Seals WS v.1.0
Cookbook
Version 2.2

This document is provided to you, free of charge, by the

eHealth platform

Willebroekkaai 38
38, Quai de Willebroek
1000 BRUSSELS

Table of contents

Table of contents 2

1. Document management 3
1.1 Document history 3

2. Introduction 4
2.1 Goal of the service 4
2.2 Goal of the document 4
2.3 eHealth platform document references 4
2.4 External document references 4
2.5 Service history 4

3. Support 5
3.1 Helpdesk eHealth platform 5
3.1.1 Certificates 5
3.1.2 For issues in production 5
3.1.3 For issues in acceptance 5
3.1.4 For business issues 5
3.2 Status 5
3.3 Authorization from sectoral committee 5

4. Global overview 6

5. Step-by-step 7
5.1 Technical requirements 7
5.1.1 Security policies to apply 7
5.1.2 WS-I Basic Profile 1.1 7
5.1.3 Tracing 7
5.2 Web service 8
5.2.1 Method Encoding Data 8
5.2.2 Method Decoding Data 10

6. Risks and security 13
6.1 Security 13
6.1.1 Business security 13
6.1.2 Web service 13

7. Test and release procedure 14
7.1 Procedure 14
7.1.1 Initiation 14
7.1.2 Development and test procedure 14
7.1.3 Release procedure 14
7.1.4 Operational follow-up 14

8. Error and failure messages 15
8.1 WS-I Basic Profile 1.1 15
8.2 Description of the possible SOAP fault exceptions. 15

1. Document management

1.1 Document history

Version	Date	Author	Description of changes / remarks
1.0	12/06/2012	eHealth platform	Initial version
2.0	06/11/2014	eHealth platform	Update of the information about encoding method (xsd, algorithms used and request example)
2.1	14/04/2021	eHealth platform	§ 5.1.2 WS-I Compliance § 5.1.3 Tracing
2.2	06/07/2022	eHealth platform	§ 3.2 Status (added) § 5.1.3 Tracing (update)

2. Introduction

2.1 Goal of the service

The web service Seals (formerly known as Codage V2) is the successor of Codage V1 and allows encoding and decoding data in a medical context.

2.2 Goal of the document

This document provides functional and technical information on how to call the Seals WS, which is provided by the eHealth platform.

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

This information should allow (the IT department of) an organization to develop and use the web service call. Some technical and legal requirements must be satisfied in order to allow the integration of the eHealth web services in client applications; this document was written to provide you with an overview of requirements that have to be met in order to correctly integrate the web services offered by the eHealth platform.

2.3 eHealth platform document references

All the document references can be found in the technical library on the portal of the eHealth platform[1]. These versions or any following versions can be used for the service.

ID	Title	Version	Date	Author
1	Glossary.pdf		01/01/2010	eHealth platform

2.4 External document references

ID	Title	Source	Date	Author
2	WS-I Basic Profile 1.1	***http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html***	24/08/2004	Web Services Interoperability Organization

2.5 Service history

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

Previous version	Previous release date	Changes
Codage WS v1.0	04/06/2009	This new version of the encryption WS is more secure and requires a permission for each functionality that is offered by the web service (encode-decode).

[1] ***https://ehealth.fgov.be/ehealthplatform***

3. Support

3.1 Helpdesk eHealth platform

3.1.1 Certificates

In order to access the secured eHealth platform environment you have to obtain an eHealth platform certificate, used to identify the initiator of the request. In case you do not have one, please consult the chapter about the eHealth Certificates on the portal of the eHealth platform

- ***https://www.ehealth.fgov.be/ehealthplatform/nl/ehealth-certificaten***
- ***https://www.ehealth.fgov.be/ehealthplatform/fr/certificats-ehealth***

For technical issues regarding eHealth platform certificates

- Acceptance: ***acceptance-certificates@ehealth.fgov.be***
- Production: ***support@ehealth.fgov.be***

3.1.2 For issues in production

eHealth platform contact centre:

- Phone: 02 788 51 55 (on working days from 7 am till 8 pm)
- Mail: ***support@ehealth.fgov.be***
- *Contact Form :*
  - ***https://www.ehealth.fgov.be/ehealthplatform/nl/contact*** (Dutch)
  - ***https://www.ehealth.fgov.be/ehealthplatform/fr/contact*** (French)

3.1.3 For issues in acceptance

Integration-support@ehealth.fgov.be

3.1.4 For business issues

- regarding an existing project: the project manager in charge of the application or service
- regarding a new project or other business issues: ***info@ehealth.fgov.be***

