UCI Share

Explanatory Notes

The figures below show the flowchart of the use case implementation *Share*, from four different perspectives:

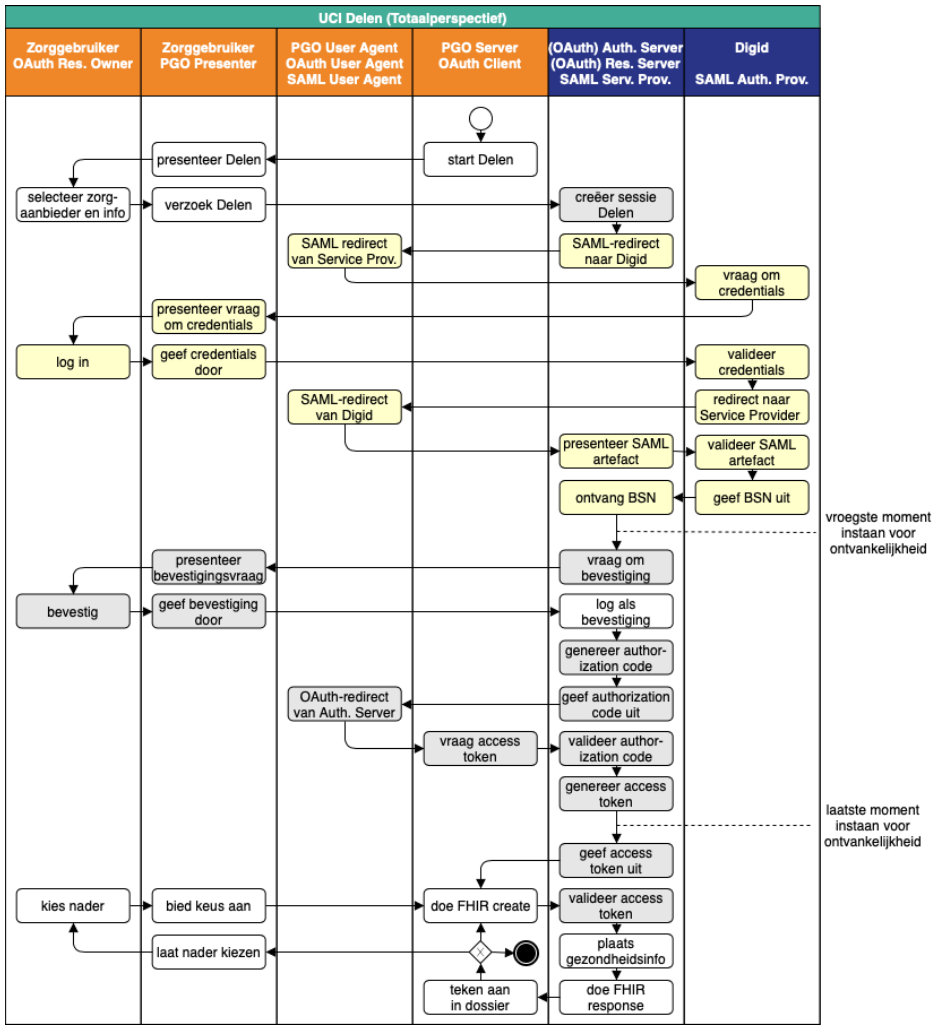
- the overall perspective, with both the happy flow and the exceptions;
- the perspective of the *PHE Server* (= *OAuth Client*), who comes under the *Individual's Service Provider*. In so far as the last-named are a participant in the MedMij Framework, they can thus read this figure as being his mandatory participation in the use case implementation *Share;*
- the perspective of the (*OAuth) Authorisation Server/*(*OAuth) Resource Server/ SAML Service Provider*, who comes under the *Care Provider's Service Provider*. In so far as the last-named are a participant in the MedMij Framework, they can thus read this figure as being his mandatory participation in the use case implementation *Share;*
- the perspective of the *Care User* (= *OAuth Resource Owner*).

The flowcharts only show the situation in which all actions are successful, up to and including the ultimate sharing of the health information (the so-called 'happy flow' situation). In line with the MedMij corporate identity, the three orange paths belong to the Individual's Domain and the two blue ones to the Care Provider's Domain. Many actions in the flowcharts are shown in colour. Together, the actions coloured in light grey form the authorisation flow in accordance with OAuth 2, whereas the actions coloured in light yellow together form the authentication flow in accordance with DigiD/SAML. In other words, these colours only refer to the standards used and say nothing about which component has to execute the step. Authentication is thus embedded in authorisation. In the flowcharts for the specific perspectives, it is only those actions in the path that belong to that perspective that are named. The actions in the other paths are compressed and shown without names.

