EVS_3.x.y_Manual

Introduction

This manual describes EVS v3.3.x. (Best supported by JAVA8)

EVS is an evolution of EVSc, using gateway-integration instead of Vitalink connector-integration. It allows the manipulation of vault contents using specific actors and specific patients, manually or based on previously exported vault contents.

EVS 3.x.y expands on this by adding Sumehr and Diarynote integration.

Pre-configuration

Due to GDPR, EVS is not configured to work out-of-the-box after initial installation anymore.

Configuration needed to have EVS working in no time:

- an actor (with an active acc-key (for Vitalink))
- a patient

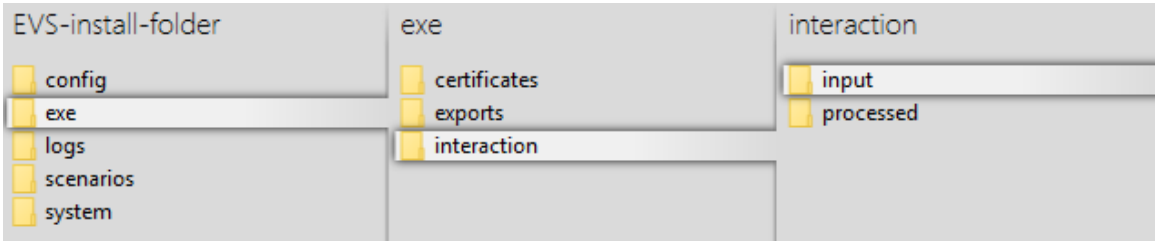
It is strongly advised to test EVS with this pre-configuration before changing extra configuration for your own needs. Once this test is executed successfully, one should start using his own configuration.

Functionality

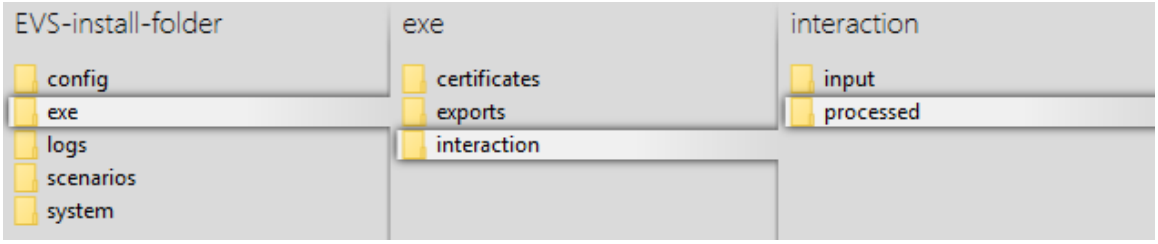
General

EVS allows a certain actor to perform a number of actions for a certain patient. These actions are different, depending on the transactiontype.

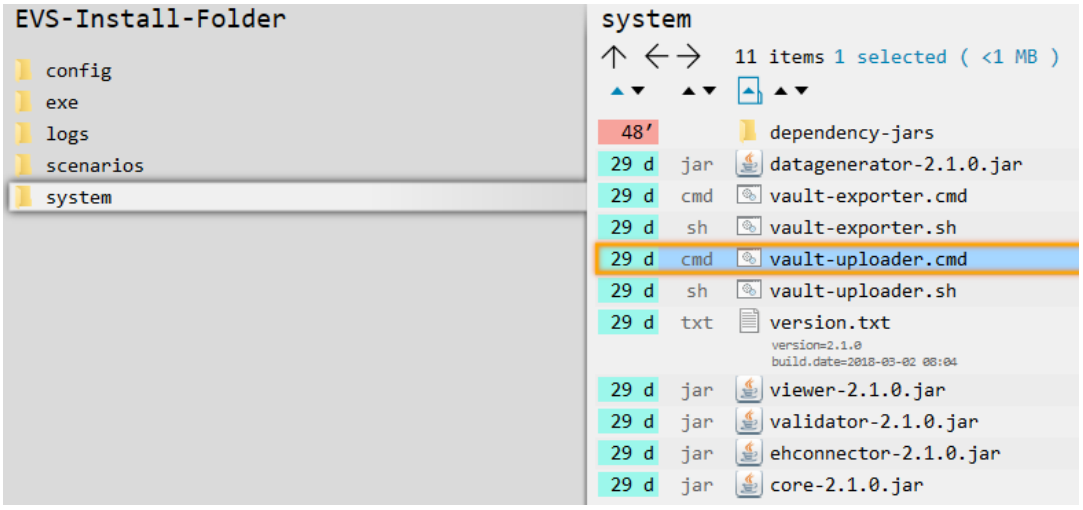
These actions are triggered by dropping input files in (subfolders of) the input-folder:



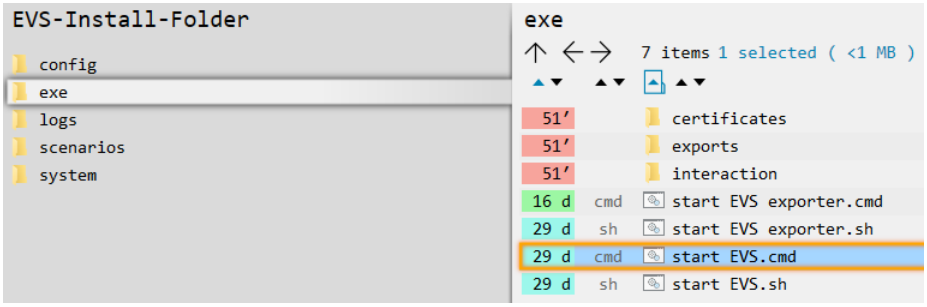
EVS watches these folder(s), executes the action(s) and generates output in the processed-folder:



EVS can be launched via the "vault-uploader.cmd" batch file:



The behaviour of EVS must be determined by passing some mandatory parameters. Instead of using the "vault-uploader.cmd" batch file, it is easier to use the example batch file "start EVS.cmd":



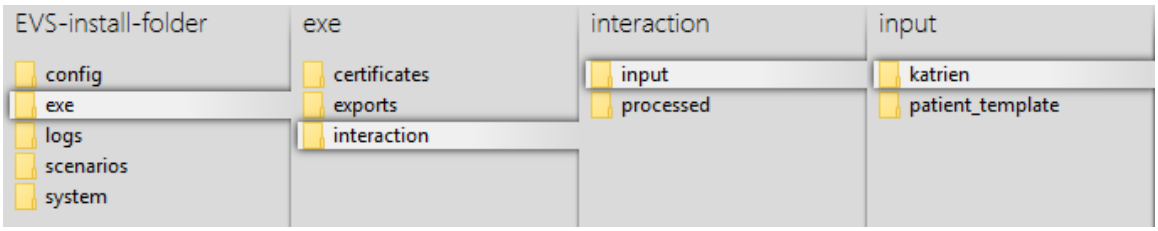
This batch file contains parameter values for a standard behaviour. How the parameters change the behaviour can be found in the paragraph Parameterisation.

Input-folder

Which patient?

The patient is determined by the folder under "..\exe\interaction\input".