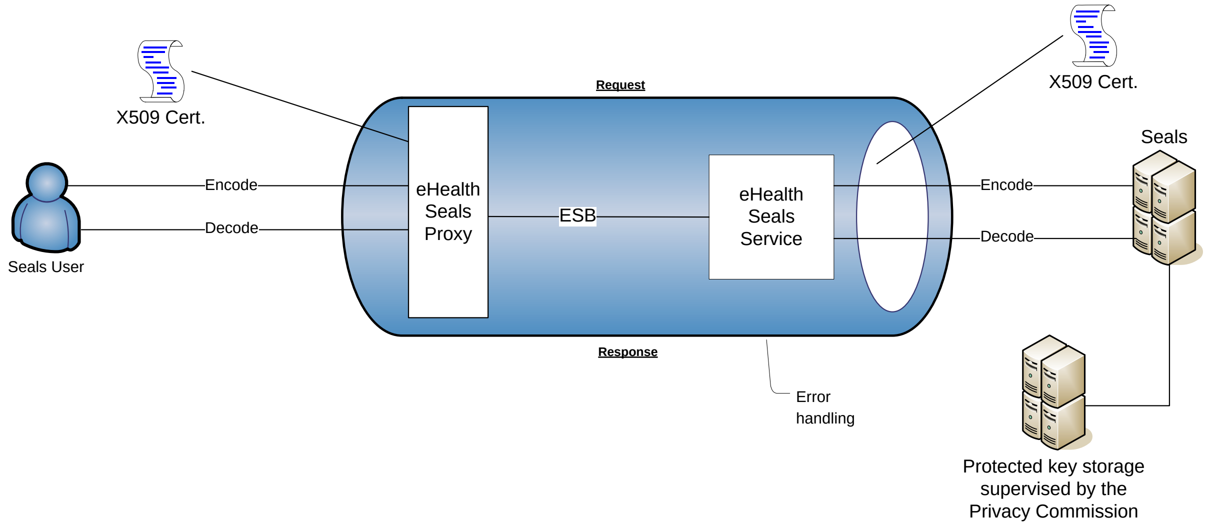
3.2 Status

The website ***https://status.ehealth.fgov.be*** is the monitoring and information tool for the ICT functioning of the eHealth services that are partners of the Belgian eHealth system.

3.3 Authorization from sectoral committee

For the encryption WS Seals, an authorization from the sector committee is required in order to use the "decoding" functionality of this WS.

4. Global overview



The Seals WS is composed of two methods:

- An encode method which allows encoding pieces of text (messages, significant data, documents …), following security checks (X509 certificate) and in accordance to well-defined rules for the submitting application
- A decode method which allows decoding previously coded pieces of text (messages, significant data, documents …), following security checks (X509 certificate) and in accordance to well-defined rules for the submitting application.

By default, the “encode method” can be used by anyone, whereas the “decode method” needs to be approved by the sector committee in order to be used. However, in order to enable, for instance, longitudinal research, the platform could preserve the link between the social security identification number (SSIN) and the code, but no other personal data. The preservation of this link will only be possible at the explicit request of the recipient if he provides a motivation for it and if the Health section of the Sector Committee of Social Security and of Health authorizes it.

Once a method is called, the encryption key attributed to the user is first retrieved in the protected key storage supervised by the Privacy Commission. Then the method is executed with this key.

5. Step-by-step

5.1 Technical requirements

5.1.1 Security policies to apply

We expect that you use SSL one way for the transport layer.

As WS security policy, we expect:

- A timestamp (the date of the request), with a time to live of one minute (if the message does not arrive during this minute, it shall not be treated).
- The signature with the certificate of
  - the timestamp, (the one mentioned above)
  - the body (the message itself)
  - and the binary security token: an eHealth certificate

  This will allow the eHealth platform to verify the integrity of the message and the identity of the message author.

A document explaining how to implement this security policy can be obtained at the eHealth platform.

5.1.2 WS-I Basic Profile 1.1

Your request must be WS-I compliant (Cfr External Ref). If it is not, you will receive one of the errors SOA-03001 – SOA-03003. (See Chapter 8 Errors).

5.1.3 Tracing

To use this service, the request SHOULD contain the following two http header values (see RFC

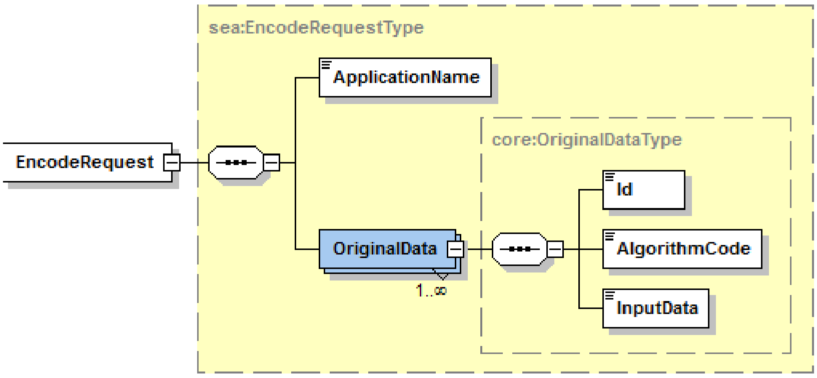
https://datatracker.ietf.org/doc/html/rfc7231#section-5.5.3):