Responsibilities regarding the data that is handled in this use case implementation are, together with those of UCI Compile , included on a separate page.

Overall perspective

Happy flow



The MedMij Framework recommends that the accessibility condition be made effective from the earliest stated moment. In release 1.1.1, the MedMij Framework permits this condition to become effective later on but not later than the final moment stated in the figure.

Explanatory Notes

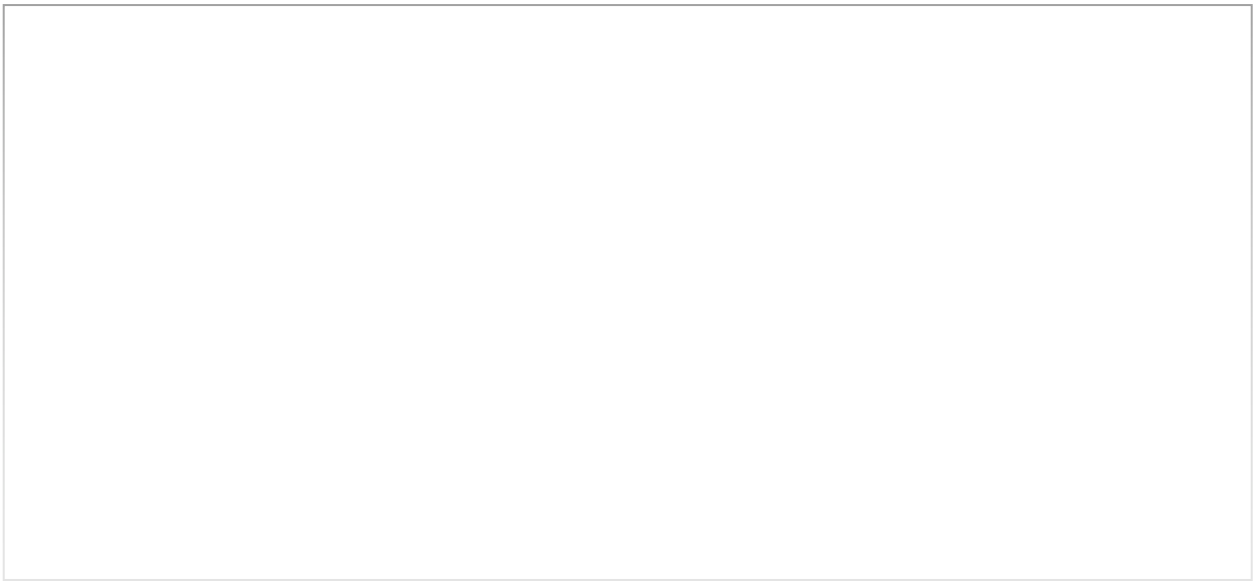
In each completion of the flow described in the diagram, there is in all cases only a single one of each of the roles named above.

The flow has the following steps:

1. The *PHE Server* starts the flow by presenting in the *PHE Presenter* of the *Care User* the possibility of a particular *Data Service* with a certain *Care Provider*. This always relates to precisely one *Data Service* (a single scope, in OAuth terms). From the *Care Providers List*, the *PHE Server* knows which *Data Services* are provided by a *Care Provider*. If desired, the *Data Service Names* from the *Data Service Names List* are used.

