Central Registry Traceability (CRT) - REST
Cookbook
Version 1.0

Updated by

This document is provided to you free of charge by the

eHealth platform

Willebroekkaai 38 – 1000 Brussel

38, Quai de Willebroeck – 1000 Bruxelles

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

Table of contents

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

1. Document management

1.1 Document history

Version*	Date	Author	Description of changes / remarks
0.1	29/10/2019	eHealth	Initial document
0.2	21/02/2020	eHealth	Table with business errors added
0.3	28/02/2020	eHealth	Added authentication & authorization information
0.4	05/06/2020	SMALS	Update according to new technical specs
1.0	16/09/2020	SMALS	Reviewed by eHealth

**Where 0.x means that this doc is in draft status and still need final validation (1.x) by lead specialist, architect & eHealth team.*

2. Introduction

2.1 Goal of the service

The goal of the Central Registry Traceability REST service (CRT REST) is to provide a lightweight service for managing notifications of implantations and removals of medical devices implanted in Belgium. These notifications contain the identifications of the patient, the installer of the implant, the prescriber of the implant, the pharmacist who delivered the implant and the organization where the implantation/removal took place.

In practice, it allows health care providers and their systems to :

- notify implantations/removals
- consult implantations/removals
  Concerning the consultations, it is possible to search the notification
  - by its identification
  - by patient
  - by implant
- delete notifications
- retrieve implant cards
  This is a PDF document which contains all the information about a notification and is stored by the application.

All actions are only possible if the health care provider has a therapeutic link with the patient.

CRT REST is designed for specialists who have a dedicated (hospital) information system. The details of the medical information remain in the patient's file at the hospital, and the registry represents a summary of this information.

The integration with the dedicated information system will make a submission transparent for the specialist.

2.2 Goal of the document

This document provides functional and technical information and allows an organization to integrate the Central Registry Traceability services in their own custom application.

This document will provide all the necessary elements to get you started developing. It explains in that context:

- the structure and content aspects of the possible requests and the replies
- technical and legal requirements (see point 3 & 5 of this document)

Examples will illustrate each of those messages. Also, a list of possible errors can be found in this document.

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

2.3 eHealth platform document references

On the portal of the eHealth platform, you can find all the referenced documents[1]. These versions or any following versions can be used for the eHealth platform service.

ID	Title	Version	Location	Author
1	Glossary.pdf	1.0	click here	eHealth
2	IAM (Identity & Access Management)	General infos	click here	eHealth
3	eHealth Services - Web Access	General infos	click here	eHealth
4	Système de cryptage end-to-end	General infos	click here	eHealth
5	END-TO-END ENCRYPTION Known recipient (KeyDepot) - REST	1.1	click here	eHealth
6	RCT REST API 1.0 (acpt)	1.0	click here	eHealth

2.4 External document references

ID	Title	Source	Location	Author
1	Central Registry Traceability V1 Cookbook	1.3	click here	eHealth/SMALS
2	Manuel destiné aux utilisateurs de RCT	1	click here	SMALS
3	Cookbook Central Registry for Traceability System to System (SOAP version)	2.1		SMALS

[1] ***www.ehealth.fgov.be/ehealthplatform***

3. Support

3.1 For issues in production

eHealth platform contact center:

- Phone: 02/788 51 55
- 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)

3.2 For issues in acceptance

Integration-support@ehealth.fgov.be

3.3 For business issues

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

3.4 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***

3.5 Others

3.5.1 I.AM Connect

OAuth 2.0 is a protocol for authorization and focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices.

OpenID Connect is a simple identity layer on top of the OAuth 2.0 protocol. It allows Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.I.AM Connect is eHealth's implementation of OpenID Connect.

To access CRT REST you'll need to obtain an "Access" token. To do so you'll need to request an I.AM Connect client in the realm "M2M" (machine-to-machine). This new client will allow you to request a new token that can be validated by CRT REST. Only token obtained from the realm "M2M" can be used to access CRT.

In short, eHealth I.AM Connect allows a client to access REST services for the eHealth domain. There is a form to complete that serve as base to register the client in a M2M realm. It must contain all information required to add the partner to the federation and to integrate with one of the eHealth environments.

The document is available here: *I.AM Connect - M2M Client registration - Version 1.0*

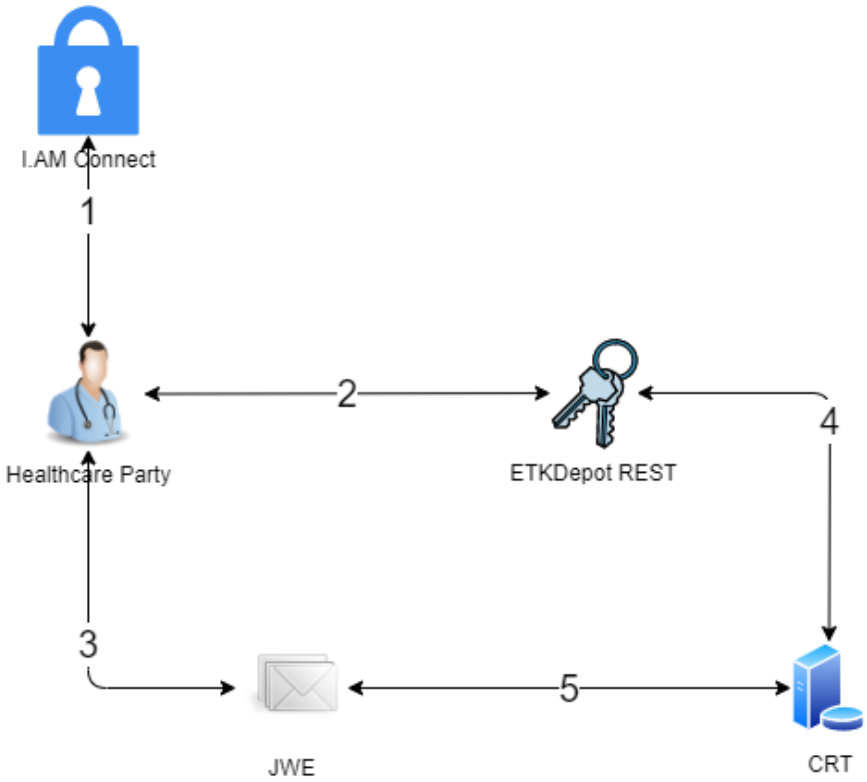
You can find more information about I.AM Connect and how to register on the eHealth portal page:

- ***https://www.ehealth.fgov.be/ehealthplatform/nl/service-architectures*** **(*Dutch*)**
- ***https://www.ehealth.fgov.be/ehealthplatform/fr/service-architectures (French)***

4. Global overview

4.1 High-level schema of the Central Registry Traceability functionalities

The system must guarantee the integrity and propriety of the transmitted messages and must ensure confidentiality measures taken when sending medical and private data. The sender of the message knows the recipient and the messages are sent directly in a synchronous way.



