Data and performance in UCI Compile and UCI Share

Explanatory Notes

The UCI Compile and UCI Share have a similar set-up. This means that the sets of data that are dealt with in both use case implementations are largely identical. Anticipating on a similar evolution in future releases of the MedMij Framework, the responsibilities for the data and performance in both use case implementations have been placed together on this page.

Consent Declaration and Confirmation Declaration

1a. The question that the *Care User* must be asked in the step "authorise" in UCI Compile is specified in the page Consent Declaration. The following applies in this regard:

- the user-friendly depiction of the identity of the *Care Provider* (Name of Care Provider) is determined by the relevant *Care Provider's Service Provider*, in its service provision relationship with the relevant *Care Provider*;
- the user-friendly depiction of the *Data Service* (Name of Data Service) is obtained from the scope that the *Authorisation Server* received in the very first step of the flow, which corresponds to the *Depiction Name* that is included with the relevant *Data Service* in the *Data Service Names List*;
- the user-friendly depiction of the identity of the *Publisher* (Name SupplierPGO) is obtained from the *OAuth Client List*, based on the redirect_uri (from OAuth) obtained in step 1.

1b. The question that the *Care User* must be asked in the step "confirm" in UCI Share is specified in the page Confirmation Declaration. The following applies in this regard:

- the user-friendly depiction of the identity of the *Care Provider* (Name of Care Provider) is determined by the relevant *Care Provider's Service Provider*, in its service provision relationship with the relevant *Care Provider*;
- the user-friendly depiction of the *Data Service* (Name of Data Service) is obtained from the scope that the *Authorisation Server* received in the flow's very first step, which corresponds to the *Depiction Name* that is included with the relevant *Data Service* in the *Data Service Names List*;
- the user-friendly depiction of the identity of the *Publisher* (Name SupplierPGO) is obtained from the *OAuth Client List*, based on the redirect_uri (from OAuth) that was obtained in step 1.

Explanatory Notes

Name of Care Provider, Name of Data Service and Name SupplierPGO are placeholders, as included in the Consent Declaration and the Confirmation Declaration.

Addressing and parameters

Explanatory Notes

At four moments in the flow of UCI Compile, and in that of UCI Share, the OAuth roles address each other, based on a URI. The table below provides summarised details of these four moments. The address determinant is the OAuth role that determines the URI (here always the *OAuth Client*), the user is the OAuth role that applies the particular address. When the address user is the *OAuth User Agent* then the user is accordingly not the determinant and the relevant data transfer proceeds via what is called the front-channel. In the other two cases, the *OAuth Client* is both the determinant and user and the transfer proceeds via what is known as the back-channel.

usage moment	address determinant	address user	addressee	parameters
Authorisation request (step 1)	*OAuth Client* (step 1)	*OAuth User Agent*	*Authorisation Endpoint* of the *OAuth Authorisation Server*	• response_type • client_id • redirect_uri • scope • state
OAuth redirect (step 10)	*OAuth Client* (step 1)	*OAuth User Agent*	*OAuth Client*	
'access token' request (step 12)	*OAuth Client* (step 12)	*OAuth Client*	*Token Endpoint* of the *OAuth Authorisation Server*	• grant_type • code • **no** client_id • redirect_uri
FHIR request (step 14)	*OAuth Client* (step 14)	*OAuth Client*	*Resource Endpoint* of the *OAuth Resource Server*	

The responsibilities that now follow determine the structure of the URIs, which the address user's address determinant uses to address the addressee, and also determine how the parameters are filled in. The structure of the address is in all cases the same, but when it comes to the port numbers the responsibilities distinguish between front-channel and back-channel.