2. The *Care User* makes explicit their selection and gets the *OAuth User Agent* to send a subrequest to the *Authorisation Server* . The address of the authorisation endpoint is taken from the *CPL*. The redirect URI indicates where the *Authorisation Server* must hereafter redirect the *OAuth User Agent* to (with the authorisation code).
3. The *Authorisation Server* now begins the OAuth flow (in his role as *OAuth Authorisation Server*) by creating a session.
4. Next, the *Authorisation Server* (now in the role of *SAML Service Provider*) starts the SAML flow by redirecting the *OAuth User Agent* to *DigiD*, also providing him with a redirect URI, that indicates to where *DigiD* must later return the *OAuth User Agent*, after the logging in of the *Care User*.
5. DigiD asks the *Care User* - via his *PHE Presenter* - for his login details.
6. When these are correct then *DigiD* redirects the *OAuth User Agent* back to the *Authorisation Server*, whilst giving him a retrieval certificate: the SAML artefact.
7. With this retrieval certificate, the *Authorisation Server* retrieves the BSN (citizen service number) directly from *DigiD*.
8. The earliest moment then comes at which the *Authorisation Server* guarantees that the *Care Provider* - for the *Data Service* in question - is receptive to the health information of this *Individual*; otherwise the happy flow terminates. One important factor here is that the *Individual* must be at least 16 years old for this. See a separate page for detailed explanatory notes.
9. If this is successful then the *Authorisation Server* presents - via the *PHE Presenter* - to the *Care User* the question of whether the last-named confirms that it will allow the requested personal health information to be provided by the *PHE Server* (as *OAuth Client*). Under the flowchart, it is specified which information, and from where, the *OAuth Authorisation Server* processes in the confirmation request to be submitted to the *Care User*.
10. Upon agreement, the *Authorisation Server* logs this as confirmation, generates an authorisation code and sends this as a retrieval certificate by means of a browser redirect - along with the redirect URI received in step 1 - to the *PHE Server*. The *Authorisation Server* sends with it the local state information that it received in the initial step of the *PHE Server*. The last-named recognises in it the request that it must associate the authorisation code with.
11. The *PHE Server* not only interprets this authorisation code as a retrieval certificate but also deduces from it that the confirmation has been given and logs the receipt of the retrieval certificate.
12. The *PHE Server* applies again to the *Authorisation Server* with this retrieval certificate but now without the intermediation of the *OAuth User Agent* for an access token.
13. The *Authorisation Server* then generates an access token. The final moment then comes at which the *Authorisation Server* must guarantee that - for the *Data Service* in question - the *Care Provider* is receptive for the health information of the relevant *Individual*. If this is the case then the *Authorisation Server* sends the access token to the *PHE Server*. If it isn't then the *Authorisation Server* terminates the happy flow and does not send an access token to the *PHE Server*.

14. The *PHE Server* is now ready to present the health information to the *Resource Server* . It obtains the address of the resource endpoint from the *CPL*. It places the access token in the message and ensures that no BSN is included in the message.
15. The *Resource Server* checks whether the received token grants entitlement to present the information, places this with underlying or other destinations and sends a reply in a FHIR response to the *PHE Server*.
16. He creates a record about this with the health information provided in the personal file. If the *Data Service* that the *Care User* has authorised consists of multiple *Transactions* then the *PHE Server* may subsequently place anew with the *Resource Server* for the *Transactions* still remaining, possibly after new user interaction. This can be done as long as the access token is valid.



During the implementation of the test for the *Care Provider*'s receptiveness for the health information to be shared, it makes sense to take privacy requirements into consideration. If the *Care Provider's Service Provider* is to build new data compilations for the receptiveness test then a processing always takes place under the responsibility of a single *Care Provider*. The combining of processing or inadequate segregation must be avoided. This may only be deviated from under the explicit instruction of the *Care Provider*(s) and requires a careful weighing-up beforehand, due to the associated privacy risks.

Exceptions

Explanatory Notes

The table below describes the situations where there are exceptions. They may be viewed as the implementation counterparts of the exceptions of the use case Share. All exceptions are discovered by the *Authorisation Server* or the *Resource Server*. In this version of the MedMij

Framework, it has been determined that they always lead to the quickest possible termination of the flow by all roles involved. However, the other roles must first be involved about this. In order to prevent the *PHE Server* from obtaining information about the treatment relationships before consent has (already or otherwise) been given for this, the distinction between the exceptions 2, 3 and 4 must not be made by the *PHE Server*.

This table only contains the exceptional situations for which the MedMij Framework lays down its own requirements for the implementation. The specification of OAuth 2.0 also contains more generic exceptional situations, such as the situation in which the redirect URI turns out to be invalid. These exceptional situations must be implemented too.

Number	Implements the exception	Exception	Action
UCI Share 1	UC Share 1	*Authorisation Server* finds the received request to be invalid.	*Authorisation Server* informs *PHE Server* about this exception. *PHE Server*informs *Care User* about it.
UCI Share 2	UC Share 2	*Authorisation Server* cannot establish the identity of the *Care User*.	*Authorisation Server* informs *PHE Server* about this exception. *PHE Se* then informs *Care User* about this.
UCI Share 3	UC Share 3	*Authorisation Server* establishes at any time that *Care Provider* is not receptive to this *Data Service* in respect to health information of *Individual*. See separate page for detailed explanatory notes.	
UCI Share 4	UC Share 4	The confirmation request is denied.	
UCI Share 5	UC Share 5	*Authorisation Server* cannot establish the authorisation.	*Authorisation Server* informs *PHE Server* about this exception. *PHE Se* then informs *Care User* about this.
UCI Share 6	UC Share 6	The validation of the authorisation code by *Authorisation Server* fails.	*Authorisation Server* informs *PHE S* about this exception. *PHE Server* th informs *Care User* about this.