1. To use the endpoints, the client has to contact the I.AM Connect service to get a secure token containing his identification.
2. CRT REST endpoints require content signing and encryption. Therefore the client needs to sign its message using its private key then encrypt it using CRT public key
3. The client sends (CRUD) implant and removal notifications to CRT.
4. CRT will decrypt the message using its private key then validate the signature with the client's public key.
5. CRT provides the client answers

4.2 Endpoints

The REST interfaces are described with a JSON / Swagger API. The base URL's for the different environments are :

- ACC: https://api-acpt.ehealth.fgov.be/rct/v1
- PROD: https://api.ehealth.fgov.be/rct/v1

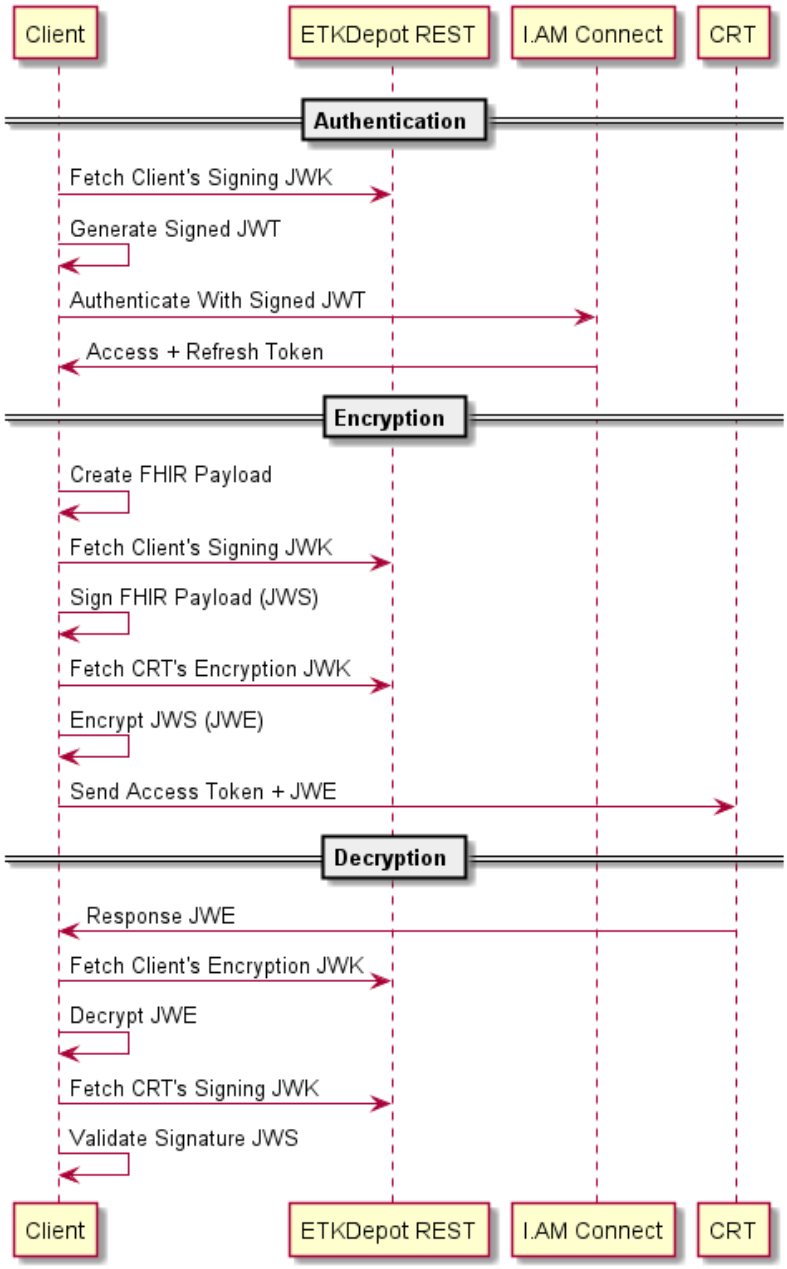
5. Step-by-step

5.1 Technical requirements

More information can be found in the eHealth documentation link :

- ***Welcome pack eHealth***

The next sequence diagram gives an overview how authentication/authorization as well as how the encryptions needs to be performed. Note that some of those calls might look unnecessary in the diagram. They are present still to put into evidence which key to use for which operation.



5.2 Process overview

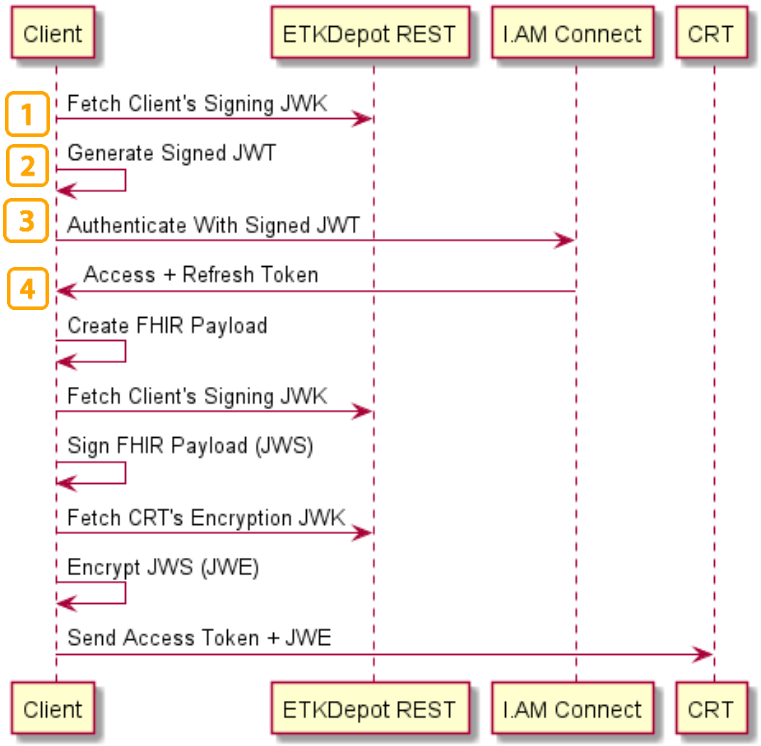
5.2.1 I.AM Connect

Only hospital can use CRT REST API.

For more details please refer to :

- ***eHealth Services - Web Access***

CRT uses the RFC-6749, Client Credentials Grant, as authentication mechanism. It implies that CRT's client will generate a signed JWT to authenticate themself to I.AM Connect. For machine to machine communication, eHealth provide a specific realm named **M2M**. Any client that wants to use CRT needs to request its own client in that realm.



1) Fetch Client's signing JWK ; Fetch own JWK to find the correct **kid** field.
2) Generate Signed JWT ; Generate the signed JWT using the client's private signing key.
3) Authenticate With Signed JWT ; Send the generated signed jwt to I.AM Connect.
4) Access + Refresh Token ; If all went well it should return an access token.

The signed JWT is a customized JWS serialized using the compact form. It is thus composed of 3 parts in base64 each separated by a dot:

I. Header
II. Body
III. Signature

It is really important to use the correct **kid** in the header part. Without this field I.AM Connect won't be able to find the correct public key to validate the signature.

Example:

eyJraWQiOiJTNDg0MTE5NTI4OTE4MDM5OTI1OCIsInR5cCI6IkpXVCIsImFsZyI6IlJTMjU2In0.eyJzdWIiOiJuaWhkaS1ob3NwaXRhbC03MTA4OTgxNS1jcnQiLCJhdWQiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2F1dGgvcmVhbG1zL00yTSIsIm5iZiI6MTU5MTYxNDg4MSwiZ3JhbnRfdHlwZSI6ImNsaWVudF9jcmVkZW50aWFscyIsImlzcyI6Im5paGRpLWhvc3BpdGFsLTcxMDg5ODE1LWNydCIsImV4cCI6MTU5MTcwMTI4MSwiaWF0IjoxNTkxNjE0ODgxLCJqdGkiOiIzOWNhYjNhNC01OTBjLTQ5MGEtODJjOC0yM2I5YjNjNDIxOTUifQ.FLgzyvHYbbMTrqOHAuAiv1nCaZIUv7WkzuzltkZBmGdRaisLTWqNpY8pZh9et0VsA95QXEnq9z685IDUG6vQxP9V6s7kbOMsxt1F6kChO8uyInctsw_jjOr0_zFU4JjVP6dWuPiLrJoNIenJ1CwIFu_f-UR8D66NiuWgnJ2DvboPqe1eBfFWqELAwYmaeb9m5aOXRSnIJhW1odDldd-qtnm7mYuoFYNesISGpLWYnE187PkjU9YpfQHk88M4naPHZXG1ResvLqhhGS628NUQqoYWqjhhazMkqaLIOqsvaqEDy73LErPX7-6wYy2CYvX52G0YJPbonNog7elqxmSBFg

Where in json format it looks like this:

Header:


```
{
 "kid": "S4841195289180399258",
 "typ": "JWT",
 "alg": "RS256"
}
```


Body:


```
{
 "sub": "nihdi-hospital-71089815-crt",
 "aud": "https://api-int.ehealth.fgov.be/auth/realms/M2M",
 "nbf": 1591614881,
 "grant_type": "client_credentials",
 "iss": "nihdi-hospital-71089815-crt",
 "exp": 1591701281,
 "iat": 1591614881,
 "jti": "39cab3a4-590c-490a-82c8-23b9b3c42195"
}
```


Sending the signed JWT to I.AM Connect is done using a POST with **Content-Type: application/x-www-form-urlencoded**. The actual body needs those attributes:

- grant_type: client_credentials
- client_assertion: The signed JWT
- scope: openid webaccess:cacerts rct:api manage
- client_assertion_type: urn:ietf:params:oauth:client-assertion-type:jwt-bearer

All those scopes are required and should be documented in I.AM Connect documentation.