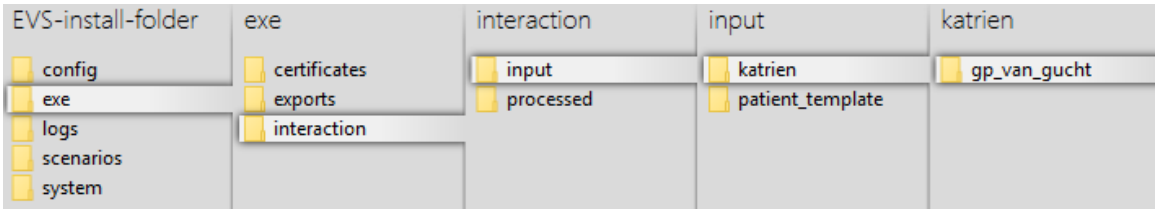
After initial installation, 1 patient named "katrien" is available:



Which actor?

The actor is determined by the folder under "..\exe\interaction\input\<patient folder>".

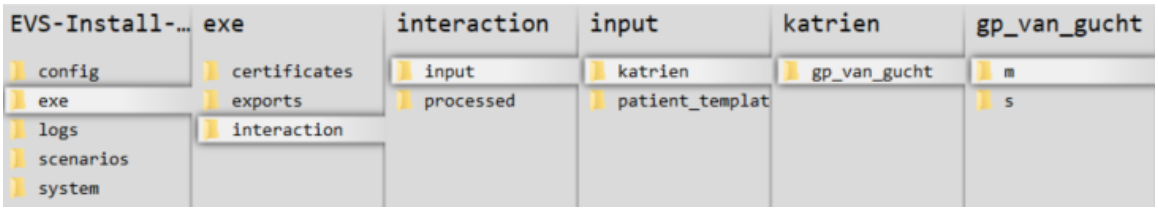
After initial installation, 1 actor named "gp_van_gucht" is available:



Which transactiontype?

The transactiontype is determined by the folder under "..\exe\interaction\input\<patient folder>\<actor folder>".

After initial installation, 3 types named "m", "s" and "d" (meaning "Medicationscheme", "Sumehr" and "Diarynote" respectively) are available:



Which files?

Any type of files, with any extension, can be dropped. They are considered as "input-files". EVS will, depending on the action folder, parse the files and extract the MSE, TS, Sumehr and Diarynote transactions.

> ⚠ All MS transactions are ignored as input.

Where to put the acceptation-environment-token from Vitalink ?

Since EVS went opensource we decided to not use a hard-coded acceptation-key anymore but a 'replaceable' key in the config which is read at run-time.

This key can be added in the (to be used) actor configuration file at the location as described hereabove in the topic 'Which actor?'.

Example:

Acceptation key location in actor config file

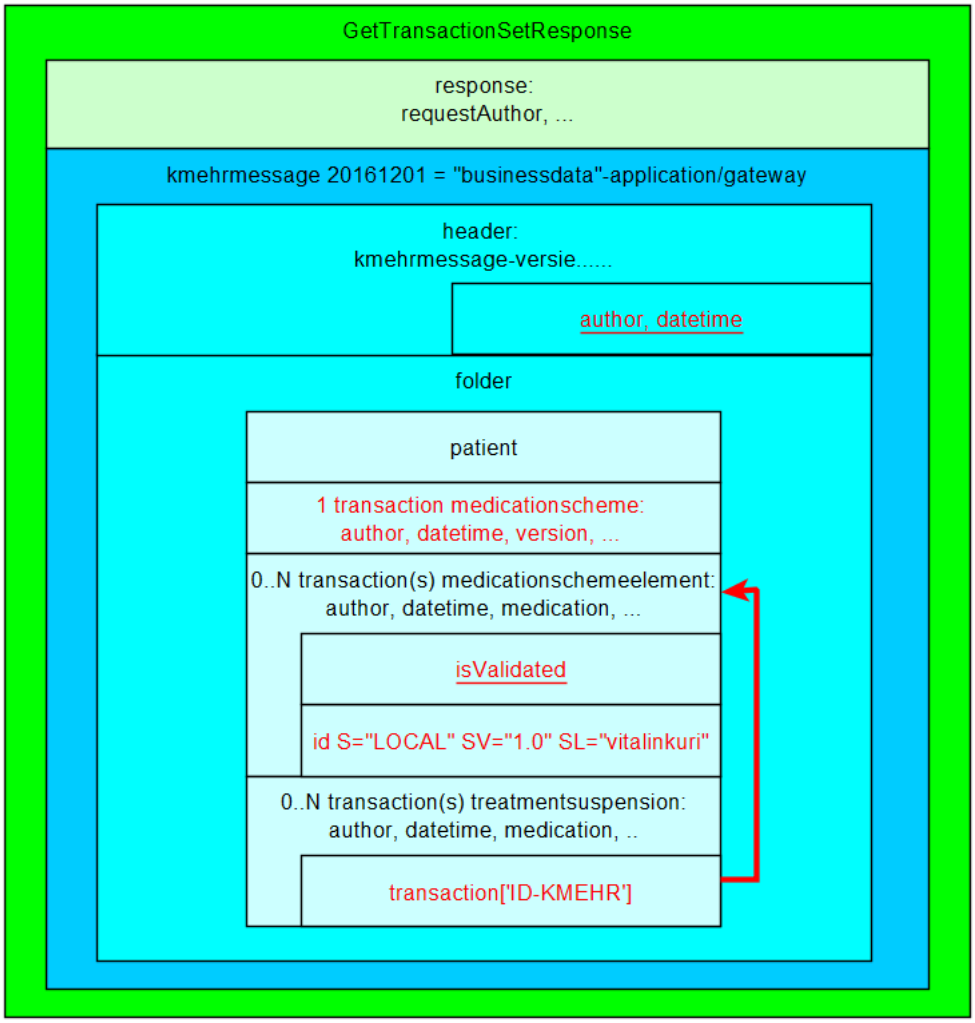
```
<authenticationConfiguration>
    <evs>
        <type>fallback</type>
        <certificates>
            <certificate>
                <type>identification</type>
                <path>..\config\certificates\SSIN=[INSS+SPECIAL CODE].acc-p12</path>
                <password>[PW]</password>
            </certificate>
        </certificates>
    </evs>
    <ehealth>
        <entry>kmehr.hubservicev3.software.id.local.value.1=[VITALINK ACC-KEY]</entry>
        <entry>user.inss=[INSZ]</entry>
        <entry>user.nihii=[NIHII]</entry>
        <entry>user.firstname=[FIRSTNAME]</entry>
        <entry>user.lastname=[LASTNAME]</entry>
        <entry>careprovider.inss=${user.inss}</entry>
        <entry>careprovider.nihii=${user.nihii}</entry>
        <entry>careprovider.firstname=${user.firstname}</entry>
        <entry>careprovider.lastname=${user.lastname}</entry>
...
```

What is a Kmehrmessage?

A Kmehrmessage is a part of the file that starts with <kmehrmessage ...> and ends with </kmehrmessage>. One file can contain 0 or more Kmehrmessages. One Kmehrmessage can contain either 0 or more MSE and TS transactions, or it can contain 0 or more Sumehr transactions.

EVS will work with Kmehrmessages of Kmehr-standard 20120401 and Kmehr-standard 20161201 as input.

All extra data needed for the communication with the gateway will be generated by the EVS. As input the data as depicted in the image below will be used:



How is a "medication" identified?

For some actions, typically removing and updating "medications", the medication that should be changed needs to have an identification. Those medications are expressed as MSE transactions. These MSE transactions are identified by using an EVS reference.

⚠ TS transactions can not be updated and should not contain EVS references! They can only be added or removed.

How is a "sumehr" identified?

For some actions, typically removing and updating "sumehrs", the sumehr that should be changed needs to have an identification. Those sumehrs are expressed as Sumehr transactions. These Sumehr transactions are identified by using an EVS reference.

