consult your eJustBox information with the method “getBoxInfo()”.
2. Based on your tests cases defined previously for the [Publication service](https://beconnected.belgium.be/alfresco/d/d/workspace/SpacesStore/924b00b9-0e9f-477a-9b26-357f48ecebab/request.testcases.ehBoxPub.webservice.xls), get the list of messages contained in your eJustBox with the method “getMessagesList()”. Execute this request on your inbox, sentbox and on your bin.
3. Pick one of the “MessageIds” returned from your messages list and use it to get the full message with the method “getFullMessage()”. Execute this request on a message from your inbox and sentbox.
4. Use again your “MessageID” to move your received message from your inbox to your recycle bin with the method “moveMessage()”.
5. Always consult the previous history of your message through a “MessageID” with the method “getMessageHistory()”.
6. Use a “MessageID” from your sentbox to consult the state of that message with “getMessageAcknowledgmentsStatus()”.

# Error and failure messages

## Error Element Status Codes

Mail delivery system error codes originating from the application:

These error codes first indicate a problem with the message and or its recipients. This table can evolve.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error code** | **Component** | **Description** | **Solution** |
| 700 | Error | Unknown technical error. | Call the Contact Center |
| 701 | Error | Business validation error. | Check all fields, especially the recipients. The publication id can be found under *PublicationId.* |
| 702 | Error | Duplicate publication id. | Chose a new publication id, and send again. The duplicate publication id can be found under *PublicationId*. |
| 703 | Error | One or more recipients are invalid. | Invalid recipients can be found under *Destination*. |

## Consultation Response Status Codes

Error codes originating from the eJustice platform:

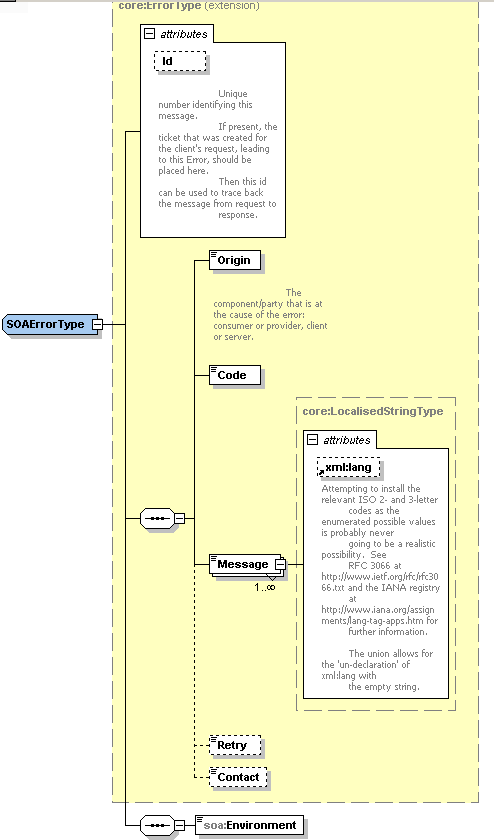
These error codes first indicate a problem in the sent arguments. This table can evolve.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error code** | **Component** | **Description** | **Solution** |
| 100 | \* | SUCCESS |  |
| 806 | \* / MessageId | The specified MessageID is invalid; please verify that the Source and the MessageID are correct and that you can access it. | Is the MessageId correct?  Is the message really present in “Source”? |
| 807 | \* / Start & EndIndex | EndIndex must be larger or equal to StartIndex; please correct StartIndex and EndIndex. | Is “EndIndex > StartIndex”? |
| 808 | \* / Start & EndIndex | A maximum of 100 messages can be returned by request; please correct StartIndex and EndIndex. | Is “EndIndex - StartIndex + 1 <= 100”? |
| 809 | GetMessage- AcknowledgmentsStatus/ MessageId | The specified MessageID is invalid; please verify that the MessageID is correct and that you are the sender. | Are you the sender of the message?  Is the MessageId correct? |
| 810 | \* / Actor | The specified actor is invalid; please verify the data and that you can access it. | Can you normally access that eJustBox? |
| 812 | MoveMessage/ Source &  Destination | You can not move a message from your Inbox to your Sentbox (even via recycle bin) and vice versa. | Is the message really present in “Source”? |
| 813 | MoveMessage/ Source &  Destination | Not all messages were moved successfully. Please verify for each message that the Source and the MessageID are correct. Also pay attention that a message in the recycle bin which was moved from the Inbox cannot be restored back to the Sentbox and vice versa. | Some messages where not found in the folder specified in “Source”.  Some messages can not be moved to “Destination”. |
| 815 | DeleteMessage | Not all messages were deleted successfully. Please verify for each message that the Source and MessageId are correct. | Not deleted Messages are listed in MessageId in the response. |
| 820 | InsertOoO | The period 01/07/2012 to 12/07/2012 is invalid because it overlaps another period. | Correct start or end date. |
| 821 | InsertOoO | The end of the period cannot be further than a year in the future. | Correct end date. |
| 822 | InsertOoO | The start date can't be after the end date. | Correct start date. |
| 823 | InsertOoO | The start date can't be in the past. | Correct start date. |
| 824 | InsertOoO | One or more substitutes cannot be chosen because they are absent. | Change one or more substitutes. |
| 825 | InsertOoO | The number of substitutes may not exceed 5. | Remove a substitute. |
| 826 | InsertOoO | The number of out of office for one eJustBox may not exceed 10. | Delete and out of office. |
| 827 | InsertOoO | One or more substitutes are unknown or not correct, please correct them. | Correct one or more substitutes. |
| 828 | InsertOoO | The user is unknown or not correct, please correct him. | Re-authenticate. |
| 829 | InsertOoO | A valid substitute is a person, not an organization. | Change one or more substitutes. |
| 830 | InsertOoO | A person cannot be substitute for himself. | Change one or more substitutes. |
| 840 | DeleteOoO | One or more OoOId are invalid. | Verify the OoOId’s. |