Authenticate Request

Example of the full HTTP request:

POST /auth/realms/M2M/protocol/openid-connect/token HTTP/1.1[\r][\n] Content-Type: application/x-www-form-urlencoded[\r][\n] Accept: application/json[\r][\n] Content-Length: 949[\r][\n] Host: api-int.ehealth.fgov.be[\r][\n] Connection: Keep-Alive[\r][\n] User-Agent: Apache-HttpClient/4.5.11 (Java/1.8.0_212)[\r][\n] Accept-Encoding: gzip,deflate[\r][\n] [\r][\n]
grant_type=client_credentials&client_assertion=eyJraWQiOiJTNDg0MTE5NTI4OTE4MDM5OTI1OCIsInR5cCI6IkpXVCIsImFsZyI6IlJTMjU2In0.eyJzdWIiOiJuaWhkaS1ob3NwaXRhbC03MTA4OTgxNS1jcnQiLCJhdWQiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2F1dGgvcmVhbG1zL00yTSIsIm5iZiI6MTU5MTYxNTQ4MywiZ3JhbnRfdHlwZSI6ImNsaWVudF9jcmVkZW50aWFscyIsImlzcyI6Im5paGRpLWhvc3BpdGFsLTcxMDg5ODE1LWNydCIsImV4cCI6MTU5MTcwMTg4MywiaWF0IjoxNTkxNjE1NDgzLCJqdGkiOiIwNDNiYzQ1ZS03YTM4LTQwNGYtOThlYS04YWNjZDM3OGEwYzEifQ.P-YkMS5oOTkgedwsdiU8y70KvK_OpxZzc-GFhPFc4PBeGTYq_oAG2sbyTJYzW9DzG8vswzArtTnwaLElg3pxRSA2KoGgsaIqJNk4qOGAXk6AwwO6pXEa7bmJlAardZ6ZlbKT_SR6yQAW3kcrnqo9SwzZpXIQpiZ1B4Um1CK6niTc9_C_pjJDxFifZrggBtgAiQ5qkyh8RGQe9XqPzof--UUJLsBZe68L0rl09o7LekuVnGu1PqKXNAus9AiXhgPV6UompCKxjxhEpE2-wzUIB4e-kNAU7f1IVbpSGgt9Ob_giCVpNkQ14hxDJ8QVX0rR3qf7X0_f39cHqll9hJNyXA&scope=openid+webaccess%3Acacerts+rct%3Aapi+manage&client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer

I.AM Connect Response

Exemple of I.AM Connect response:

{
"access_token":
"eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJaX3VEb2ttaTJvMDl5VldybHprcE5PTW55MDJCMEhpS04tTlZJZHBUT0swIn0.eyJqdGkiOiIwNmY2NjMyMy01OWMwLTRhODYtODdkYS1kMjczNjQwYzZiNWIiLCJleHAiOjE1OTE2MTUxODQsIm5iZiI6MCwiaWF0IjoxNTkxNjE0ODg0LCJpc3MiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2F1dGgvcmVhbG1zL00yTSIsImF1ZCI6ImZhbWhwLXJjdC1hcGkiLCJzdWIiOiJkNTU0N2Y0Yy0zZWU1LTQ2ZTAtOWYzMi0yN2Q5YTIzNWNkNDgiLCJ0eXAiOiJCZWFyZXIiLCJhenAiOiJuaWhkaS1ob3NwaXRhbC03MTA4OTgxNS1jcnQiLCJhdXRoX3RpbWUiOjAsInNlc3Npb25fc3RhdGUiOiIwMDZkOGRmZS1iYzZhLTRlNzctOTMxOS01YzhmYjgzYjY0YWIiLCJhY3IiOiIxIiwicmVzb3VyY2VfYWNjZXNzIjp7ImZhbWhwLXJjdC1hcGkiOnsicm9sZXMiOlsibWFuYWdlIl19fSwic2NvcGUiOiJvcGVuaWQgcHJvZmlsZSBlbWFpbCB3ZWJhY2Nlc3M6Y2FjZXJ0cyByY3Q6YXBpIiwiZW1haWxfdmVyaWZpZWQiOmZhbHNlLCJjbGllbnRIb3N0IjoiODUuOTEuMTYyLjEiLCJjbGllbnRJZCI6Im5paGRpLWhvc3BpdGFsLTcxMDg5ODE1LWNydCIsIndlYmFjY2VzcyI6eyJjYWNlcnRzIjp7InNpZyI6eyJqa3UiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2V0ZWUvdjEvcHViS2V5cy9jYWNlcnRzL2p3a3M_aWRlbnRpZmllcj03MTA4OTgxNSZ0eXBlPU5JSElJLUhPU1BJVEFMJnVzZT1zaWcmYXBwbGljYXRpb25JZGVudGlmaWVyPVJDVFRFU1QifSwiZW5jIjp7ImprdSI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvZXRlZS92NhY2VydHMvandrcz9pZGVudGlmaWVyPTcxMDg5ODE1JnR5cGU9TklSUktSE9TUElUQUwmdXNlPWVuYyZhcHBsaWNhdGlvbklkZW50aWZpZXI9UkNUVEVTVCJ9fX0sInByZWZlcnJlZF91c2VybmFtZSI6InNlcnZpY2UtYWNjb3VudC1uaWhkaS1ob3NwaXRhbC03MTA4OTgxNS1jcnQiLCJjbGllbnRBZGRyZXNzIjoiODUuOTEuMTYyLjEiLCJ1c2VyUHJvZmlsZSI6eyJvcmdhbml6YXRpb25zIjpbeyJob3NwaXRhbCI6eyJuaWhpSI6IjcxMDg5ODE1In19XX0sImVtYWlsIjoic2VydmljZS1hY2NvdW50LW5paGRpLWhvc3BpdGFsLTcxMDg5ODE1LWNydEBwbGFjZWhvbGRlci5vcmcifQ.GLUP39mYv96dJgwux99e-DPCYb2OvcNLLJVNygvPLW3646BotYTkvsMNG59PSrjqznPF7ax23jhG5VYRRtVOIzw8MjllJcoXE7OxJbhQvV61re6hxTFdF1dDYXh4M9HmNOlWD4KIhi5r0LUiVUoajDv9wDJgHnYvmQS1leL0JkZ29HUj8tO1PfYjfS208k3xN3TXLMrsfiSmOWzgfDnMfuckvumAbUfvYwDhjz71Kg4wFP-DZy-kXLeE96NvOII7FXOmQYhAC70mcsr5nPDV8lID0JTPlK4jI_yXmd1Esmg1rfk5xdMVI_k5Jtl5UhdejAOErt6eYBug7enywRCNlw",
"expires_in": 300,
"refresh_expires_in": 1800,
"refresh_token":
"eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJlNDc2MjQyNi1mNmE1LTQyNTktYmZkZC04NTZiNmU3NDVkN2EifQ.eyJqdGkiOiI4ZGViN2MyOC05NWQyLTQ1NTYtYjkyOS03NDcyYjUxYjg4NjQiLCJleHAiOjE1OTE2MTY2ODQsIm5iZiI6MCwiaWF0IjoxNTkxNjE0ODg0LCJpc3MiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2F1dGgvcmVhbG1zL00yTSIsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvYXV0aC9yZWFsbXMvT