How is a "Diarynote" identified?

For some actions, typically removing "diarynotes", the diarynote that should be changed needs to have an identification. Those sumehrs are expressed as Diarynote transactions. These Diarynote transactions are identified by using an EVS reference.

Identification by EVS reference

An EVS reference is put in 1 (and only 1) free text field in the concerned MSE, Sumehr or Diarynote transaction.

The EVS reference can be freely chosen, and facilitates the definition and execution of scenarios.

In the next example, this REF is "===EVSREF:901===". EVS detects the reserved format "===EVSREF:<any text>===" and finally uses "<any text>" as unique REF.

The REF must contain a minimum of 3 characters and can contain up to 512 characters, so both ===EVSREF:123=== and ===EVSREF:A123456789B123456789=== are considered valid REFs.

If multiple EVS REFs have been given for 1 MSE or Sumehr transaction, EVS will not execute the action and will raise an error.

```
<kmehrmessage ...>
    <header>
        <standard>
            <cd S="CD-STANDARD" SV="1.4">20161201</cd>
        </standard>
        ...
    </header>
    <folder>
        <id S="ID-KMEHR" SV="1.0">1</id>
        <patient>
            <id S="ID-PATIENT" SV="1.0">83051839468</id>
            <firstname>Katrien</firstname>
            <familyname>Van Gucht</familyname>
            <sex>
                <cd S="CD-SEX" SV="1.0">female</cd>
            </sex>
        </patient>
        <transaction>
            ...
            <cd S="CD-TRANSACTION" SV="1.4">medicationschemeelement</cd>
            ...
            <item>
                <id S="ID-KMEHR" SV="1.0">2</id>
                <cd S="CD-ITEM" SV="1.4">medication</cd>
                <content>
                    <medicinalproduct>
                        <intendedcd S="CD-DRUG-CNK" SV="2010-07">2933901</intendedcd>
                        <intendedname>Dafalgan bruistab 20x 500mg</intendedname>
                    </medicinalproduct>
                </content>
                ...
                <instructionforpatient L="nl">===EVSREF:901=== Actual instruction for patient.</instructionforpatient>
            </item>
        </transaction>
    </folder>
</kmehrmessage>
```

Which actions for Medicationscheme?

Depending on the folder where the input-file is dropped, EVS will execute an action.

> Only transactions of transactiontype "Medicationscheme" can be used for input. Any other transactiontype will be ignored.

To be translated:

Actions

ADD voegt nieuwe lijnen toe, dus daar sturen we nog geen vitalink URI mee waardoor de kluis het als een 'nieuwe' lijn beschouwt,

ADD vraagt achterliggend ook eerst de hele kluisinhoud op, zodat we de nieuwe lijn kunnen toevoegen aan de bestaande kluisinhoud want alles wat je niet zou meesturen wordt gewist uit vitalink (zoals updateREF)

REPLACE doet (als ik me niet vergis) eerst een empty van de kluis door een puttransactionsetrequest te doen zonder medicatielijnen. Vervolgens een tweede call met daarin de lijnen van het inputbestand

UPDATEREF vraagt ook eerst alles op, en tracht dan de medicatielijn(en) uit het inputbestand op te zoeken in Vitalink, om vervolgens die lijnen als gewijzigd te markeren in uitgaande bericht

Action "add"

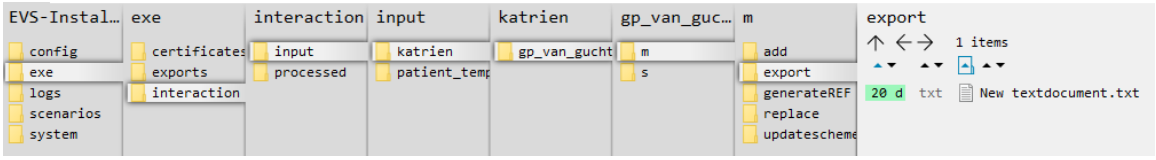
This action will add a transaction to the vault for all transactions found in each Kmehrmessage found in all dropped files. If one of the MSE transactions within a kmehrmessage does not have an EVS REF yet, an EVS REF will be generated.

In the example below, 3 transactions are dropped to be added to the vault:

Action "export"

This action will export the contents of the vault that belong to transactiontype "Medicationscheme", without any change to the vault itself. EVS will do this action once for each dropped file, without parsing this file.

In the example below, a newly created file will trigger an export of the contents of the vault:



Action "generateREF"

This action will generate an EVS REF for each MSE transaction currently in the vault.

> ⚠ The 'old' EVS, with Vitalink connector-integration, added the references to the input file, followed by putting this in the vault.

If an EVS REF exists already in the MSE transaction, no new EVS REF will be generated.

If no EVS REF exists, the new EVS REF is put in the instructionforpatient field.

Action "replace"

This action will replace the contents of the vault that belong to transactiontype "Medicationscheme" by all the transactions found in the input file. Be aware of the fact that dropping multiple files in the replace-folder will result in a vault with as contents the transactions of the last input file! If one of the MSE transactions within a file does not have an EVS REF yet, this will be generated.

If you want to 'empty' the medicationscheme in the vault. Drop an empty textfile OR a valid XML without any MSE in the REPLACE folder.

In the next example, after processing the next 3 input files, the vault contains 4 transactions.

Action "updateschemeREF"

This action will update the complete contents of the vault that belong to transactiontype "Medicationscheme", based on the input file compared with the current contents of the vault.

The next actions will take place:

- an MSE transaction with EVS REF not yet existing in the vault will be added to the vault, together with all TS transactions which are linked with this MSE transaction
- an MSE transaction with EVS REF already existing in the vault will cause an update if any difference between the input-transaction and vault-transaction is found
- all MSE transactions without corresponding EVS REF in the input file will be removed from the vault

If the input-file contains MSE transactions with EVS REFs that are not unique, the action will not be executed and an error will be thrown.

Action "updateREF"

Looking a lot like updateSchemREF, this action will update 1 or more targetted MSE transactions within the transactiontype MedicationScheme on the vault without touching the other MSE's, not included in the source file. So the source file does NOT include the complete MS with updated + non-updated MSE's but an MS with only the MSE's you want to alter.

An ERROR status is thrown in following cases:

- 1 or more of the included MSE in the MS of the sourcefile contains an EVSREF that was not found in the MS on the vault, no update will happen untill corrected
- 1 or more of the included MSE in the MS of the sourcefile does not contain an EVSREF, no update will happen untill corrected

If no difference is found between input MS and vault MS, no update will happen.

The next actions will take place:

- If no ERROR is detected in the source file, before executing the update (PutTransaction), a list is provided in the EVS console of all MSE's entries with their maching EVSREF's found within the targetted MS on the vault, together if it will be updated or not.

```
MS entry ===EVSREF:100=== will not be updated
MS entry ===EVSREF:101=== will not be updated
MS entry ===EVSREF:102=== will be UPDATED
MS entry ===EVSREF:103=== will not be updated
MS entry ===EVSREF:104=== will not be updated
MS entry ===EVSREF:105=== will be UPDATED
MS entry ===EVSREF:106=== will not be updated
MS entry ===EVSREF:107=== will not be updated
MS entry ===EVSREF:108=== will not be updated
MS entry ===EVSREF:109=== will not be updated
MS entry ===EVSREF:110=== will not be updated
MS entry ===EVSREF:111=== will not be updated
MS entry ===EVSREF:112=== will not be updated
MS entry ===EVSREF:113=== will not be updated
MS entry ===EVSREF:114=== will not be updated
MS entry ===EVSREF:115=== will not be updated
MS entry ===EVSREF:116=== will not be updated
MS entry ===EVSREF:117=== will not be updated
MS entry ===EVSREF:118=== will not be updated
MS entry ===EVSREF:119=== will not be updated
```

- an MSE transaction with EVSREF already existing in the vault will be updated if any difference between the input-MSE transaction and vault-MSE transaction is found and versionnr of MSE + MS will be +1

Empty

The 'old' EVS, with Vitalink connector-integration, offered an action 'empty'. EVS doesn't offer this action anymore. Now, emptying the vault can be done by dropping an empty file for the actions "updateschemeREF" or "replace".