1. User-Agent: information identifying the software product and underlying technical stack/platform. It MUST include the minimal identification information of the software such that the emergency contact (see below) can uniquely identify the component.
   a. Pattern: {minimal software information}/{version} {minimal connector information}/{connector-package-version}
   b. Regular expression for each subset (separated by a space) of the pattern: [[a-zA-Z0-9-\/]*\/[0-9azA-Z-_.]*
   c. Examples:
   User-Agent: myProduct/62.310.4 Technical/3.19.0
   User-Agent: Topaz-XXXX/123.23.X freeconnector/XXXXX.XXX
2. From: email-address that can be used for emergency contact in case of an operational problem.
   Examples:From: ***info@mycompany.be***

5.2 Web service

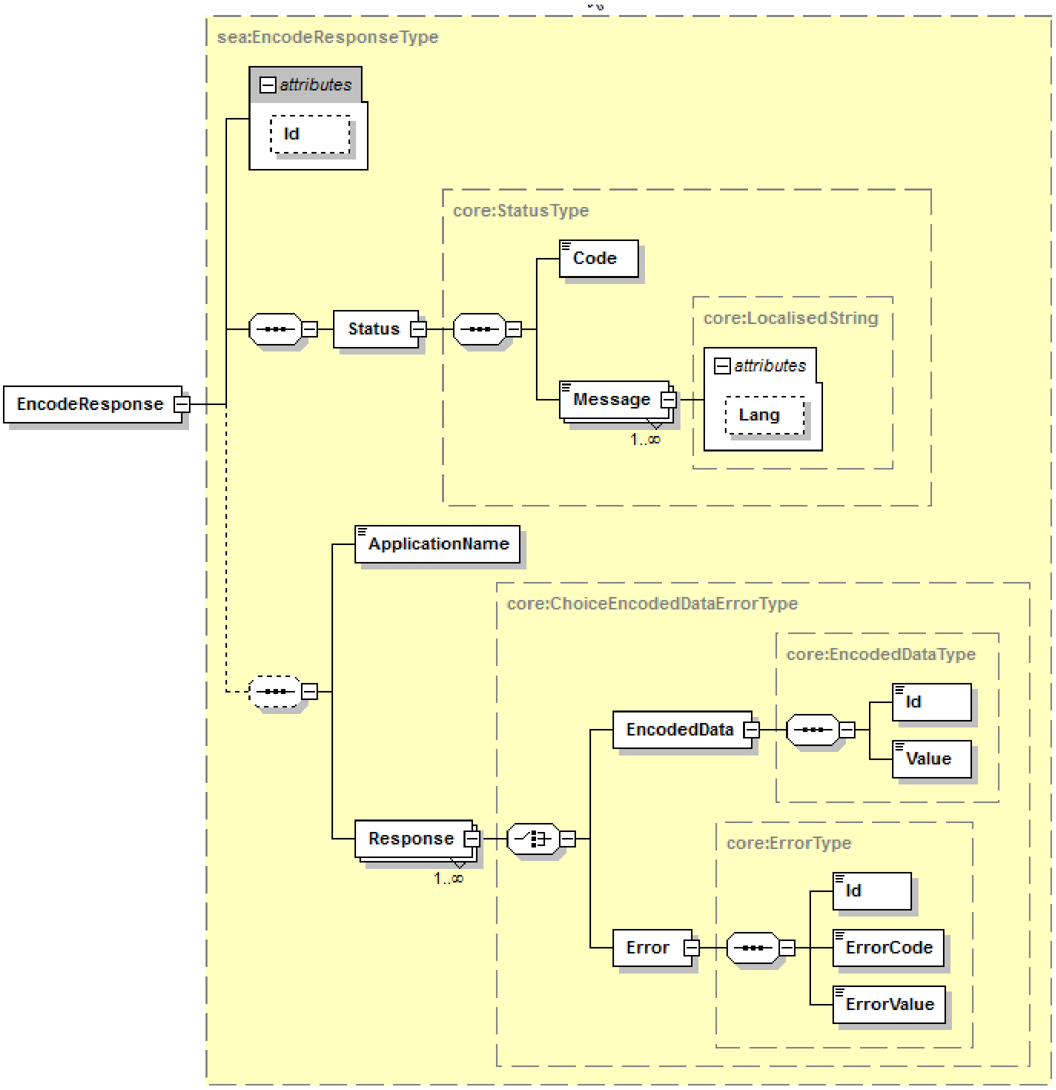
5.2.1 Method Encoding Data

5.2.1.1 Input arguments "EncodeRequest"



Field name	**Description**
ApplicationName	The name of the application for which encoding must be performed should be entered into the *ApplicationName* field.
OriginalData	The various values to be encoded should be entered into the *OriginalData* block. The *Id* field only serves to create a link between the request and the response, as the order of the items can/will **not** be guaranteed. The calling application is then also responsible for entry and management thereof. The application will never use this Id. In other words, there is no validation of the uniqueness of this value. However this value should contain at least one string character (so not only number) Depending on the type of data, the ***AlgorithmCode*** flag will contain the algorithm type used to make the encryption. For the **new** SEALS v1 users: - Algorithm "**AESECB**": This algorithm realizes a no randomized encryption. This means that each time a user encrypts a same input with the same key, he receives the same output. - Algorithm "**AESCCM**": This algorithm realizes a randomized encryption. This means that each time a user encrypts a same input with the same key, he receives a different output. For the **existing** SEALS v1 users: - Algorithm "**F**": This algorithm realizes a no randomized encryption. This means that each time a user encrypts a same input with the same key, he receives the same output. - Algorithm "**T**": This algorithm realizes a randomized encryption. This means that each time a user encrypts a same input with the same key, he receives a different output. For the **migrated** Codage v1 users - Algorithm "**AESECB_CR**": This