2) Send the Overdracht to all Vendors

An addition to the Nuts Bolt process where if the same Customer is registered with multiple Vendors, a Task is created for each 'eOverdracht-Receiver' service at each Vendor, i.e. multiple tasks per customer.

Advantages:

- The receiving Vendors do not need knowledge of which other systems are used: each of the Vendors takes over the resources relevant to their role in the treatment: no Customer specific configuration required.
- There is no need for a central client administration where all patients are known, although the users of the various Vendors/systems then have to know whether or not to accept the client. Configuring multiple Customers as in 1) also remains an option.
- It doesn't require any modification to the Nictiz architecture as far as I can tell, just the process/agreements within Nuts.

Disadvantages:

- This only works if all sending parties support it. So the principle that there can be more than one 'eOverdracht-Receiver' Service for a Customer in the address book, and if this is the case that a separate Overdracht must be made for each Service, must then become part of the agreement system.
- For the sender it adds complexity if the Task status does not have the same status at different Vendors, for example one Vendor accepts it and the other does not.
- Each 'Vendor' in the Nuts address book defines its own list of 'Customers'. There must then be a label/identifier in the address book through which the sender can see that the Customer at Vendor A is the same as the Customer at Vendor B, and only show this Customer once in the search results for the user.

Note: A variation on this is the idea of creating one Task and sending a notification to the different Vendors. But I think this is problematic in terms of accepting and processing the file: which of the Vendors accepts the Task? The Nuts architecture and the Nuts authorization tokens allow this. If this can be solved with work processes within the Customer, this is a direction that can also be further investigated.

Revision #2
Created 2023-02-23 21:29:03 UTC by Valeriy Lazarev
Updated 2023-03-24 10:41:13 UTC by Valeriy Lazarev