## Soap Fault Error Codes

They contain the following attributes:

|  |  |
| --- | --- |
| **Field name** | **Descriptions** |
| Id | Unique number identifying this message. If present, the ticket that was created for the client's request, leading to this error. Should be placed here. Then this id can be used to trace back the message from the request. |
| Origin | The component/party causing the error: consumer or provider, client or server. |
| Code | The Error Code |
| Message | A human readable message |
| Retry | An optional Boolean that indicates if it is worth resending the same Request. |
| Contact | An optional field specifying a contact description. |
| Environment | The eJustice environment in which the error occurs: integration, acceptation or production. |



### Schema Validation Errors

When invoking the Web Service, you must provide a valid XML.

Before executing any action, the eJustBox system verifies if the XML is valid by running a validation check towards the SendMessageRequest XSD.

If the validation fails, a SOAP Fault is returned with the following code and message:

|  |  |
| --- | --- |
| **Code** | **Message** |
| SOA-03006 | XSD compliance failure |

Example:

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

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<soapenv:Body xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" wsu:Id="id-6">

<soapenv:Fault>

<faultcode>soapenv:Client</faultcode>

<faultstring>SOA-03006</faultstring>

<detail>

<soa:SystemError xmlns:soa="urn:be:fgov:ejustice:errors:soa:v1" Id="5bbd8a2a-bb21-4cf8-99bc-8d52c18e2801">

<Origin>Consumer</Origin>

<Code>SOA-03006</Code>

<Message xml:lang="en">XSD compliance failure.</Message>

<soa:Environment>Production</soa:Environment>

</soa:SystemError>

</detail>

</soapenv:Fault>

</soapenv:Body>

</soapenv:Envelope>

### Technical Errors

Technical errors are errors inherent to the internal working of the eJustice Web Service. These errors can also occur if the token used to call the web service is not valid.

They contain the standard SOAP Fault attributes.

The table provides the different codes and messages returned in a SOAP fault message:

|  |  |
| --- | --- |
| Code | Message |
| SOA-00001 | An internal error has occurred. Please contact the Contact Center. |

This list can evolve.

Example:

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

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<env:Body xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">

<env:Fault>

<faultcode>soapenv:Server</faultcode>

<faultstring>SOA-00001</faultstring>

<detail>

<soa:SystemError Id="ec582704-d623-4b05-ab7f-98d5c9706dd1" xmlns:soa="urn:be:fgov:ejustice:errors:soa:v1">

<Origin>Server</Origin>

<Code>SOA-00001</Code>

<Message xml:lang="en">An internal error has occured. Please contact service desk.</Message>

<soa:Environment>Production</soa:Environment>

</soa:SystemError>

</detail>

</env:Fault>

</env:Body>

</soapenv:Envelope>

# List of sources

This list is a summary of all links used in the document referring to internet resources. eJustice is not responsible for these links.

|  |  |
| --- | --- |
|  | **Date** |
| http://www.w3.org/standards/xml/ | 24/03/11 |
| http://www.oasis-open.org/specs/ | 24/03/11 |
| <http://www.w3.org/TR/soap12-mtom/> | 30/09/11 |