Which actions for Sumehr?

Only transactions of transactiontype "Sumehr" can be used for input. Any other transactiontype will be ignored.

Action "add"

This action will add a transaction to the vault for all transactions found in each Kmehrmessage found in all dropped files. If one of the Sumehr transactions within a kmehrmessage does not have an EVS REF yet, an EVS REF will be generated.

In the example below, 3 transactions are dropped in the add folder to be added to the vault:

Action "empty"

This action will remove all transactions from the vault that belong to transactiontype "Sumehr". EVS will do this action once for each dropped file, without any parsing.

In the example below, a newly created file will trigger emptying of the vault by removing all existing transactions of transactiontype "Sumehr":

Action "export"

This action will export the contents of the vault that belong to transactiontype "Sumehr", without any change to the vault itself. EVS will do this action once for each dropped file, without parsing this file.

In the example below, a newly created file will trigger an export of the contents of the vault of transactiontype "Sumehr":

Action "generateRef"

This action will generate an EVS REF for each Sumehr transaction currently in the vault.

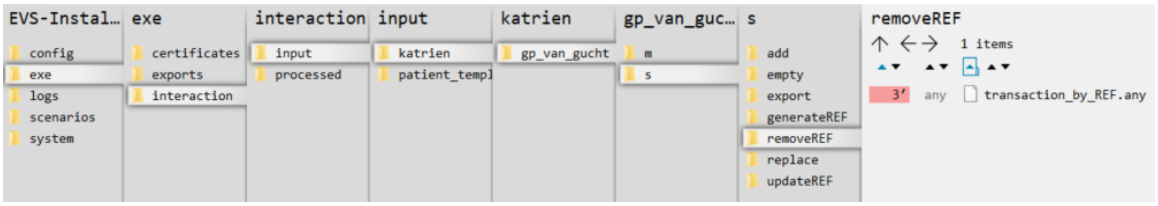
> ⚠ The 'old' EVS, with Vitalink connector-integration, added the references to the input file, followed by putting this in the vault.

If an EVS REF exists already in the Sumehr transaction, no new EVS REF will be generated.

If no EVS REF exists, the new EVS REF is put in the first text field.

Action "removeRef"

This action will remove the transactions that belong to transactiontype "Sumehr" identified by the EVS REF in the input file from the vault.



Action "replace"

This action will replace the contents of the vault that belong to transactiontype "Sumehr" by all the transactions found in the input file. Be aware of the fact that dropping multiple files in the replace-folder will result in a vault with as contents the transactions of the last input file! If one of the Sumehr transactions within a file does not have an EVS REF yet, this will be generated.

In the next example, after processing the next 3 input files, the vault contains 4 transactions.

Action "updateRef"

This action will update the transactions that belong to transactiontype "Sumehr" identified by the EVS REF in the input file.

Which actions for Diarynote?

> ⓘ Only transactions of transactiontype "Diarynote" can be used for input. Any other transactiontype will be ignored.

Action "add"

This action will add a transaction to the vault for all transactions found in each Kmehrmessage found in all dropped files. If one of the Diarynote transactions within a kmehrmessage does not have an EVS REF yet, an EVS REF will be generated.

Action "export"

This action will export the contents of the vault that belong to transactiontype "Diarynote", without any change to the vault itself. EVS will do this action once for each dropped file, without parsing this file.

Action "generateRef"

This action will generate an EVS REF for each Diarynote transaction currently in the vault.

If an EVS REF exists already in the Diarynote transaction, no new EVS REF will be generated.

If no EVS REF exists, the new EVS REF is put in the first Text or TextWithLayout field. If none of these exists, a new TextWithLayout field is created.

Action "updateRef"

This action will update the transactions that belong to transactiontype "Diarynote" identified by the EVS REF in the input file.

Which actions for PopulationBasedScreening?

Action "export"

This action will export the contents of the vault that belong to transactiontype "PopulationBasedScreening", without any change to the vault itself. EVS will do this action once for each dropped file, without parsing this file.

Which actions for ChildRecord?

Action "export"

This action will export the contents of the vault that belong to transactiontype "ChildRecord", without any change to the vault itself. EVS will do this action once for each dropped file, without parsing this file.

Which actions for Vaccination?

Action "export"

This action will export the contents of the vault that belong to transactiontype "Vaccination", without any change to the vault itself. EVS will do this action once for each dropped file, without parsing this file.

An overview pdf is generated automatically at the end of the export, visualizing all vaccinations of this patient in 1 list.

Processed-folder

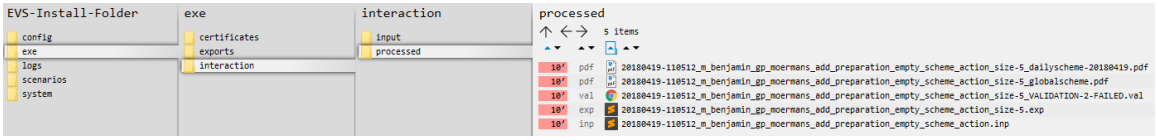
After execution of an action a variable number of output files are generated in the processed folder.

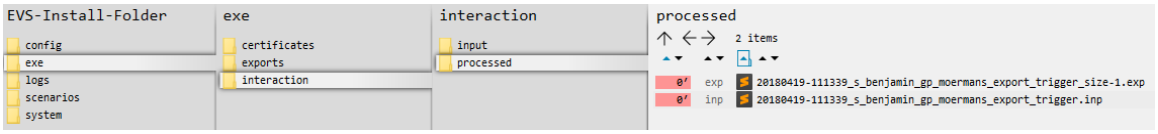
The next 2 screenshots show the output of a successful action.

The first screenshot shows the output of a successful add-action for transactiontype "Medicationscheme".

The second screenshot shows the output of a successful export-action for transactiontype "Sumehr".

Each output file will be explained below:





Each filename of transactiontype "Medicationscheme" exists out of:

<date>-<time>_<transactiontype>_<patient>_<actor>_<action>_<input filename>_size-<nr of MSE transactions>_<output suffix>.<output extension>

Name	Output suffix	Extension	Description	Remarks
Validation file	VALIDATION-OK VALIDATION-<###>-FAILED	.val	The report of the validation.	The filename contains the number of warnings and errors when the validation fails.
Global scheme PDF	globalscheme	.pdf	A visualisation of the global scheme in PDF format.	-
Daily scheme PDF	dailyscheme-<date>	.pdf	A visualisation of the daily scheme in PDF format.	This is the scheme of the medication that should be taken today.
Export file	-	.exp	An export of the contents belonging to transactiontype "Medicationscheme" of the vault.	-
Input file	-	.inp	The original input file.	The filename does not include the number of transactions in the vault.
Gateway scheme PDF (from 3.4.0 onwards)	gatewayscheme	.pdf	A visualisation of the global scheme in PDF format, generated by the gateway. Only generated in case EVS connects to the Vitalink Gateway.	-

Each filename of transactiontype "Sumehr" exists out of:

<date>-<time>_<transactiontype>_<patient>_<actor>_<action>_<input filename>_size-<nr of Sumehr transactions>.<output extension>

Name	Extension	Description	Remarks
Export file	.exp	An export of the contents belonging to transactiontype "Sumehr" of the vault	-
Input file	.inp	The original input file.	The filename does not include the number of transactions in the vault.