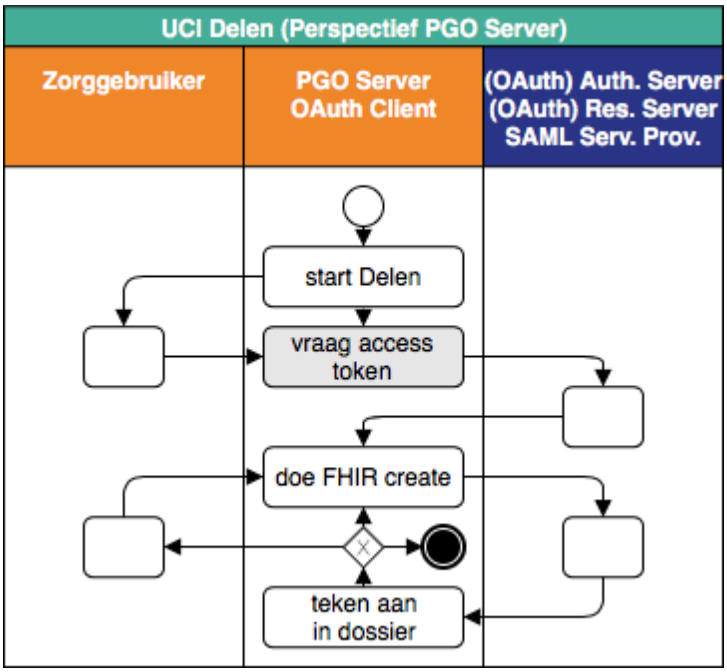
UCI Share 7	UC Share 6	The validation of the access token by *Resource Server* fails.	*Resource Server* informs *PHE Server* about this exception. *PHE Server* th informs *Care User* about this.
UCI Share 8	UC Share 5	*Resource Server* cannot place the requested information with underlying systems in time or at all.	*Resource Server* informs *PHE Server* about this exception. *PHE Server* th informs *Care User* about this.

Specific perspectives

Perspective of PHE Server (happy flow)

Explanatory Notes

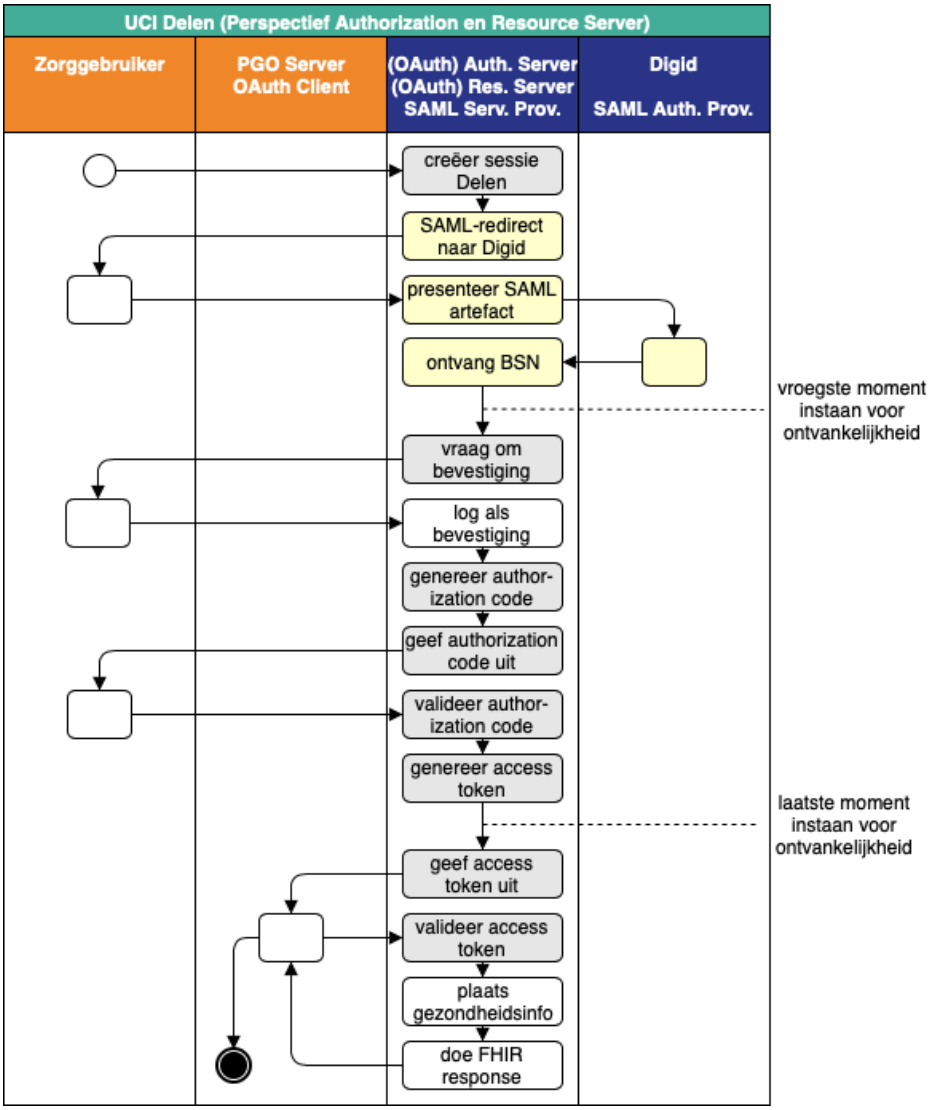
Please find below the same flowchart but this time shown from the perspective of the *PHE Server*. In other words, all in-between steps that are not visible to the *PHE Server* are shorted. *Care User* is "hidden behind the browser" and *DigiD* "behind the *Authorisation Server*".