TJNIiwic3ViIjoiZDU1NDdmNGMtM2VlNS00NmUwLTlmMzItMjdkOWEyMzVjZDQ4IiwidHlwIjoiUmVmcmVzaCIsImF6cCI6Im5paGRpLWhvc3BpdGFsLTcxMDg5ODE1LWNydCIsImF1dGhfdGltZSI6MCwic2Vzc2lvbl9zdGF0ZSI6IjAwNmQ4ZGZlLWJjNmEtNGU3Ny05MzE5LTVjOGZiODNiNjRhYiIsInJlc291cmNlX2FjY2VzcyI6eyJmYW1ocC1yY3QtYXBpIjp7InJvbGVzIjpbIm1hbmFnZSJdfX0sInNjb3BlIjoib3BlbmlkIHByb2ZpbGUgZW1haWwgd2ViYWNjZXNzOmNhY2VydHMgcmN0OmFwaSJ9.G86eSL_gSw7jZ819J9fZIGORTof7fVCX-fBFC2mA3_k",
"token_type": "bearer",
"id_token":
"eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJaX3VEb2ttaTJvMDl5VldybHprcE5PTW55MDJCMEhpS04tTlZJZHBUT0swIn0.eyJqdGkiOiIxMzYwMWZjNy0yMGFhLTQ0NDQtYThjOC04NmJhOWEzOWExZDMiLCJleHAiOjE1OTE2MTUxODQsIm5iZiI6MCwiaWF0IjoxNTkxNjE0ODg0LCJpc3MiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2F1dGgvcmVhbG1zL00yTSIsImF1ZCI6Im5paGRpLWhvc3BpdGFsLTcxMDg5ODE1LWNydCIsInN1YiI6ImQ1NTQ3ZjRjLTNlZTUtNDZlMC05ZjMyLTI3ZDlhMjM1Y2Q0OCIsInR5cCI6IklEIiwiYXpwIjoibmloZGktaG9zcGl0YWwtNzEwODk4MTUtY3J0IiwiYXV0aF90aW1lIjowLCJzZXNzaW9uX3N0YXRlIjoiMDA2ZDhkZmUtYmM2YS00ZTc3LTkzMTktNWM4ZmI4M2I2NGFiIiwiYWNyIjoiMSIsImVtYWlsX3ZlcmlmaWVkIjpmYWxzZSwiY2xpZW50SG9zdCI6Ijg1LjkxLjE2Mi4xIiwiY2xpZW50SWQiOiJuaWhkaS1ob3NwaXRhbC03MTA4OTgxNS1jcnQiLCJ3ZWJhY2Nlc3MiOnsiY2FjZXJ0cyI6eyJzaWciOnsiamt1IjoiaHR0cHM6Ly9hcGktaW50LmVoZWFsdGguZmdvdi5iZS9ldGVlL3YxL3B1YktleXMvY2FjZXJ0cy9qd2tzP2lkZW50aWZpZXI9NzEwODk4MTUtmdHlwZT1OSUhJSS1IT1NQSVRBTCZ1c2U9c2lnJmFwcGxpY2F0aW9uSWRlbnRpZmllcj1SQ1RURVNUIn0sImVuYyI6eyJqa3UiOiJodHRwczovL2FwaS1pbnQuZWhlYWx0aC5mZ292LmJlL2V0ZWUvdjEvcHViS2V5cy9jYWNlcnRzL2p3a3M_aWRlbnRpZmllcj03MTA4OTgxNSZ0eXBlPU5JSElJLUhPU1BJVEFMJnVzZT1lbmMmYXBwbGljYXRpb25JZGVudGlmaWVyPVJDVFRFU1QifX19LCJwcmVmZXJyZWRfdXNlcm5hbWUiOiJzZXJ2aWNlLWFjY291bnQtbmloZGktaG9zcGl0YWwtNzEwODk4MTUtY3J0IiwiY2xpZW50QWRkcmVzcyI6Ijg1LjkxLjE2Mi4xIiwidXNlclByb2ZpbGUiOnsib3JnYW5pemF0aW9ucyI6W3siaG9zcGl0YWwiOnsibmloaWkiOiI3MTA4OTgxNSJ9fV19LCJlbWFpbCI6InNlcnZpY2UtYWNjb3VudC1uaWhkaS1ob3NwaXRhbC03MTA4OTgxNS1jcnRAcGxhY2Vob2xkZXIub3JnIn0.Wl0pfD2eTFayxZgg7KzJsECoNbCEH2Hm1nHvL7z6BV_noh35SMghZ3Yd-b4-6Sc8PmWDGIUiB4he1SJ4NphxKl4JfbyZDOT4RlE7VT_Q_0xTXMaWpPWkzZOmD5g5aagkvEcEjtGM3_ejAOcQ1ecOocoqvvZZeDageiwZg0lZkVtYZjuLvJ3Ebe3SVQN-2hOEzXINCAsAu1m5uCUrln5IxsuSKg2A5gcD8aSWulESJL5nzAuNQJGwC510feEaabIRT1DVZnJqHQM1WgJpoyzoxSR8DhZeg-btxouQje9PtR0Iid1x1G8E4w4gZ_xnBtaIYtNzPWW67-Zg5uVYbTZe3Q",
"not-before-policy": 0,
"session_state": "006d8dfe-bc6a-4e77-9319-5c8fb83b64ab",
"scope": "openid profile email webaccess:cacerts rct:api"
}

Required Access Token Role:

The access token should absolutely have those attributes to work with CRT.

{
"resource_access": {
"famhp-rct-api": {
"roles": [
"manage"
]
}
}
}

Required KeyDepot Information:

{

```
 "webaccess": {
  "cacerts": {
   "sig": {
    "jku": "https://api-int.ehealth.fgov.be/etee/v1/pubKeys/cacerts/jwks?identifier=71089815&type=NIHII-HOSPITAL&use=sig&applicationIdentifier=RCTTEST"
   },
   "enc": {
    "jku": "https://api-int.ehealth.fgov.be/etee/v1/pubKeys/cacerts/jwks?identifier=71089815&type=NIHII-HOSPITAL&use=enc&applicationIdentifier=RCTTEST"
   }
  }
 }
}
```


Required Hospital Information:


```
{
 "userProfile": {
  "organizations": [
   {
    "hospital": {
     "nihii": "71089815"
    }
   }
  ]
 }
}
```

5.2.2 JOSE

Currently, a client which uses RCT's SOAP services must encrypt the data before sending it. The format is CMS / PCKS#7 which is abstracted away by eHealth crypto lib. For the REST service we were asked to use the format JWE using the json serialization.

"JOSE is short for Javascript Object Signing and Encryption, which is the IETF Working Group that developed the JSON Web Signature (JWS), JSON Web Encryption (JWE) and JSON Web Key (JWK) specifications. JWS and JWE use JSON and base64url encoding to secure messages in a (relatively) simple, compact and web safe format while JWK defines a JSON representation of cryptographic keys. The actual algorithms for JWS, JWE and JWK are defined in JSON Web Algorithms (JWA)."

Source: https://bitbucket.org/b_c/jose4j/wiki/Home

JOSE: Javascript Object Signing and Encryption.