Each filename of transactiontype "Diarynote" exists out of:

<date>-<time>_<transactiontype>_<patient>_<actor>_<action>_<input filename>_size-<nr of Diarynote transactions>.<output extension>

Name	Extension	Description	Remarks
Export file	.exp	An export of the contents belonging to transactiontype "Diarynote" of the vault	-
Input file	.inp	The original input file.	The filename does not include the number of transactions in the vault.

If for some reason the action fails, an error output file is generated:

Name	Extension	Description	Remarks
Error file	.err	The report containing the error.	The content of the error file will identify the problem.

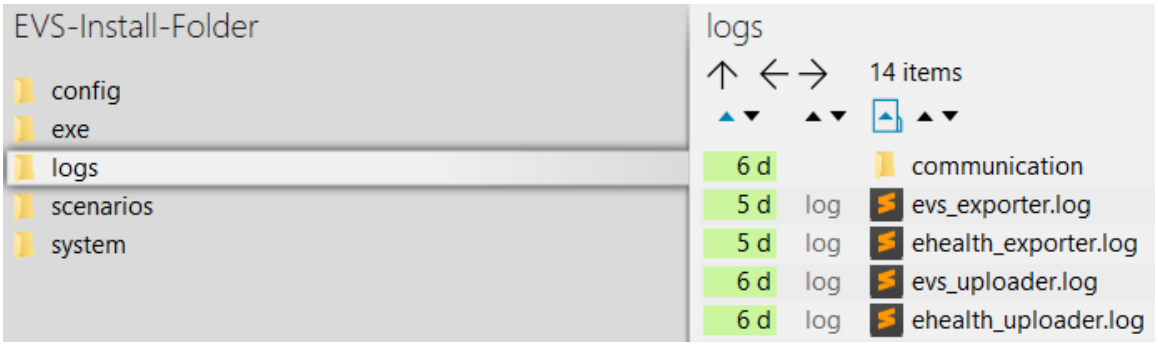
Logs-folder

Root

This folder will be automatically generated once "start EVS.cmd" or "start EVS exporter.cmd" has been run for the first time.

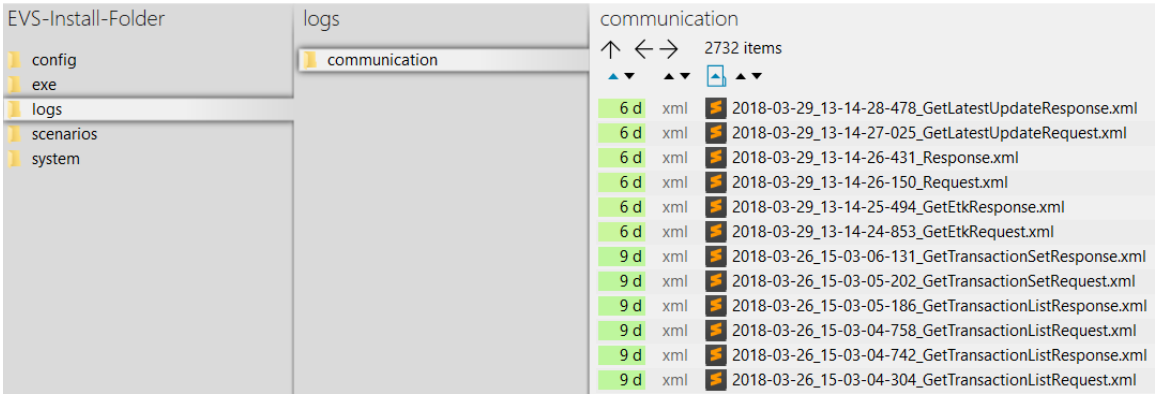
This folder contains:

- the eHealth technical connector logs for EVS: ehealth_uploader*.log
- the eHealth technical connector logs for EVS-exporter: ehealth_exporter*.log
- the proprietary EVS log for EVS: evs_uploader.log
- the proprietary EVS log for EVS-exporter: evs_exporter.log
- a folder 'communication'



Communicaton-folder

This folder contains all the requests and responses done by EVS when communicating with the gateway.

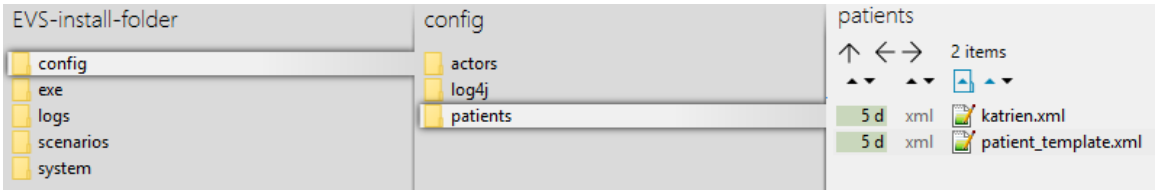


Configuration

This paragraph explains how to configure EVS.

How to add a patient?

Extra patients can be added by creating files in the next folder:



After initial installation, 2 patient config files are already present. Both can be used as example (copy-paste) to add extra patients. The name of the file, without the extension, needs to be used to identify for which patient the action needs to performed.

After copy paste of the example files, insert the correct info for the new patient:

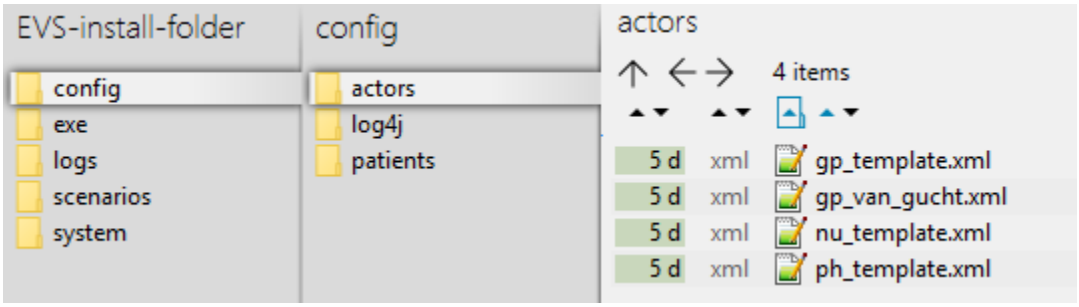
```
<patient>
    <id>93051822361</id>
    <firstName>Example Patient Firstname</firstName>
    <lastName>Example Patient Lastname</lastName>
    <gender>female</gender>
    <birthDate>1993-05-18</birthDate>
    <usualLanguage>nl-BE</usualLanguage>
</patient>
```

Since EVS follows all the rules for eHealth and Vitalink, it is up to the user to make sure the proper eHealth dependencies (informed consent, therapeutic relation, ...) are set in function of the wanted behaviour.

> **Restart EVS**
>
> EVS (and EVS-exporter) should be restarted when newly added patients will be used.

How to add an actor?

Extra actors can be added by creating files in the next folder:



> The 'old' EVS, with Vitalink connector-integration, used another syntax in this files. EVS is not compatible with this old format!