Perspective of Authorisation Server/Resource Server (happy flow)

Explanatory Notes

Please find below the same flowchart but this time shown from the perspective of the *Authorisation/Resource Server*. In other words, all in-between steps that are not visible to the *PHE Server* are shorted. *Care User* is "hidden behind the browser".

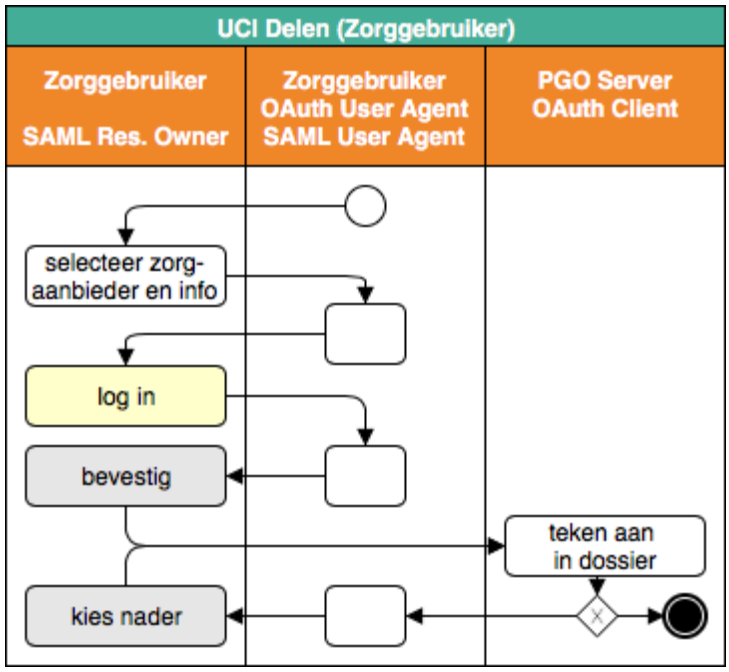


Perspective of Care User (happy flow)

Explanatory Notes

Please find below the same flowchart but this time shown from the perspective of the *Care User*. In other words, all in-between steps that are not visible to the *Care User* are shorted. Almost everything is "hidden behind the browser". We have only kept the final step of *PHE Server* visible because the marking of the shared health information has meaning (i.e.