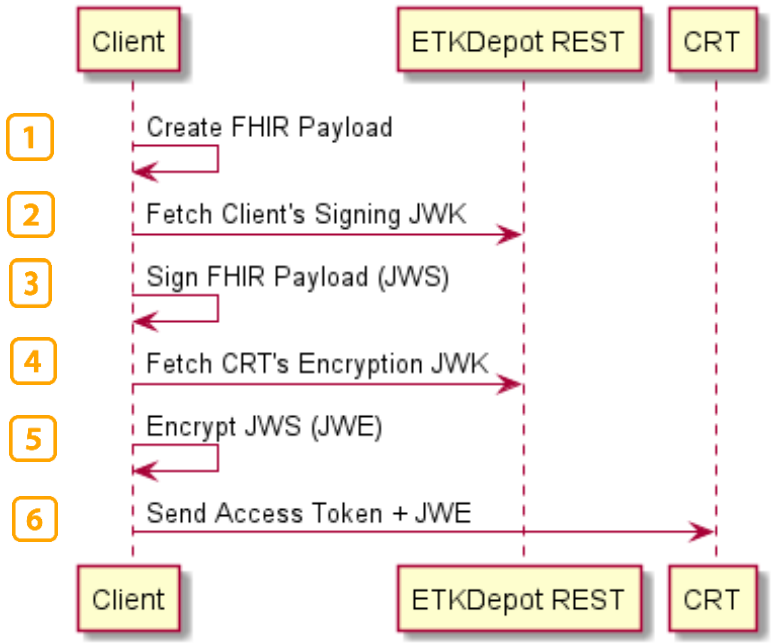
JWS: JSON Web Signature, RFC-7515

JWE: JSON Web Encryption, RFC-7516

JWK: JSON Web Key, RFC-7517

JWA: JSON Web Algorithms, RFC-7518

5.2.2.1 Encryption

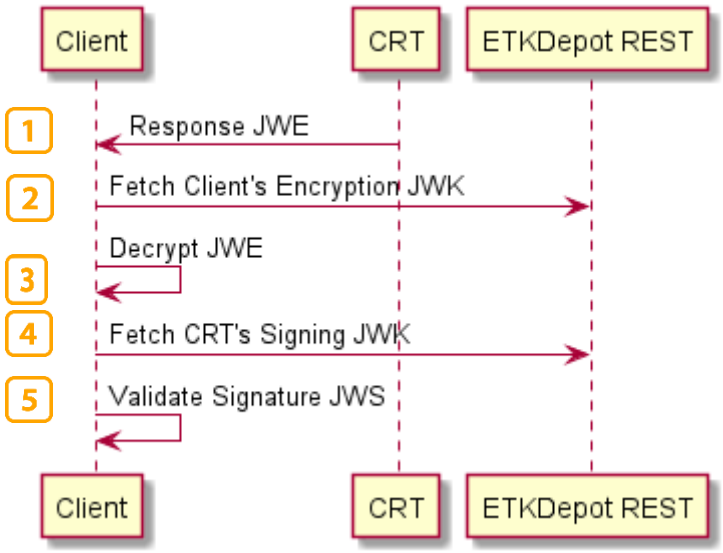


1) Create FHIR Payload
2) Fetch Client's Signing JWK. Client will fetch its own JWK to find the correct `kid`.
3) Sign FHIR Payload (JWS). Using its private signing key, the client will create a JWS.
4) Fetch CRT's Encryption JWK. Fetch CRT public encryption key.
5) Encrypt JWS (JWE).The client will encrypt its JWS into a JWE using CRT's public encryption key.
6) Send Access Token + JWE. The client will identify itself using the I.AM Connect access token to send its encrypted payload.

Algorithm used:

- Signing: `RS256`
- Key Encryption: `RSA-OAEP`
- Content Encryption: `A128GCM`

5.2.2.2 Decryption



1) Response JWE. CRT sent some encrypted payload.
2) Fetch Client's Encryption JWK. Validate that the JWE's `kid` correctly identify one of the client's encryption key.
3) Decrypt JWE. Decrypt the payload using the appropriate client private encryption key.
4) Fetch CRT's Signing JWK. Validate that the JWS's `kid` correctly identify CRT public signing key.
5) Validate Signature JWS. Validate CRT signature using its public signing key.

Algorithm used:

- Signing: `RS256`
- Key Encryption: `RSA-OAEP`
- Content Encryption: `A128GCM`

5.3 The Central Traceability Register Rest Service

The REST interface is described with a JSON/ Swagger API.

Notifications of implants and notifications of removals of implants are hereafter referred to as '*surgicalNotifications*'. A '*surgicalNotifications*' is a compound set of information about an implant intervention or about a the removal of a device from a previous implant intervention, information about the patient, the medical device(s), the prescriber, the pharmacy and deliverer (optional), the installer and the organization where the intervention took place. This set of healthcare-related information is assembled together into a single logical package and provides a single coherent statement of meaning. The 'language' or 'format' used describing this set is the HL7 FHIR® (Fast Healthcare Interoperability Resources) specification which is designed to enable the exchange of healthcare-related information in more standardised and easier way.

All of the methods consume and produce the JavaScript Object Notation (JSON), a lightweight, text-based and language-independent data interchange format. The MIME type to be used is **application/json** with default encoding UTF-8.

5.3.1 POST /rct/v1/surgicalNotifications

This method can carry out two functionalities :

- create a surgical notification about new implants into the registry.
- create a surgical notification about an implant removal into the registry.

Depending on the FHIR payload either one of the two will be executed.

The removal must be based on a previously entered notification of an implantation of a device, which is uniquely identified by its *technicalId.* This means only notified implants in the registry can be subject of a removal.

5.3.1.1 Request

Element	Description
protected **unprotected** **alg** **header** **kid** **tag** **encrypted_key**	See https://tools.ietf.org/html/rfc7515 See https://tools.ietf.org/html/rfc7516
ciphertext	Contains the *surgicalNotifications* information about the implantation intervention and the medical device(s) implanted or about the removal intervention of medical device(s) already implanted.. This information must be composed in the HL7 FHIR® standards and MUST be encrypted since it contains privacy statements about a patient.

Example:

POST /rct/v1/surgicalNotifications HTTP/1.1

Accept: application/json
Authorization: Bearer eyhbGciOiJSUz5cCI...zP6EbTYSMc8lQ2ERryw

Content-Type: application/json
Content-Length: 8889
Host: api-int.ehealth.fgov.be
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.5.11 (Java/1.8.0_101)


```
{
  "protected": "eyJlbmMiOiJBMTI4R0NNIn0",
  "unprotected": {
       "alg": "RSA-OAEP-256"
  },
  "header": {
    "kid": "E582223718666271910972020160643499960 50"
  },
  "encrypted_key": "r-mPwsRh0coOcydXHhrxw...3p4Y14D7yyQ",
  “ciphertext”: “…”,
  "tag": "PEX5b9s7yoCEvKNkCWKB4Q"
}
```


5.3.1.2 Response

Element	Description
notification	The parameter of the response header will contain the URI of the newly created *surgicalNotifications* e.g. **/surgicalNotifications/201910303220165**

Http response status code	Error Code	Error description
201	CREATED	The request has been fulfilled, resulting in the creation of a new resource *surgicalNotifications*.

Example:

*HTTP/1.1 **201** Created*
Max-Forwards: 20
Content-Length: 0
Connection: keep-alive
X-CorrelationID: Id-7169565ebffe45bf328aaf07 0
Date: Wed, 26 Feb 2020 12:50:04 GMT
notification: 202002268227705
Set-Cookie: 7c539e2154b98fe31ab78f0c6ad2accabe768800729203f2e87015db; path=/webrct/; HttpOnly
Set-Cookie: BIGipServer~VAS~router.int.paas.vasdc.be=!S0spvjToKIscLcc...5nP+9yjigfU=; path=/webrct/; Httponly
Content-Type: text/plain
Strict-Transport-Security: max-age=15768000

5.3.2 PUT /rct/v1/surgicalNotifications/{surgicalNotificationId}

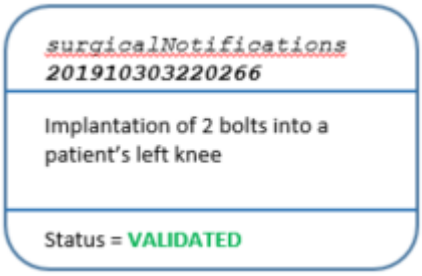
This method creates a new updated version of a surgical notification.

This method is to be seen as performing two steps :

1. The old *surgicalNotifications* referenced by the *surgicalNotificationId* is logically deleted in the registry and thus leaving a historical trace.
2. A new *surgicalNotifications* is created with a new *surgicalNotificationId* returned in output.