After initial installation, some actor config files are already present. All can be used as example (copy-paste) to add extra actors. The name of the file, without the extension, needs to be used to identify by which actor the action needs to be performed.

The template examples are given for 5 different types of actors: doctor, nurse and pharmacy, hospital, retirement home

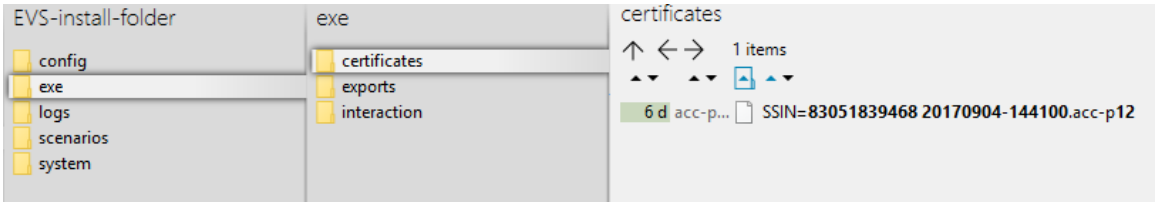
After copy paste of the appropriate example file, insert the correct info for the new actor:

```
<authenticationConfiguration>
    <evs>
        <type>fallback</type>
        <certificates>
            <certificate>
                <type>identification</type>
                <path>..\exe\certificates\[NAME_OF_ACC_P12_CERTIFICATE_FILE]</path>
                <password>[CERTIFICATE_PASSWORD]</password>
            </certificate>
        </certificates>
    </evs>
    <ehealth>
        <entry>user.inss=[SSIN]</entry>
        <entry>user.nihii=[NIHII11]</entry>
        <entry>user.firstname=[FIRSTNAME]</entry>
        <entry>user.lastname=[LASTNAME]</entry>
        <entry>careprovider.inss=${user.inss}</entry>
        <entry>careprovider.nihii=${user.nihii}</entry>
        <entry>careprovider.firstname=${user.firstname}</entry>
        <entry>careprovider.lastname=${user.lastname}</entry>
        <entry>main.kmehr.quality=persphysician</entry>
        <entry>kmehr.default.hcpartylist=identifier</entry>
        <entry>kmehr.hubservicev3.hcpartylist=identifier,software</entry>
        <entry>kmehr.hubservicev3.identifier.id.inss.value=${user.inss}</entry>
        <entry>kmehr.hubservicev3.identifier.id.inss.sv=1.0</entry>
        <entry>kmehr.hubservicev3.identifier.id.idhcparty.value=${user.nihii}</entry>
        <entry>kmehr.hubservicev3.identifier.id.idhcparty.sv=1.0</entry>
        <entry>kmehr.hubservicev3.identifier.cd.cdhcparty.value=persphysician</entry>
        <entry>kmehr.hubservicev3.identifier.cd.cdhcparty.sv=1.1</entry>
        <entry>kmehr.hubservicev3.identifier.firstname=${user.firstname}</entry>
        <entry>kmehr.hubservicev3.identifier.lastname=${user.lastname}</entry>
        <entry>kmehr.hubservicev3.identifier.address.cd.sv=1.0</entry>
        <entry>kmehr.hubservicev3.identifier.address.cd.value=</entry>
        <entry>kmehr.hubservicev3.identifier.address.cd.schemes=cdaddress</entry>
        <entry>kmehr.hubservicev3.identifier.address.nis=24062</entry>
        <entry>sessionmanager.samlattribute.1=urn:be:fgov:identification-namespace,urn:be:fgov:ehealth:1.0:certificateholder:person:ssin,${user.inss}</entry>
        <entry>sessionmanager.samlattribute.2=urn:be:fgov:identification-namespace,urn:be:fgov:person:ssin,${user.inss}</entry>
        <entry>sessionmanager.samlattributedesignator.1=urn:be:fgov:identification-namespace,urn:be:fgov:ehealth:1.0:certificateholder:person:ssin</entry>
        <entry>sessionmanager.samlattributedesignator.2=urn:be:fgov:identification-namespace,urn:be:fgov:person:ssin</entry>
        <entry>sessionmanager.samlattributedesignator.3=urn:be:fgov:certified-namespace:ehealth,urn:be:fgov:person:ssin:doctor:boolean</entry>
        <entry>sessionmanager.samlattributedesignator.4=urn:be:fgov:certified-namespace:ehealth,urn:be:fgov:person:ssin:ehealth:1.0:doctor:nihii11</entry>
    </ehealth>
</authenticationConfiguration>
```

The needed info in the above template is:

- the certificate name e.g. 'xxx.p12'
- the password of the certificate,
- the SSIN,
- the NIHII number
- the actor's name

The location of the certificate can be freely chosen, but it is good practice to put it in the same folder as the example by installation:



Since EVS follows all the rules for eHealth and Vitalink, it is up to the user to make sure the proper eHealth dependencies (therapeutic relation, ...) are set in function of the wanted behaviour.

> **Restart EVS**
>
> EVS (and EVS-exporter) should be restarted when newly added actors will be used.

Parameterisation

The next parameters can be passed when launching EVS:

Name	Values	Meaning
rootdir	".. \exe\interaction"	Relative or absolute path to the folder that needs to be watched by EVS. This folder should contain the requested actions.

writeAsIs	true|false	false: All patient data from the source Kmehrmessage will be replaced by the corresponding data of the used input patient. Since the Kmehr data model is used for this transformation, it is possible that other Kmehr structure elements are slightly changed too. true: The Kmehrmessage will be sent to the vault untouched. Use this when really no manipulation on the source Kmehrmessage is desired.
exportAfterUpload	true|false	true: Each action, excepted "export" itself, should be followed by an export. false: No export is needed after execution of the triggered action.
validateExportAfterUpload	true|false	true: Each action should be followed by validation of the vault content. false: No validation is needed.
generateGlobalMedicationScheme	true|false	true: Each action should be followed by the generation of a global scheme visualisation PDF. false: No global scheme visualisation is needed.
generateDailyMedicationScheme	true|false	true: Each action should be followed by the generation of a daily scheme visualisation PDF. false: No daily scheme visualisation is needed.
generateGatewayMedicationScheme (from 3.4.0 onwards)	true|false	true: Each action should be followed by the generation of a global scheme visualisation by the gateway. false: No global scheme visualisation by the gateway is needed.
dailyMedicationSchemeDate	date ("yyyy-MM-dd")	If no date has been given, it will generate a daily medication scheme of the current date. If a date has been given, it will generate a daily medication scheme of the given date.
startTransactionId	number	This number will be the number for the first transaction within the kmehrmessage of a putTransactionSetRequest in the context of a medicationscheme.
shiftAction	no_tag_and_no_shift tag_and_no_shift shift_and_tag	See Appendix C: Parameter shiftAction
hub (from 3.5.0 onwards)	VITALINK RSW RSB (from 3.6.0 onwards)	The Hub to send requests to.
searchType (from 3.5.0 onwards)	LOCAL GLOBAL	LOCAL: instruct the hub to search only locally. GLOBAL: instruct the hub to search also elsewhere. Only applicable after the hubs will have been linked which is not the case at the time of writing (23/08/2018)

Example of a parameterisation:

start EVS.cmd

```
1  ../system/vault-uploader.cmd -rootDir="..\exe\interaction" -writeAsIs=false -exportAfterUpload=true
   -validateExportAfterUpload=true -generateGlobalMedicationScheme=true  -generateDailyMedicationScheme=false
```

Appendix A: Folder structure EVS 2.x.y

This paragraph gives a brief overview of the folder structure after initial installation. It can be used as reference while using and configuring EVS.