algorithm realizes a no randomized encryption. This means that each time a user encrypts a same input with the same key, he receives the same output. - Algorithm "**AESCCM_CR**": This algorithm realizes a randomized encryption. This means that each time a user encrypts a same input with the same key, he receives a different output. The value to be encoded is entered into the ***InputData*** field.

5.2.1.2 Output arguments “EncodeResponse”



Field name	Description
Id	The number attributed to the request/reply by the eHealth platform.
Status	The *Status* block will contain a code and a message with ‘ language reference’. If no error has occurred during the transaction, the *Code* will be ‘200‘ and the *Message* ‘SUCCESS’. Otherwise: The *Code* will be an error code, which unequivocally identifies the problem (see section 8 “error and failure messages” for the possible values). A problem can be related to the infrastructure (availability of the web service ...) or content of the request. The *Message* will be a description of the error.
ApplicationName	The application for which the request is made is repeated in the ‘*applicationName*’ field.

Response	When all proceeds normally during encoding, the next request is sent. When many pieces of *OriginalData* are provided as input, there will be as many *encodedata/error* blocks as output. The Ids are repeated to allow the linking of the input to the output. The decoded value is naturally shown in the *outputdata* field.

5.2.1.3 Example

Request:

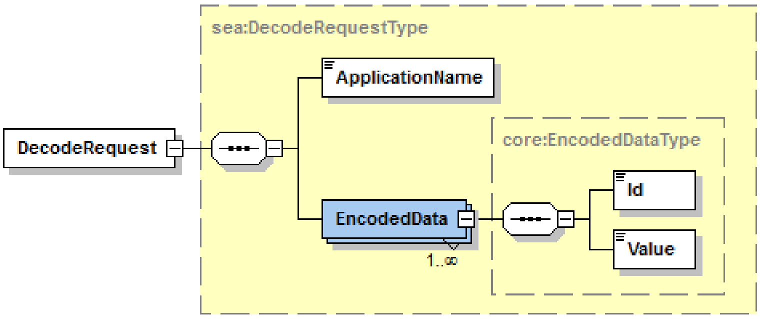
```
<urn:EncodeRequest>
        <ApplicationName>MONITORING</ApplicationName>
        <OriginalData>
                <Id>_1</Id>
                <AlgorithmCode>AESECB_CR</AlgorithmCode>
                <InputData>coucou</InputData>
        </OriginalData>
</urn:EncodeRequest>
```

Reply:

```
<ns5:EncodeResponse Id="9E0-000MYAY-00-9" xmlns:ns2="urn:be:fgov:ehealth:errors:soa:v1"
xmlns:ns3="urn:be:fgov:ehealth:errors:service:v1" xmlns:ns4="urn:be:fgov:ehealth:monitoring:protocol:v1"
xmlns:ns5="urn:be:fgov:ehealth:seals:protocol:v1">
        <Status>
                <Code>200</Code>
                <Message Lang="EN">Success</Message>
        </Status>
        <ApplicationName>MONITORING</ApplicationName>
        <Response>
                <EncodedData>
                        <Id>_4</Id>
                        <Value>O0Y7RnFsVzM4Ylc4eWduTjVSTCtGVnd6UT09DQo=</Value>
                </EncodedData>
        </Response>
</ns5:EncodeResponse>
```