2. The *OAuth Client* compiles, in accordance with RFC 3986, the URI that they use to address either himself or the *OAuth Authorisation Server* or the *OAuth Resource Server*. The URI has a hostname that is a fully-qualified domain name , in accordance with RFC3696, section 2, and has the pattern scheme://host[:port] path, whereby:

- scheme is always https, in lowercase;
- host is a hostname in which
    - only the characters [a-z], [0-9], "." (full stop) and "-" (hyphen) occur;
    - each full stop separates two successive segments and is not part of either of the separated segments;
    - the first character of a segment is not a hyphen;
    - each segment is at least one character long;
    - the last segment is at least two characters long;
    - the last character must not be a hyphen;
    - it has a maximum of 255 characters;
    - it has at least two segments;
- path has the syntax of path-abempty from section 3.3 of RFC 3986 (and thus can be empty) but does not end with a /.

Explanatory Notes

The requirement that https is in lowercase follows the canonical form as specified in section 3.1 of RFC 3986. The requirements laid down for the hostname are based on (amongst others) RFC 952 and RFC 1123. The last segment is what is known as the 'top-level domain'.

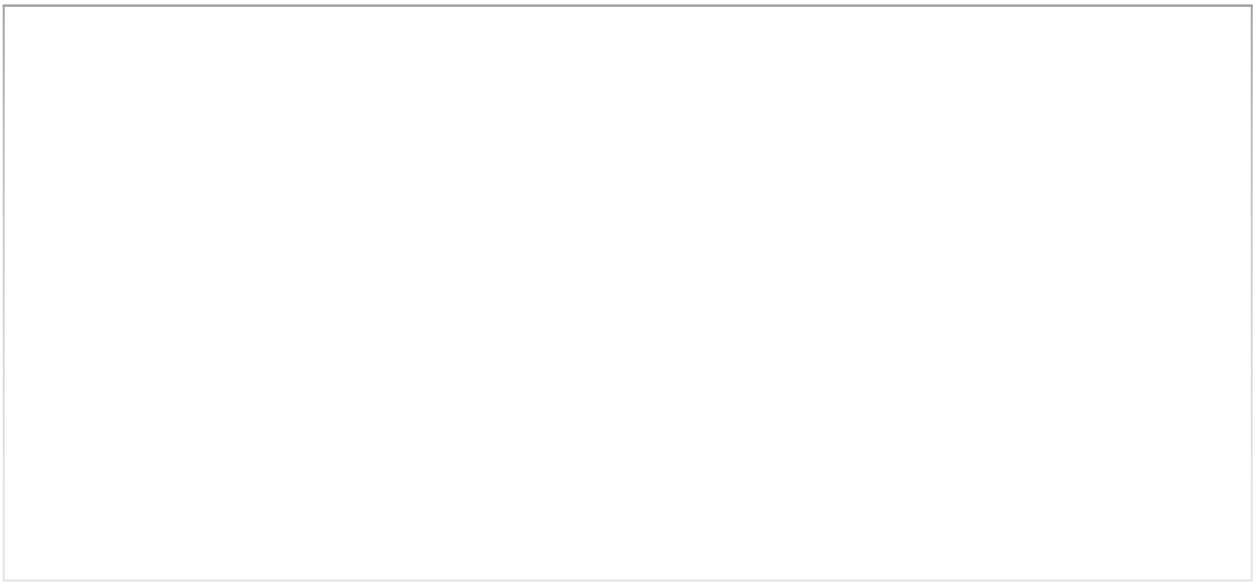
3a. If the address user is the *OAuth User Agent*

- then it is mandatory to use the port number intended for https that is included in the IANA Service Name and Transport Protocol Port Number Registry;
- and, if the addressee is the *Authorisation Endpoint* of the *OAuth Authorisation Server* then the *OAuth Client* (as address determinant) obtains the URI, including host and path, from the *Care Providers List*, based on the applicable *Care Provider* and *Data Service.*

3b. If the address user is the *OAuth Client* then the *OAuth Client* (as address determinant) obtains the first components of the URI, namely host, path and any port, from the *Care Providers List*, based on the applicable *Care Provider* and either the *Data Service* (if addressee is *OAuth Authorisation Server*) or the *System Role* (if addressee is *OAuth Resource Server*). Other elements of the general URI syntax, such as user, password, query and fragment, are absent from the addresses.