Path								Reserved path	Reserved name	Explanation
EVS								✗	✗	The root folder. The name and location can be freely chosen. Keep in mind that paths used in scenarios, patient and actor files are possibly impacted by changes to this!
	\config							✓	✓	Everything that defines the behaviour of EVS, configured as needed by the user.
		\actors						✓	✓	All the actors that can be used by EVS.
		\log4j						✓	✓	Settings of the log4j library. Please refer to https://logging.apache.org/log4j/2.x/manual/configuration.html for more explanation.
		\patients						✓	✓	All the patients that can be used by EVS.
	\exe							✓	✓	
		\certificates						✗	✗	The certificates used in the actor configuration files.
		\exports						✗	✗	The folder were the EVS-exporter will put exported vault contents, see Appendix B: EVS-exporter
		\interaction						✗	✗	
			\input					✓	✓	
				\katrien				✓	✗	
					\gp_van_gucht			✓	✗	
						\m		✓	✓	This folder contains the actions for transactiontype "Medicationscheme"
							\add	✓	✓	
							\export	✓	✓	
							\generateREF	✓	✓	
							\replace	✓	✓	
							\updateschemeREF	✓	✓	
						\s		✓	✓	This folder contains the actions for transactiontype "Sumehr"
							\add	✓	✓	
							\empty	✓	✓	
							\export	✓	✓	
							\generateREF	✓	✓	
							\removeREF	✓	✓	
							\replace	✓	✓	
							\updateREF	✓	✓	
						\d	add	✓	✓	This folder contains the actions for transactiontype "Diarynote"

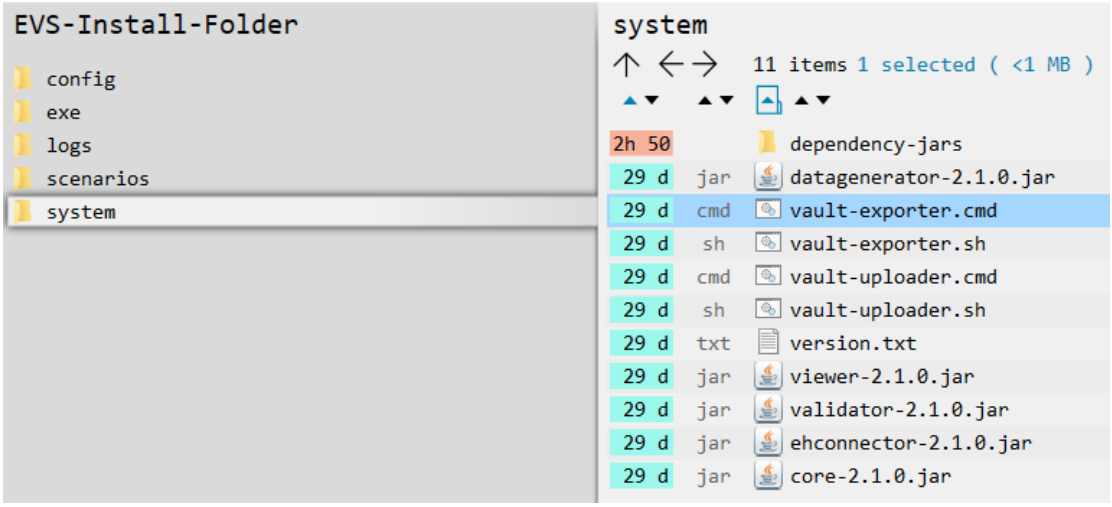
							export	✓	✓	
							generateREF	✓	✓	
							updateREF	✓	✓	
				\patient_template				✓	✓	
			\processed					✓	✓	
	\logs							✗	✗	Can be configured through the log4j settings.
		\communication						✓	✓	
	\scenarios							✗	✗	
		\basic_example						✗	✗	
	\system							✓	✓	
		\dependency-jars						✓	✓	

Appendix B: EVS-exporter

Besides the interaction provided by dropping files in the input folder, EVS offers as extended functionality the continuous monitoring of the vault contents. This functionality is provided by EVS-exporter.

Launching

EVS-exporter can be launched via the "vault-exporter.cmd" batch file:



The behaviour of EVS-exporter must be determined by passing some mandatory parameters. Instead of using the "vault-exporter.cmd" batch file, it is easier to use the example batch file "start EVS exporter.cmd":

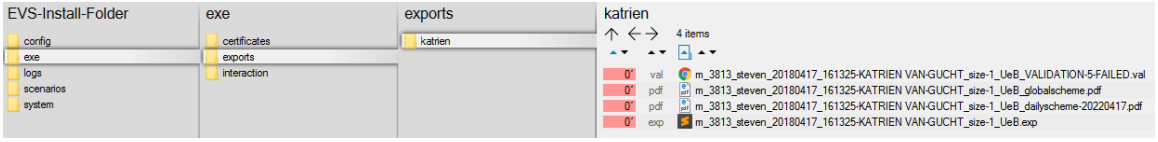
After initial installation, launching the EVS-exporter means that the vault contents of the patient "katrien" will be monitored.

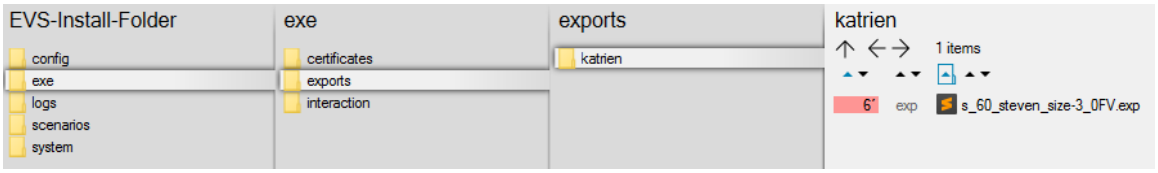
Output

When EVS-exporter detects for the given patients a change in the vault contents, an export is executed. The export is also executed after initial launch.

The exported files are put in the next folder, with for each monitored patient a subfolder. The subfolder is automatically created when the monitoring for this patient is initially started.

The files contain the same as the files generated by EVS in the processed-folder, but the filenames differ. As such, there are different naming conventions for "Medicationscheme", "Sumehr" and "Diarynote".





For EVS-exporter, each filename of transactiontype "Medicationscheme" exists out of:
<current-date>-<current-time>_<transactiontype>_<version>_<patient>_<date>_<time>-<author>_size-<nr of MSE transactions>_<unique code>_<output suffix>.<output extension>

Name	Source
current-date	Date of performing the export (from 3.3.0 onwards)
current-time	Time of performing the export (from 3.3.0 onwards)
Transactiontype	
Version	"Version" from the MS transaction. In case of an empty medicationscheme, the "version" is derived from the getLatestUpdate method.
Patient	Name of the patient as defined in the EVS configuration.
Date	Date of the latest update derived from the MS transaction.
Time	Time of the latest update derived from the MS transaction.
Author	"Author" of the latest update, derived from the MS transaction->UpdatedBy as returned by the gateway.
Nr of MSE transactions	The amount of MSE transactions in the vault.

Unique code	Code making the filename unique in case an export exists already.
Output suffix	Hard coded, depending on file type. For the validation file, the number of warnings and errors and possible failure are shown.
Output extension	Hard coded, depending on file type.