5.2.2 Method Decoding Data

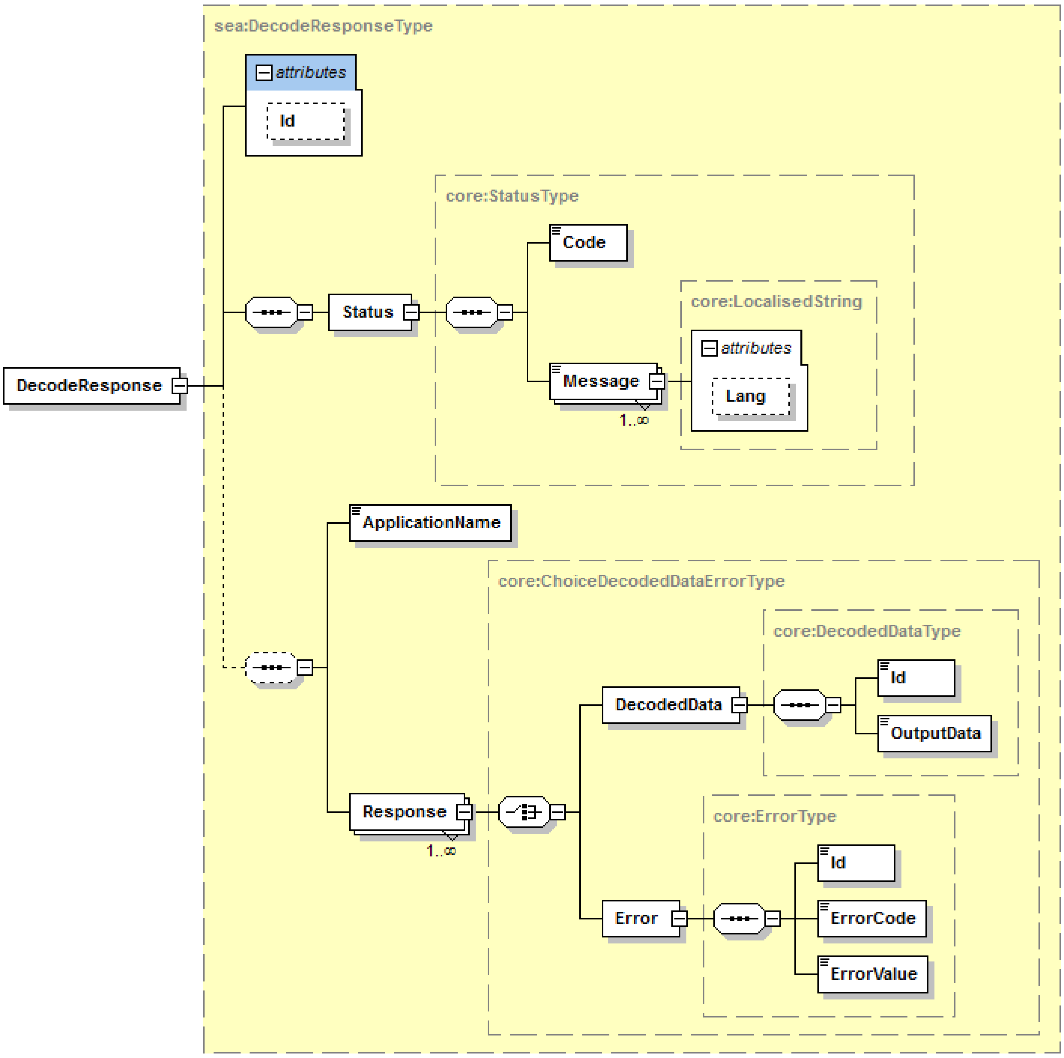
5.2.2.1 Input arguments "DecodeRequest"



Field name	Description
ApplicationName	The name of the application for which decoding must be performed should be entered into the *ApplicationName* field.

EncodedData	The various values to be decoded are entered into the E*ncodedData* block. The ***id*** field only serves to create a link between the *request* and the *response*, as the order of the items can/will **not** be guaranteed. The calling application is then also responsible for entry and management thereof. The application will never use this Id. In other words, there is no validation of the uniqueness of this value. However this value should contain at least one string character (so not only number) The value to be decoded is entered into the ***value*** field.

5.2.2.2 Output arguments "DecodeResponse"



Field name	**Description**
Id	The number attributed to the request/reply by the eHealth platform.
Status	The *Status* block will contain a code and a message with ' language reference'. If no error has occurred during the transaction, the *Code* will be '200' and the *Message* 'SUCCESS'. Otherwise: The *Code* will be an error code, which unequivocally identifies the problem (see section 8 "error and failure messages" for the possible values). A problem can be related to the

	infrastructure (availability of the web service ...) or content of the request. The *Message* will be a description of the error.
ApplicationName	The application for which the request is made is repeated in the '*applicationName*' field.
Response	When all proceeds normally during decoding, the next request is sent. When many pieces of *encodedData* are provided as input, there will be as many *decodedata/error* blocks as output. The ids are repeated to allow the linking of the input to the output. The decoded value is naturally shown in the *outputdata* field.