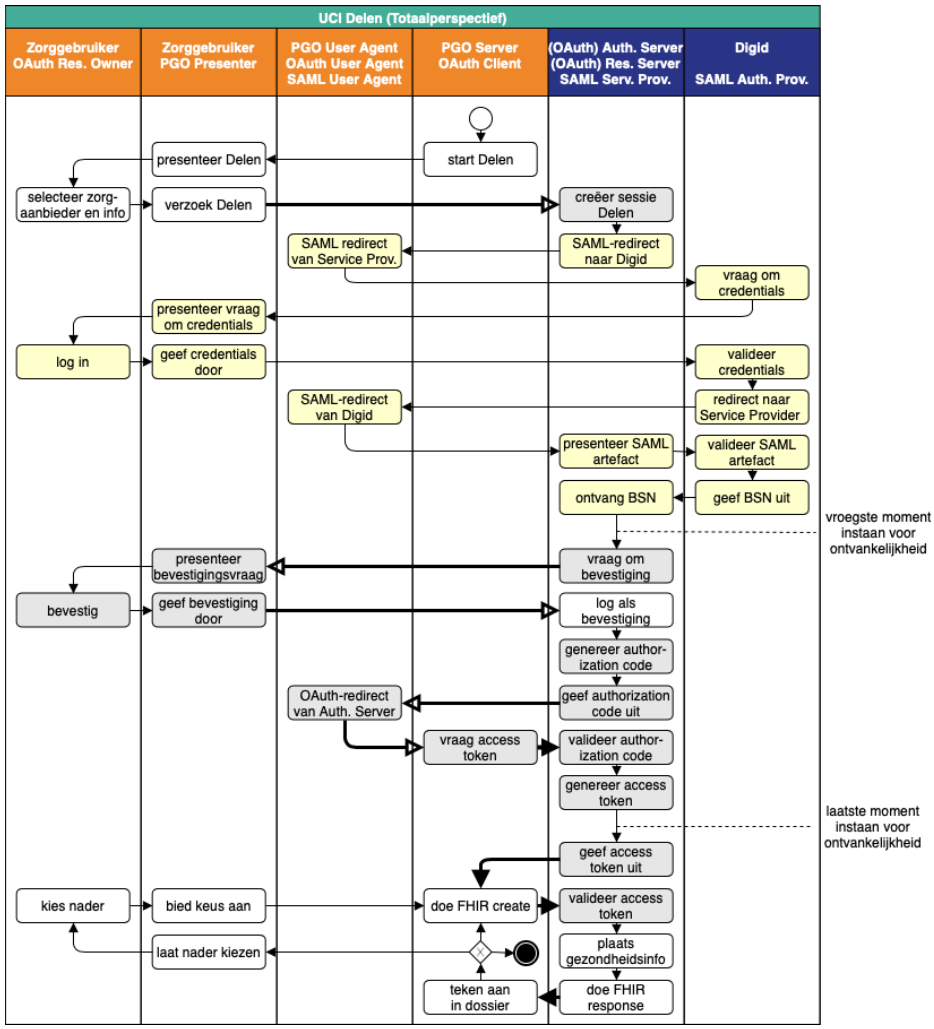
significance) for the *Care User*. The *PHE Server* will probably inform the *Care User* that the sharing was successful but is not obliged to do so.



Frontchannel and backchannel

Explanatory Notes

In the following flowchart of UCI Share, the thick arrows depict the *MedMij data transfer*, with the five cases of frontchannel transfer (open arrowhead) and four cases of backchannel transfer (closed arrowhead) being indicated within it.


UCI Delen (Totaalperspectief)
Zorggebruiker OAuth Res. Owner
Zorggebruiker PGO Presenter
PGO User Agent OAuth User Agent SAML User Agent
PGO Server OAuth Client
(OAuth) Auth. Server (OAuth) Res. Server SAML Serv. Prov.
Digid SAML Auth. Prov.
start Delen
presenteer Delen
selecteer zorg-aanbieder en info
verzoek Delen
creëer sessie Delen
SAML redirect van Service Prov.
SAML-redirect naar Digid
vraag om credentials
presenteer vraag om credentials
log in
geef credentials door
valideer credentials
SAML-redirect van Digid
redirect naar Service Provider
presenteer SAML artefact
valideer SAML artefact
ontvang BSN
geef BSN uit
vroegste moment instaan voor ontvankelijkheid
presenteer bevestigingsvraag
vraag om bevestiging
bevestig
geef bevestiging door
log als bevestiging
genereer author-ization code
OAuth-redirect van Auth. Server
geef authorization code uit
vraag access token
valideer author-ization code
genereer access token
laatste moment instaan voor ontvankelijkheid
geef access token uit
kies nader
bied keus aan
doe FHIR create
valideer access token
laat nader kiezen
plaats gezondheidsinfo
teken aan in dossier
doe FHIR response