Schematically this gives us :

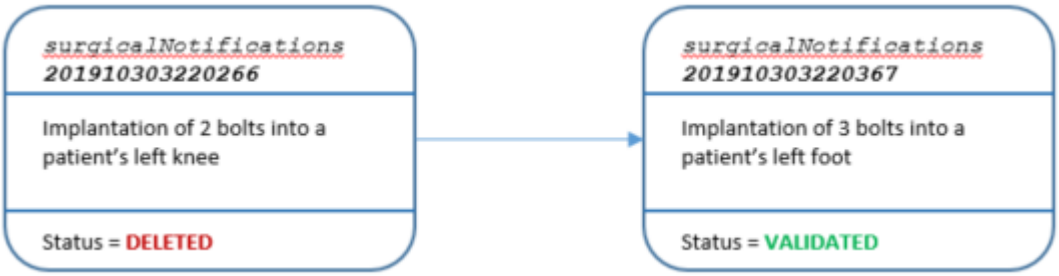
If an old erroneous notification in the registry as given below . . .



. . . is corrected by the method

PUT /surgicalNotifications/201910303220266

. . . this will result in the following situation in the registry . . .



5.3.2.1 Request

Element	Description
protected **unprotected** **alg** **header** **kid** **tag** **encrypted_key**	See https://tools.ietf.org/html/rfc7515 See https://tools.ietf.org/html/rfc7516
surgicalNotificationId	The unique public identifier of an existing *surgicalNotifications* we want to update.
ciphertext	Contains the updated *surgicalNotifications* information. This information MUST be composed in the HL7 FHIR® standards and MUST be encrypted since it contains privacy statements about a patient.

Example:

PUT /rct/v1/surgicalNotifications/202002268227705 HTTP/1.1

Accept: application/json
Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR . . . cCIgOilQ2ERHCnXOO_u5zGryw
Content-Type: application/json

Content-Length: 8870
Host: api-int.ehealth.fgov.be
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.5.11 (Java/1.8.0_101)


```
{
  "protected": "eyJlbmMiOiJBMTI4R0NNIn0",
  "unprotected": {
    "alg": "RSA-OAEP-256"
  }
  ,"header": {
    "kid": "E582223718666271910972020160643499960 50"
  },
  "encrypted_key": "mc0d0Amun_DL8BAwcidIyL...Dp_ri-POre9hyXc0h2A",
  “ciphertext”: “…”,
  "tag": "pLHmNoBrrEXpb9qDZGmUDA"
}
```


5.3.2.2 Response

Element	Description
notification	The parameter of the response header will contain the URI of the newly created *surgicalNotifications* e.g. **/surgicalNotifications/201910303220367**

Http response status code	Error Code	Error description
201	CREATED	The request has been fulfilled, resulting in the creation of a new resource *surgicalNotifications*.

Example:

*HTTP/1.1 **201** Created*
Max-Forwards: 20
Content-Length: 0
Connection: keep-alive
X-CorrelationID: Id-7169565ebffe45bf328sdfsd0
Date: Wed, 26 Feb 2020 12:50:34 GMT
notification: 202002268227814
Set-Cookie: 7c539e2154b98fe31ab78f0c6asddsfsdfaé rzerzd2ccabe87015db; path=/webrct/; HttpOnly
Set-Cookie: BIGipServer~VAS~router.int.paas.vasdc.be=!S0spvjToKIscLcc...5nP+9yjigfU=; path=/webrct/; Httponly
Content-Type: text/plain
Strict-Transport-Security: max-age=15768000

5.3.3 GET /rct/v1/surgicalNotifications

This methods allows consultation of surgical notifications in different flavours depending on the combinations of parameters.

5.3.3.1 Request

Element	Description
id	A mandatory identifier to perform a search by. Different formats of identifiers are possible depending on the type of identifier specified in the parameter **type** defined just below: ○ If type = **notification** then id must be a *surgicalNotifications* identifier

	e.g. id = 201910303220367 ○ If type = **patient** then id must be the Belgian INSS number (International Number Social Security) of the patient e.g. id = 68031904851 ○ If type = **riziv** then id must be the notification code (assigned by the RIZIV) of the medical device that was implanted or explanted e.g. id = 000001694629 *(ABC CERVICAL PLATE, 4 HOLE, 20 mm)* ○ If type = **uid** then id must be the UDI-DI part of the Unique Device Identification of the medical device that was implanted or explanted e.g. id = (01)07813252193791 *(Neuroform 2.5mmx15mm)*
type	A mandatory type of identifier that can be used to perform the search by. Following types are available: ○ **notification** – identifiers of this type represent the id of *surgicalNotifications* known to the registry. ○ **patient** - identifiers of this type represent the 11-digit SSIN (Belgian Social Security Identification Number) ○ **riziv** - represent the notification code of a medical device attributed by the RIZIV ○ **uid** - represents the UDI-DI part of the Unique Device Identification of medical devices
procedure	The type of surgical procedure to be filtered in the output. This parameter is not mandatory. The following values are possible: ○ **implantation** – will only return *surgicalNotifications* of implants in the output ○ **removal** - will only return *surgicalNotifications* of removals in the output If the parameter is omitted both implantations and removals will be returned in the output.
startDate	Start date from which surgical notifications are to be collected. The format must be YYYY-MM-DD and must be a valid date. This date must be prior or equal to **endDate**.
endDate	End date untill when surgical notifications are to be collected. The format must be YYYY-MM-DD and must be a valid date. This date must be equal or superior to **startDate**.
page	Number of the requested page in a paged resource collection. Page numbers are 1-based.
pageSize	Page size in a paged resource collection. Pagesize number is 1-based.

Examples:

GET …/surgicalNotifications?id=201910303220266&type=notification

➔ will return 0 or 1 *surgicalNotifications*

GET …/surgicalNotifications?id=68031904589&type=patient

➔ will return 0 or more *surgicalNotifications* about the patient 68031904589

GET …/surgicalNotifications?id=000001694629&type=riziv&procedure=removal

➔ will return 0 or more *surgicalNotifications* of removals of the medical device 000001694629

GET …/surgicalNotifications?id=000001694629&type=riziv&procedure=removal&startDate=2019-03-19

➔ will return 0 or more *surgicalNotifications* of removals of the medical device 000001694629 as from march the 13th in 2019.

etc . . .

GET /rct/v1/surgicalNotifications?id=20190303220266
&type=notification&page=1&pageSize=1 HTTP/1.1

Accept: application/json
Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia. . .
UT0swIn0.eyJqdGkiOiI5NzI1ZTc5Zi1CrBUBvork22dnk8lQ2ERHCnXOO_u5zGryw
Host: api-int.ehealth.fgov.be
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.5.11 (Java/1.8.0_101)

5.3.3.2 Response

Http response status code	Error Code	Error description
200	OK	The HTTP request was successful. A *surgicalNotifications* is returned.

Example:

HTTP/1.1 ***200*** *OK*
Max-Forwards: 20
Content-Length: 1185
Connection: keep-alive
X-CorrelationID: Id-8569565ed801125b9c88896b 0
Cache-control: private
Date: Wed, 26 Feb 2020 12:50:13 GMT
Set-Cookie: 7c539e2154b98fe31ab78f0c6ad45fce=9ad2accabe768800729203f2e87015db; path=/webrct/; HttpOnly
Set-Cookie: BIGipServer~VAS~router.int.paas.vasdc.be=!wyMuHD4. . .IG5v9lB9yJc=; path=/webrct/; Httponly
Content-Type: application/json
Strict-Transport-Security: max-age=15768000


```
{
  "protected": "eyJlbmMiOiJBMTI4R0NNIn0",
  "unprotected": {
    "alg":"RSA-OAEP-256"
  },
  "header": {
    "kid":"E54095366290630488470034383456390448943"
  },
  "encrypted_key": "MzxyFl6FFkY...HrJlDg7brWhBWo"
  ,"tag": "2VDVgov7NHvDPySUCt9EkQ"
}
```


5.3.4 DELETE /rct/v1/surgicalNotifications/{surgicalNotificationId}

This method deletes a single surgical notification by its identifier.