5.2.2.3 Example

Request:

```
<urn:DecodeRequest>
        <ApplicationName>MONITORING</ApplicationName>
        <EncodedData>
                <Id>_1</Id>
                <Value>O0Y7RnFsVzM4Ylc4eWduTjVSTCtGVnd6UT09DQo=</Value>
        </EncodedData>
</urn:DecodeRequest>
```

Reply:

```
<ns5:DecodeResponse Id="9E0-000MY9M-00-7" xmlns:ns2="urn:be:fgov:ehealth:errors:soa:v1"
xmlns:ns3="urn:be:fgov:ehealth:errors:service:v1" xmlns:ns4="urn:be:fgov:ehealth:monitoring:protocol:v1"
xmlns:ns5="urn:be:fgov:ehealth:seals:protocol:v1">
        <Status>
                <Code>200</Code>
                <Message Lang="EN">Success</Message>
        </Status>
        <ApplicationName>MONITORING</ApplicationName>
        <Response>
                <DecodedData>
                        <Id>_1</Id>
                        <OutputData>coucou</OutputData>
                </DecodedData>
        </Response>
</ns5:DecodeResponse>
```

6. Risks and security

6.1 Security

6.1.1 Business security

In case the development adds an additional use case based on an existing integration, the eHealth platform must be informed at least one month in advance with a detailed estimate of the expected load in order to ensure an effective capacity management.

In case of technical issues on the web service, the partner may obtain support from the contact center. (See section 3.2)

In case the eHealth platform 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 eHealth delivered, he is obliged to contact and inform the eHealth platform immediately and he is not allowed to publish this bug or vulnerability in any case.

6.1.2 Web service

Web service security used in this manner is in accordance with the common standards. Your call will provide:

- SSL one way
- Time-to-live of the message: one minute.
- Signature of the timestamp, body and binary security token. This will allow the eHealth platform to verify the integrity of the message and the identity of the message author.
- No encryption on the message.

7. Test and release procedure

7.1 Procedure

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

7.1.1 Initiation

If you intend to use the service of the eHealth platform, please contact ***info@ehealth.fgov.be***. The Project department will provide you with the necessary information and mandatory documents.

7.1.2 Development and test procedure

You have to develop a client in order to connect to our web service. Most of the required integration info to integrate is published in the technical library on the portal of the eHealth platform.

In some cases, the eHealth platform provides you with a mock-up service or test cases in order for you to test your client before releasing it in the acceptance environment.

7.1.3 Release procedure

When development tests are successful, you can request to access the acceptance environment of the eHealth platform.

From this moment, you start integration and acceptance tests. The eHealth platform suggests testing during minimum one month.

After successful acceptance tests, the partner sends his test results and performance results with a sample of "eHealth request" and "eHealth answer" by email to the point of contact at the eHealth platform.

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

For further information and instructions, please contact: ***integration-support@ehealth.fgov.be***.

7.1.4 Operational follow-up

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

8. Error and failure messages

There are different possible types of response:

- If there are no technical errors, responses as described in section 5 are returned.
- In the case of a technical error, a SOAP fault exception is returned (see table below).

If an error occurs, first please verify your request.

Following table contains a list of common system error codes for the eHealth Service Bus.

8.1 WS-I Basic Profile 1.1

Error code	Component	Description	Solution/Explanation
SOA-03001	**Consumer**	Malformed message	This is the default error for content related errors in case no more details are known.
SOA-03002	**Consumer**	Message must be SOAP	Message does not respect the SOAP standard.
SOA-03003	**Consumer**	Message must contain SOAP body	Message respects the SOAP standard, but body is missing.
SOA-03004	**Consumer**	WS-I compliance failure	Message does not respect WS-I standard.

8.2 Description of the possible SOAP fault exceptions.

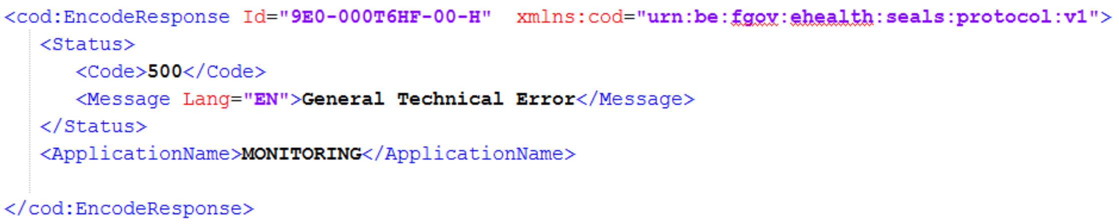
Error code	Component	Description	Solution/Explanation
SOA-00001	Unknown	Service error	This is the default error sent to the consumer in case no more details are known.
SOA-01001	Consumer	Service call not authenticated	From the security information provided • or the consumer could not be identified • or the credentials provided are not correct
SOA-01002	Consumer	Service call not authorized	• The consumer is identified and authenticated but is not allowed to call the given service.
SOA-02001	Provider	Service not available. Please contact service desk	• An unexpected error has occurred • Retries will not work • Service desk may help with root cause analysis
SOA-02002	Provider	Service temporarily not available. Please try later	• An unexpected error has occurred • Retries should work • If the problem persists service desk may help
SOA-03001	Consumer	Malformed message	This is default error for content related errors in case no more details are known.
SOA-03002	Consumer	Message must be SOAP	Message does not respect the SOAP standard

SOA-03003	Consumer	Message must contain SOAP body	Message respects the SOAP standard, but body is missing
SOA-03004	Consumer	WS-I compliance failure	Message does not respect the WS-I standard
SOA-03005	Consumer	WSDL compliance failure	Message is not compliant with WSDL in Registry/Repository
SOA-03006	Consumer	XSD compliance failure	Message is not compliant with XSD in Registry/Repository
SOA-03007	Consumer	Message content validation failure	From the message content (conform XSD): ● Extended checks on the element format failed ● Cross-checks between fields failed

You could also receive a business error. This is a list of possible business errors:

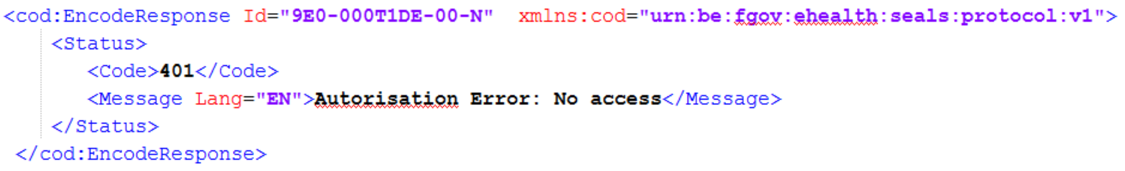
Method Encoding

- When there is a *technical problem* during the backend of our application, the following request will be received by the calling application.



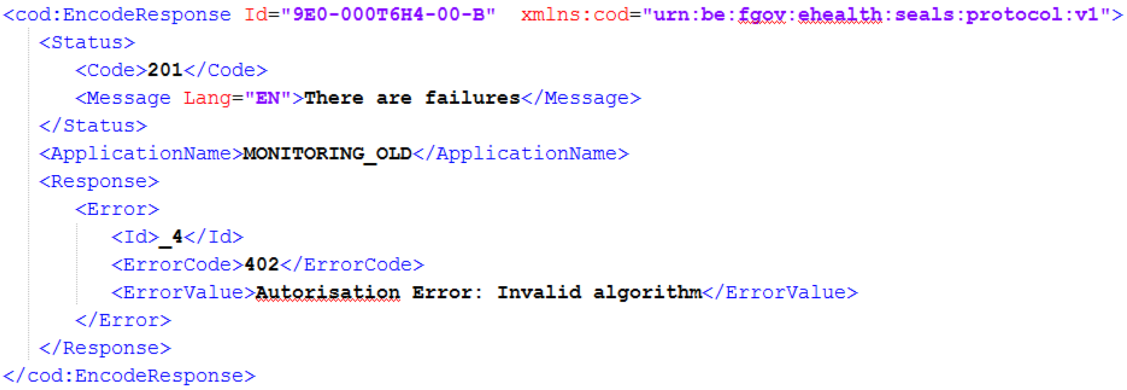
```
<cod:EncodeResponse Id="9E0-000T6HF-00-H"  xmlns:cod="urn:be:fgov:ehealth:seals:protocol:v1">
   <Status>
      <Code>500</Code>
      <Message Lang="EN">General Technical Error</Message>
   </Status>
   <ApplicationName>MONITORING</ApplicationName>

</cod:EncodeResponse>
```

- When a request is drawn for an application to which the calling application has no access, the following response is sent:



```
<cod:EncodeResponse Id="9E0-000T1DE-00-N"  xmlns:cod="urn:be:fgov:ehealth:seals:protocol:v1">
   <Status>
      <Code>401</Code>
      <Message Lang="EN">Autorisation Error: No access</Message>
   </Status>
</cod:EncodeResponse>
```

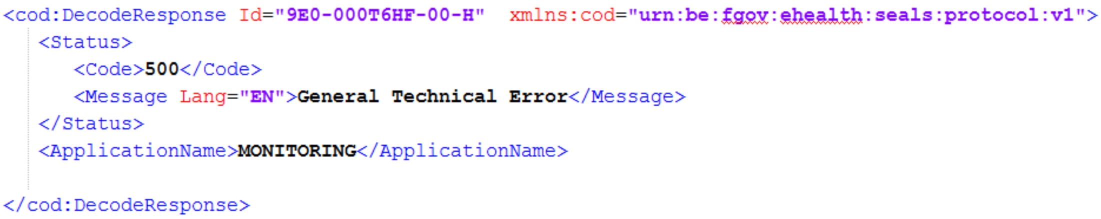
- When an unexpected error occurs while contacting the service and the cause of this error is not a technical failure of the backend, the following request will be sent as a response to the calling application. When you receive this type of error, you should contact the eHealth Support Team to find the cause and solve it as quickly as possible.



```
<cod:EncodeResponse Id="9E0-000T6H4-00-B"  xmlns:cod="urn:be:fgov:ehealth:seals:protocol:v1">
   <Status>
      <Code>201</Code>
      <Message Lang="EN">There are failures</Message>
   </Status>
   <ApplicationName>MONITORING_OLD</ApplicationName>
   <Response>
      <Error>
         <Id>_4</Id>
         <ErrorCode>402</ErrorCode>
         <ErrorValue>Autorisation Error: Invalid algorithm</ErrorValue>
      </Error>
   </Response>
</cod:EncodeResponse>
```

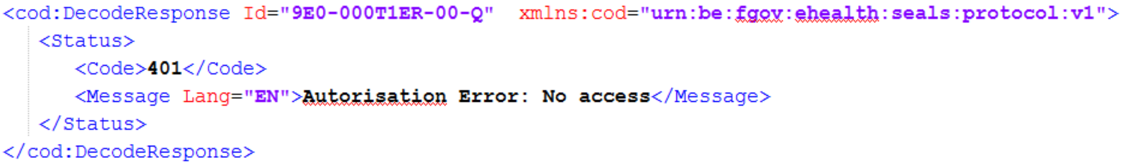
Method Decoding

- When a technical problem occurs during the backend of our application, the following request will be received by the calling application.



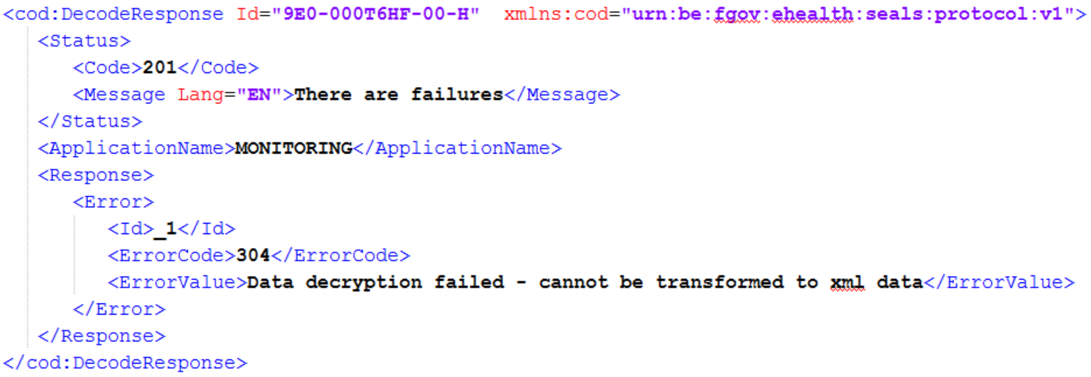
```
<cod:DecodeResponse Id="9E0-000T6HF-00-H"  xmlns:cod="urn:be:fgov:ehealth:seals:protocol:v1">
   <Status>
      <Code>500</Code>
      <Message Lang="EN">General Technical Error</Message>
   </Status>
   <ApplicationName>MONITORING</ApplicationName>

</cod:DecodeResponse>
```

- When a request is formulated for an application to which the calling application has no access, the following response is sent:



```
<cod:DecodeResponse Id="9E0-000T1ER-00-Q"  xmlns:cod="urn:be:fgov:ehealth:seals:protocol:v1">
   <Status>
      <Code>401</Code>
      <Message Lang="EN">Autorisation Error: No access</Message>
   </Status>
</cod:DecodeResponse>
```

- When an unexpected error occurs while contacting the service and the cause of this error is not a technical failure of the backend, the following request will be sent as a response to the calling application. When this type of error is received, you should contact the eHealth team to find the cause and to solve the problem as quickly as possible.



```
<cod:DecodeResponse Id="9E0-000T6HF-00-H"  xmlns:cod="urn:be:fgov:ehealth:seals:protocol:v1">
   <Status>
      <Code>201</Code>
      <Message Lang="EN">There are failures</Message>
   </Status>
   <ApplicationName>MONITORING</ApplicationName>
   <Response>
      <Error>
         <Id>_1</Id>
         <ErrorCode>304</ErrorCode>
         <ErrorValue>Data decryption failed - cannot be transformed to xml data</ErrorValue>
      </Error>
   </Response>
</cod:DecodeResponse>
```