Explanatory Notes

The requirement that neither host norpath are allowed to end with a / makes it possible for the URI, especially in the fourth of the named moments, to be supplemented with URL endpieces that are specific to information standards, without the boundary with the MedMij initial entry becoming unrecognisable.



In other words, the *Care Providers List* is used by the *OAuth Client* to recognise the endpoint that fits with the applicable *Care Provider, Data Service* and - for the resource endpoint - *System Role*. This is why a single endpoint address must follow from a single such set. However, in the contrary situation this is not a requirement. It is possible to re-use endpoint addresses in any combination desired by the *Care Provider's Service Provider* for multiple such sets.

4. For a single *OAuth Authorisation Server*, the hostnames in the addresses for his *Authorisation Endpoint* and his *Token Endpoint* are identical.

Explanatory Notes

This responsibility is included in the light of the depiction in the Netwerklayer, of a single *Authorisation Server* on a single *ZA Node*. The *Resource Endpoint* may indeed be addressed by another hostname, because the *Resource Server* is another role. Don't forget - this responsibility only relates to the hostnames, not to all the endpoint addresses. The addresses of *Authorisation Endpoint* and *Token* Endpoint are in fact allowed to differ in respect of elements other than the hostname.

5. The parameters in the Authorisation request are filled in as follows:

parameter	**entry**	**explanatory notes**

parameter	entry	explanatory notes
response_type	literal value code	This is the result of responsibility 6 in the Application layer.
client_id	the same hostname of the *OAuth Client* that is also included in the *OAuth Client List*	
redirect_uri	such that the hostname included therein is the same as the client_id and no port number is included	See responsibility 3 above.
scope	mandatory, with: • the relevant (single) *Care Provider Name*, stripped of the suffix @medmij, followed by • a tilde (~), followed by • the *Data Service ID* of the relevant (single) *Data Service* from the *Data Service Names.*	The scope accordingly consists of two components in a specific order that are separated by a tilde. In the current version of the MedMij Framework, there must only be one of each. When the *Care Provider Name* is interpreted by the recipient, they will have to add the suffix @medmij again.
state	in accordance with section 4.1.1. of RFC 6749	The *OAuth Client* uses this to give information to the *OAuth Authorisation Server*, from which the former can subsequently (during the redirect) deduce which request the Authorisation code belongs to. This information is in other respects meaningless for the *OAuth Authorisation Server.*

6. The *OAuth Client* ensures that when it comes to the Authorisation request it is the http method GET that is used, not POST.

Explanatory Notes

In the OAuth specification, section 3.1 it is made mandatory for the Authorisation Server to accept GET, with POST being kept optional. Because GET is easily the most appropriate http

method for the authorisation request in the MedMij Framework, this responsibility applies in order to ensure that the *Authorisation Server* is not faced with unnecessary implementation costs. Although this responsibility is a responsibility of the *OAuth Client*, since this comes under the responsibility of a MedMij participant the request is ultimately executed by the *OAuth User Agent*.

7. The parameters in the access token request are filled in as follows:

parameter	**entry**	**explanatory notes**
grant_type	literal value Authorisation_code	This is the result of responsibility 6 in the Application layer.
code	in accordance with responsibility 10a-d in the Application layer	See the explanatory notes for responsibility 10a-d in the Application layer.
client_id	**not used**	This is not necessary because responsibility 12 in the Application layer guarantees that the access token is only provided to the *OAuth Client* to whom consent has been granted by the *OAuth Resource Owner* .
redirect_uri	the same value as in the previous Authorisation request	

Performance

8. After receiving an access token request in *UCI Compile* or *UCI Share*, the *Authorisation Server*will - if an access token must be issued in response - provide this access token to the *PGO Server* after a maximum of ten (10) seconds. This behaviour of the *Authorisation Server* is available at least 99.5% of the time.

9. After receiving an FHIR request in *UCI Compile* or *UCI Share*, then - if an FHIR response must be given in response - the *Resource Server* will provide this FHIR response to the *PGO Server* after a maximum of sixty (60) seconds. This behaviour of the *Resource Server* is available at least 98.5% of the time.