5.3.4.1 Request

Element	Description
surgicalNotificationId	The unique public identifier of an existing *surgicalNotifications* we want to delete.

Example:

DELETE /rct/v1/surgicalNotifications/202002268227705 HTTP/1.1

Accept: application/json
Authorization: Bearer eyJhbGciOiJSUzI1Nc...k22dnk8lQ2ERHCnXOO_u5zGryw
Content-Length: 0
Host: api-int.ehealth.fgov.be
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.5.11 (Java/1.8.0_101)
http-outgoing-3 >>

5.3.4.2 Response

Http response status code	Error Code	Error description
204	NO_CONTENT	The request has been fulfilled, resulting in the creation of a new resource *surgicalNotifications*.

Example:

HTTP/1.1 ***204*** *No Content*
Max-Forwards: 20
Connection: keep-alive
X-CorrelationID: Id-8a69565e01ffbaa6a345a5f1 0
Date: Wed, 26 Feb 2020 12:50:27 GMT
Set-Cookie: 7c539e2154b98fe31ab78f0c6ad45fce=9ad2accabe768800729203f2e87015db; path=/webrct/; HttpOnly
Strict-Transport-Security: max-age=15768000
http-outgoing-3

6. Risks and security

6.1 Risks & safety

6.2 Security

6.2.1 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. This will ensure an effective capacity management.

In case of technical issues on the WS, the partner may obtain support from the contact center (see Chap 3)

In case the eHealth platform finds a bug or vulnerability in its software, we advise the partner 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 the eHealth platform delivered, he is obliged to contact and inform us immediately. He is not allowed to publish this bug or vulnerability in any case.

6.2.2 Web service

WS 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.

6.2.3 The use of username, password and token

The username, password and token are strictly personal. Partners and clients are not allowed to transfer them. Every user takes care of his username, password and token and he is forced to confidentiality of it. Moreover, every user is responsible of every use, which includes the use by a third party, until the inactivation.

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 eHealth platform service, 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 WS. Most of the required integration info to integrate is published on the portal of the eHealth platform.

Upon request, the eHealth platform provides you in some cases, 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 the 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 his 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 eHealth platform service for one of his 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 three different possible types of response:

- If there are no technical or business errors, a business response is returned
- If a business error occurred, it is contained in a business response
- If a technical error occurred, a REST fault exception is returned

The response returned in case of errors and failures of the CRT REST service will be either one of two models depending on the status code.

- The *Problem* model:

```
{
  "type" : "string($uri)",
  "title" : "string",
  "status": "integer($int32)",
  "detail" : "string",
  "instance": "string($uri)"
}
```

Where

- **type** : An URI reference that identifies the problem type. When dereferenced,it SHOULD provide human-readable documentation for the problem type (e.g. using HTML).
- **title** : A short, summary of the problem type. Written in english and readable for engineers (usually not suited for non technical stakeholders and not localized)
- **status** : The HTTP status code generated by the origin server for this occurrence of the problem.
- **detail** : A human-readable explanation specific to this occurrence of the problem
- **instance** : A URI reference that identifies the specific occurrence of the problem. It may or may not yield further information if dereferenced.

Example:

```
{
  "type": "about:blank",
  "title": "RCT-0028 : The patient SSIN is not a valid SSIN.",
  "status": "409",
  "detail": "The SSIN field of the individual does not have a valid format",
  "instance": "2ebb2b80-ca6e-4659-9959-06fd4991e72d"
}
```

- The *InvalidParamProblem* model:

```
{
  "type" : "string($uri)",
  "title" : "string",
  "status": "integer($int32)",
  "detail" : "string",
  "instance": "string($uri)",
  "invalidParams" : [
```

```
    {
      "in" : "string",
      "name" : "string",
      "reason" : "string",
      "value" : "no type specified"
    }
  ]
}
```


Where

- **type** : An URI reference that identifies the problem type. When dereferenced,it SHOULD provide human-readable documentation for the problem type (e.g. using HTML).
- **title** : A short, summary of the problem type. Written in english and readable for engineers (usually not suited for non technical stakeholders and not localized)
- **status** : The HTTP status code generated by the origin server for this occurrence of the problem.
- **detail** : A human-readable explanation specific to this occurrence of the problem
- **instance** : A URI reference that identifies the specific occurrence of the problem. It may or may not yield further information if dereferenced.
- **in** : The location of the invalid parameter (body, path, query, header)
- **name** : The name of the invalid parameter
- **reason** : A message explaining the violation
- **value** : The value of the erroneous parameter

Example:


```
{
  "type": "about:blank",
  "title": "Bad request",
  "status": "400",
  "detail": "The input message is incorrect",
  "instance": "2ebb2b80-ca6e-4659-9959-06fd4991e72d",
  "invalidParams": [
    {
      "in": "query",
      "name": "id",
      "reason": "The parameter `id´ must be a correct RIZIV number",
      "value": "0"
    }
  ]
}
```


8.1 POST /surgicalNotifications

Http standard error code	means	Error description
409	BUSINESS_FAILURE	Business validation of the request message failed
others	UNEXPECTED_ERROR	All HTTP error status codes not intercepted above

The response returned will be the *Problem* model.

8.2 POST /surgicalNotifications/{surgicalNotificationId}

Http standard error code	means	Error description
404	NOT_FOUND	The requested resource could not be found
409	BUSINESS_FAILURE	Business validation of the request message failed
others	UNEXPECTED_ERROR	All HTTP error status codes not intercepted above

The response returned will be the *Problem* model.

8.3 PUT /surgicalNotifications/{surgicalNotificationId}

Http standard error code	means	Error description
400	BAD_REQUEST	The server cannot or will not process the request due to an apparent client error (e.g., malformed request syntax, size too large, ..)
404	NOT_FOUND	The requested resource could not be found
409	BUSINESS_FAILURE	Business validation of the request message failed
others	UNEXPECTED_ERROR	All HTTP error status codes not intercepted above

8.4 GET /surgicalNotifications

Http standard error code	means	Error description
400	BAD_REQUEST	The server cannot or will not process the request due to an apparent client error (e.g., malformed request syntax, size too large, ..)
404	NOT_FOUND	The requested resource could not be found
others	UNEXPECTED_ERROR	All HTTP error status codes not intercepted above

The response returned will be the *Problem* model or in case of the BAD_REQUEST (400) the *InvalidParamProblem* model.

8.5 DELETE /surgicalNotifications/{surgicalNotificationId}

Http error code	Error Code	Error description
400	BAD_REQUEST	The server cannot or will not process the request due to an apparent client error (e.g., malformed request syntax, size too large, ..)
404	NOT_FOUND	The requested resource could not be found
others	UNEXPECTED_ERROR	All HTTP error status codes not intercepted above

The response returned will be the *Problem* model.

8.6 Business Errors

In case of business error, you can contact Smals Support .

Below you can find the list of all possible business errors from a code error 500 (server side).

Code	Error description
RCT-00002	The replaced notification number is not a valid notification number.
RCT-00106	The patient SSIN is not a valid SSIN.
RCT-00107	The patient SSIN was not found.
RCT-00108	The NIHII number of the hospital is not a valid NIHII number.
RCT-00109	The NIHII number of the hospital was not found.
RCT-00110	The intervention must be of the type "explantation".
RCT-00111	The intervention date cannot be in the future.
RCT-00112	The intervention date must be equal to or after the prescription and delivery date.
RCT-00113	No valid SSIN or NIHII number is specified for the specialist.
RCT-00114	The specialist was not found.
RCT-00115	The specialist does not have at least one of the required qualities.
RCT-00116	The intervention does not contain at least one intervention device.
RCT-00117	No valid notification code was specified for the implant.
RCT-00118	The implant was not found.
RCT-00119	No delivery transaction was found for the implant.
RCT-00120	The delivery date cannot be in the future.
RCT-00121	The delivery date must be equal to or after the prescription date and equal to or before the intervention date.
~~RCT-00122~~	~~No valid SSIN or NIHII number is specified for the deliverer.~~
~~RCT-00123~~	~~The deliverer was not found.~~
~~RCT-00124~~	~~The deliverer does not have at least one of the required qualities.~~
RCT-00125	The prescription date cannot be in the future.
RCT-00126	The prescription date must be equal to or before the delivery and intervention date.

RCT-00127	No valid SSIN or NIHII number is specified for the prescriber.
RCT-00128	The prescriber was not found.
RCT-00129	The prescriber does not have at least one of the required qualities.
RCT-00130	The notification number of the implantation is not a valid notification number.
RCT-00131	Not enough implants with the specified notification code could be found in the specified implantation notification.
RCT-00132	The notification number is not a valid notification number.
RCT-00133	The patient SSIN is not a valid SSIN.
RCT-00134	The implant notification code is not a valid notification code.
RCT-00135	The intervention type is not one of the allowed values: "IMPLANTATION" or "EXPLANTATION".
RCT-00136	The intervention start date cannot be in the future.
RCT-00137	The intervention start date cannot be after the intervention start date.
RCT-00138	The intervention end date cannot be in the future.
RCT-00139	The intervention end date cannot be before the intervention end date.
RCT-00140	One or more explantation notifications exist for one or more implants in this implantation notification. All of these need to be deleted before the implantation notification can be deleted.
RCT-00141	No therapeutic link exists between the specialist and the patient.
RCT-00142	No notification found or no therapeutic link with the patient exists for the existing notification.
RCT-00143	The NIHII or CBE number of the pharmacy must be provided.
RCT-00144	The number of the pharmacy is not a valid NIHII or CBE number.
RCT-00145	No pharmacy could be found for the given NIHII or CBE number.
RCT-00146	A notification with the same data as the entered notification already exists in the registry. The existing notification has the notification number: [insert notification number here].
RCT-00147	The requested notification has been replaced, please use the following notification number: XXXXXXXXXXXXXXX.
RCT-00148	No notification found for the requested notification number.
RCT-00149	The requested notification does not have the correct type.
RCT-00150	The author of the intervention transaction does not correspond with the sender of the message.
RCT-00151	You are not authorized to delete or modify this notification.
RCT-00152	The SSIN or NIHII number of the specialist must be provided.
~~RCT-00153~~	~~The SSIN or NIHII number of the deliverer must be provided.~~
RCT-00154	The SSIN or NIHII number of the prescriber must be provided.
RCT-00155	Too many results match the given search criteria. Please use more specific search criteria in order to reduce the number of results.
RCT-00156	Not all of the necessary data are available in the authentic sources for this implant.
RCT-00157	The SSIN XXXXXXXXXXX has been replaced by the following SSIN: XXXXXXXXXXX. Please use the new SSIN.
RCT-00158	The SSIN has been replaced by another SSIN but the mutation has not yet been processed by CRT. Please try again later.
RCT-00158	The SSIN has been replaced by another SSIN but the mutation has not yet been processed by CRT. Please try again later.

RCT-00201	The current state of the notified implant is "deleted".
RCT-00301	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00302	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00303	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00303	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00304	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00304	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00304	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00305	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00306	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00307	One or more external services that are necessary for processing the request are currently unavailable.
RCT-00401	One or more external services that are necessary for processing the request returned a technical error.
RCT-00402	One or more external services that are necessary for processing the request returned a technical error.
RCT-00403	One or more external services that are necessary for processing the request returned a technical error.
RCT-00404	One or more external services that are necessary for processing the request returned a technical error.
RCT-00405	One or more external services that are necessary for processing the request returned a technical error.
RCT-00406	One or more external services that are necessary for processing the request returned a technical error.
RCT-00406	One or more external services that are necessary for processing the request returned a technical error.
RCT-00406	One or more external services that are necessary for processing the request returned a technical error.
RCT-00407	One or more external services that are necessary for processing the request returned a technical error.

There error codes that have been crossed out mean that this error is not applicable anymore due to a business change. In this case deliverer (pharmacy/pharmacist) became an optional input.

8.7 Structural message Errors

RCT-00501	FHIR message structure : Wrong number of resources / elements This error can occur whenever : 1. A specific resource can only be declared once in the entries of the bundle. - 0 or more than 1 serviceRequest in a bundle resource - 0 or more than 1 procedure in a bundle resource 2. The patient resource - 0 or more than 1 identifier with a SSIN code in a patient resource 3. The Procedure resource - 0 or more than 1 performer in a procedure resource 4. The Practitioner resource - More than 1 identifier with a SSIN code in a practitioner resource - More than 1 identifier with a NIHDI code in a practitioner resource 5. The Organization resource - 0 or more than 1 identifier with a NIHDI code in an organization resource 6. The Device resource - 0 or more than 1 identifier with a NIHDI code in a device resource - More than 1 udiCarrier in a device resource 7. The SupplyDelivery resource - More than 1 receiver in a supplyDelivery resource 8. The ServiceRequest resource - 0 or more than 1 bodySite in the serviceRequest resource (bodySite can be null but not empty) 9. The coding element - 0 or more than 1 coding in a codeableConcept element - The code in the serviceRequest resource

RCT-00502	FHIR message structure : Required resource / element is missing To see if a resource is missing, the application looks for an entry with the required resourceType attribute 1. The Bundle resource - Missing bundle resource in the encrypted fhir payload - Missing entry list in the bundle resource - Missing timestamp in a bundle resource 2. The Practitioner resource - Missing identifier with either NIHDI or SSIN as code for a practitioner resource 3. The SupplyDeliver resource - No supplyDeliveries in the entries of the bundle resource - Missing suppliedItem in a supplyDelivery resource - Missing itemReference element in a suppliedItem element - Missing reference in a supplier element - the supplier element is optional in a supplyDelivery resource 4. The ServiceRequest resource - Missing authoredOn element in a serviceRequest resource - Missing code element in a serviceRequest resource - Missing requester element in a serviceRequest resource 5. The Procedure resource - Missing performedDateTime element in a procedure resource - Missing actor element in a performer element 6. The Patient resource - Missing identifier element in a patient resource - Missing gender element in a patient resource 7. The Device resource - Missing identifier element in a device resource 8. The Organization resource - Missing identifier element in an organization resource - Missing reference element in a receiver element 9. The identifier element required a value and a code - Missing value element in a identifier element - The practitioner has a list of identifiers - The patient has a list of identifiers - The organization has a list of identifiers - The device has a list of identifiers 10. The reference element - Missing reference in a reference element - the actor in a performer - the onBehalfOf in a performer - the requester in a serviceRequest resource - the subject in a serviceRequest resource - the itemReference in a suppliedItem element 11. The coding element - Missing code in a coding element - code in a serviceRequest resource

RCT-00503	FHIR message structure : Wrong code 1. Wrong surgical notification code - The code mentionned in the serviceRquest is unknown (for an implantation/explantation)
RCT-00504	FHIR message structure : Wrong number of resources identified by a single reference 1. Wrong number of resources identified by the same reference - 0 or more than 1 device resources are identified by a single identifier - 0 or more than 1 organization resources are identified by a single identifier - 0 or more than 1 practitioner resources are identified by a single identifier
RCT-00505	FHIR message structure : The payload is not well formatted This error relates to : 1. The json body - The body is incorrect json format - The encrypted bundle is incorrect json format 2. The fhir model is not respected - A single element is defined instead of a list - An element is defined at the wrong level - An unknown element is defined
RCT-00506	Wrong parameters used during the rest call This resource level error can occur whenever : 1. The id and type parameters are required - Missing id and/or type parameters in the request 2. The type parameter - The type is unknown 3. The dates parameters - Incorrect format for the startDate - Incorrect format for the endDate 4. The procedure parameter - The value of the procedure parameter is unknown