For EVS-exporter, each filename of transactiontype "Sumehr" exists out of:
<current-date>-<current-time>_<transactiontype>_<version>_<patient>_size-<nr of Sumehr transactions>_<unique code>.<output extension>

Name	Source
current-date	Date of performing the export (from 3.3.0 onwards)
current-time	Time of performing the export (from 3.3.0 onwards)
Transactiontype	
Version	"Version" from the MS transaction. In case of an empty medicationscheme, the "version" is derived from the getLatestUpdate method.
Patient	Name of the patient as defined in the EVS configuration.
Nr of Sumehr transactions	The amount of Sumehr transactions in the vault.
Unique code	Code making the filename unique in case an export exists already.
Output extension	Hard coded, depending on file type.

For EVS-exporter, each filename of transactiontype "Diarynote" exists out of:
<current-date>-<current-time>_<transactiontype>_<version>_<patient>_size-<nr of Diarynote transactions>_<unique code>.<output extension>

Name	Source
current-date	Date of performing the export (from 3.3.0 onwards)
current-time	Time of performing the export (from 3.3.0 onwards)
Transactiontype	
Version	"Version" from the MS transaction. In case of an empty medicationscheme, the "version" is derived from the getLatestUpdate method.
Patient	Name of the patient as defined in the EVS configuration.
Nr of Sumehr transactions	The amount of Diarynote transactions in the vault.
Unique code	Code making the filename unique in case an export exists already.
Output extension	Hard coded, depending on file type.

When the export fails, an error file will be generated, which is the same behaviour as for the folder-triggered export action of EVS.

Parameterisation

The next parameters can be passed when launching EVS-exporter:

Name	Values	Meaning

transactionType	"medicationscheme" "sumehr" "diarynote"	EVS-exporter supports 3 transactiontypes: medicationscheme, sumehr and diarynote. Change this value accordingly.
patients	name(s) as defined in EVS configuration, comma separated	This(these) is(are) the patient(s) that will be monitored and whose vault content(s) will be exported.
actor	name as defined in EVS configuration	This is the actor that will be used for exporting.
exportDir	default "..\exe\exports"	This is the path where the output files will be generated. This location can be freely chosen.
validate	true|false	true: Each export should be followed by validation of the vault content. false: No validation is needed.
generateGlobalMedicationScheme	true|false	true: Each action should be followed by the generation of a global scheme visualisation PDF. false: No global scheme visualisation is needed.
generateDailyMedicationScheme	true|false	true: Each action should be followed by the generation of a daily scheme visualisation PDF. false: No daily scheme visualisation is needed.
generateGatewayMedicationScheme	true|false	true: Each action should be followed by the generation of a global scheme visualisation by the gateway false: No global scheme visualisation by the gateway is needed.
dailyMedicationSchemeDate	date("yyyy-MM-dd")	If no date has been given, it will generate a daily medication scheme of the current date. If a date has been given, it will generate a daily medication scheme of the given date.
hub (from 3.5.0 onwards)	VITALINK RSW RSB (from 3.6.0 onwards)	The Hub to send requests to.
searchType (from 3.5.0 onwards)	LOCAL GLOBAL	LOCAL: instruct the hub to search only locally. GLOBAL: instruct the hub to search also elsewhere. Only applicable after the hubs will have been linked which is not the case at the time of writing (23/08/2018)

Example of a parameterisation:

start EVS exporter.cmd

```
1  ../system/vault-exporter.cmd -transactionType=medication-scheme -patients=katrien -actor=gp_van_gucht -exportDir=
   "..\exe\exports" -validate=true -generateGlobalMedicationScheme=true -generateDailyMedicationScheme=false
```

Appendix C: Parameter shiftAction

This paragraph gives an explanation about what shiftAction is and how to use it.

What is "shiftAction"?

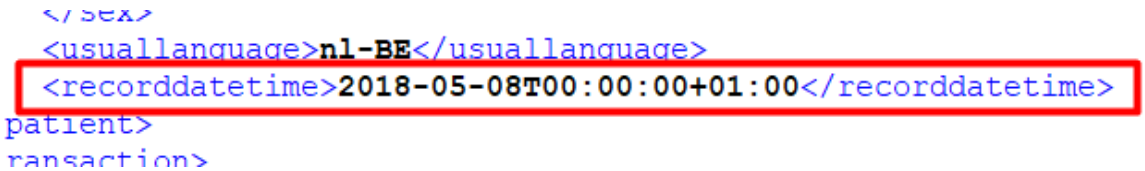
shiftAction is a parameter than can be passed when launching EVS.
Using shiftActions ensures that the dates of the tests are within a usable range. Not in the past and not too far in the future.
Because of this the dates of the tests don't need to be manually altered anymore.

How to use "shiftAction"?

shiftAction has three different values:

- tag_and_no_shift
- shift_and_tag
- no_tag_and_no_shift

Tag sets or updates a reference date as a "recorddatetime"-tag in the XML-file. This reference date is ALWAYS today's date.



Shift ensures that all other dates in the XML-file are adjusted in reference to the reference date.

tag_and_no_shift will set a new reference date to today's date or update an existing reference date.
This reference date will be added/updated in the XML-file as a "recorddatetime"-tag.
it will not change any other date in the XML-file.

shift_and_tag ensures that all other dates in the XML-file are adjusted in reference to today's date.
Afterwards it will update the "recorddatetime"-tag.

Suppose that today's date is 06 Jun 2018:

	Original date	Date after shift_and_tag
<recorddatetime>	25 May 2018	06 Jun 2018
transaction 1: tomorrow	26 May 2018	07 Jun 2018
transaction 2: yesterday	24 May 2018	05 Jun 2018
transaction 3: three days from now	28 May 2018	09 Jun 2018

Problem case:

Software vendor wants to alter all dates in the xml to other, more relevant dates.

For example, sw-vendor's application can only show the current week of the medicationscheme, but all dates in the XML are starting and ending in 2030.

Solution:

1. Enter a referencedate in the xmlrequest (underneath <usuallanguage> as seen in this chapter) to the first startmoment of the medication.
2. change startup parameter to shift_and_tag
3. send the xmlrequest to the vault

⚠ The "recorddatetime"-tag needs to exist in the XML-file prior to executing it, else it will not change the dates according to today's date but it will add a "recorddatetime"-tag.

no_tag_and_no_shift will not create or update the "recorddatetime"-tag and it will not adjust all other dates according to today's date.

⚠ If a "recorddatetime"-tag already exists in the XML-file and you run EVS with no_tag_and_no_shift, the tag will be removed.

e.g. -dailyMedicationSchemeDate="2016-01-13" -shiftAction=no_tag_and_no_shift

Appendix D: F.A.Q.

Q1: EVS could be started but error is shown of 'InvalidTherapeuticRelationException'

Description: Following error is show in the console output window:

```
            Caused by: org.imec.ivlab.ehconnector.hub.exception.incurable.
InvalidTherapeuticRelationException: Authorization failed: Invalid therapeutic relation for actor with
SSIN [83051839468 - 11425214001] on subject with SSIN [83051839468].
```

A1: Looks like there is a missing therapeutic relation between the actor and the patient, suggestions to resolve this issue:

- Make sure you run JAVA 8.
- duplicate the patient katrien folder in the /exe/interaction/input folder and rename it to your testpatient name (existing person with valid INSZ number aka rijksregisternummer).
- Log in with itsme or another CSAM tool and give your consent + confirm your therapeutic relation with doctor (arts): in this case 'katrien van gucht' at following website: https://wwwacc.ehealth.fgov.be/webconsent/consent/

Q2: