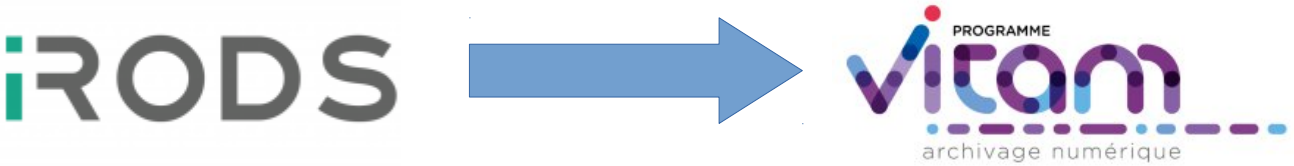
Using iRODS as an entry point to VITAM for long-term data preservation



Irods metadata :

Archived : False	→	Archived : True
Sent : False	→	Sent : True
X-Request-Id : Null	→	X-Request-Id : aopazieoaze

PROGRAMME vitam archivage numérique

RESIP

iRODS

Entry point

Post SIP

Response X-Request-Id

Get status

Response status

PROGRAMME vitam archivage numérique

New long-term preservation system at CINES

http://www.programmevitam.fr/pages/english/pres_english/

IRODS workflow presentation

- An archival agency submits a new object
- « Read » permission given to the « rods » user
- This object is then converted to a SEDA 2.1 archive with the Resip tool
- The initial object is deleted from iRODS
- Metadata ARCHIVED is set to « False »
- The SEDA 2.1 archive is sent to VITAM via its API (POST)
- VITAM replies with a X-Request-Id
- This request ID is stored into a metadata
- Metadata SENT is set to « True »
- A GET request is sent to the VITAM API in order to get the archive status
- If the reply contains « <ReplyCode>OK</ReplyCode> », the archiving process went well
- Metadata ARCHIVED is set to « True »
- The SEDA 2.1 archive is deleted from iRODS

Conversion to SEDA 2.1 format

- We used the Resip tool, which is part of the « sedatools » from VITAM: https://github.com/ProgrammeVitam/sedatools
- We compiled the Java code with Maven 3.6.3
- Configuration is done in ExportContext.config to set the SEDA 2.1 metadata in the manifest.xml file.

An excerpt from ExportContext.config

[...]
"archiveTransferGlobalMetadata" : {
"comment" : "**Test from Irods to Vitam**",
"date" : null,
"nowFlag" : true,
"messageIdentifier" : "**SIP herbarium image test from Irods**",
"archivalAgreement" : "**IN-MNHN-0**",
[…]
"transferRequestReplyIdentifier" : "**MNHN**",
"archivalAgencyIdentifier" : "**CINES**",
"archivalAgencyOrganizationDescriptiveMetadataXmlData" : null,
"transferringAgencyIdentifier" : "**CINES**",
"transferringAgencyOrganizationDescriptiveMetadataXmlData" : null
}

The archive.sh script

```
my_file=`echo $1 | cut -d "/" -f 5`
echo "file=$my_file" >> /tmp/output.txt
my_archive="$my_file.zip"
echo "archive=$my_archive" >> /tmp/output.txt

my_tmp_dir="/tmp/herbadrop/$my_file.tmp"
echo "tmp_dir=$my_tmp_dir" >> /tmp/output.txt

# Move to workdir

if [  ! -d $my_tmp_dir ]; then
mkdir -p $my_tmp_dir
fi

cd /tmp

# We fetch the file

/bin/iget $1 $my_tmp_dir

ls $my_tmp_dir >> /tmp/output.txt

# SEDA 2.1 conversion

java -jar /opt/test-sedatools/sedatools/resip/target/resip-2.3.0-SNAPSHOT-shaded.jar
-c /var/lib/irods/msiExecCmd_bin/ExportContext.config -d $my_tmp_dir -g $my_archive -i
-w /tmp/ -x

# The archive is registered into iRODS

/bin/iput -R access $my_archive
```

The vitam.sh script

```
#!/bin/bash

my_archive=`echo $1 | cut -d "/" -f 5`
echo "My Vitam archive is: $my_archive" >> /tmp/output.txt

cd /tmp

curl -k -X POST -H 'X-Tenant-Id: 8' -H 'X-Access-Contract-Id: IN-MNHM-8' -H 'X-Context-Id:
DEFAULT_WORKFLOW' -H 'Content-Type: application/octet-stream' -H 'X-Action: RESUME' -H
'X-SSL-CLIENT-CERT: […] --data-binary @$my_archive -i https://10.100.129.47:8443/ingest-
external/v1/ingests
```

The get.sh script

```
#!/bin/bash

my_archive=`echo $1 | cut -d "/" -f 5`
echo "My Vitam archive is: $my_archive" >> /tmp/output.txt

x_request_id=`imeta ls -d $my_archive X-Request-Id | grep value | cut -d " " -f 2`
echo "X-Request-Id for GET is: $x_request_id" >> /tmp/output.txt

curl -X GET -k -H 'X-Tenant-Id: 8' -H 'X-Access-Contract-Id: IN-MNHN-0' -H 'X-SSL-CLIENT-CERT:
[…] -H 'Content-Type: application/octet-stream' -H 'Accept: */*'
-i "https://10.100.129.47:8443/ingest-external/v1/ingests/$x_request_id/archivetransferreply"
```

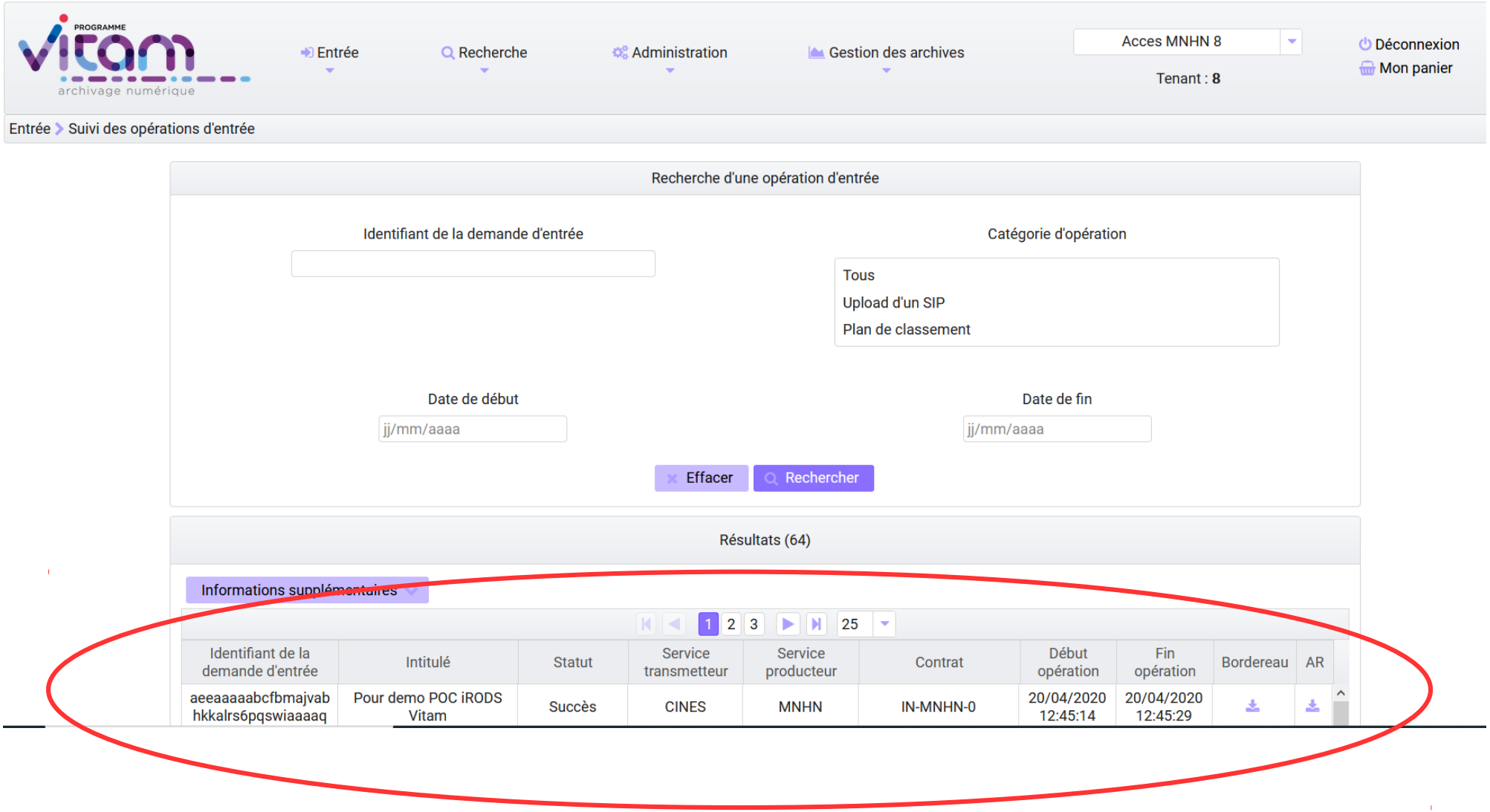
The vitam.re rule file 1/2

```
pep_api_data_obj_put_post(*INSTANCE_NAME, *COMM, *DATAOBJINP, *BUFFER, *PORTAL_OPR_OUT) {

if(*COMM.user_user_name != "rods")
{
        *obj_path = *DATAOBJINP.obj_path ;
        *user = *COMM.user_user_name ;
        writeLine("serverLog" , "*user stored object *obj_path");
        *cmd = "archive.sh" ;
        *par = *obj_path ;
        msiSetACL( "default" , "read" , "rods" , *obj_path );
        writeLine("serverLog" , "Sending *obj_path to SEDA 2.1 generator");
        msiExecCmd( *cmd , *par , "null" , "null" , "null" , *Result );
        msiGetStdoutInExecCmdOut( *Result , *Out );
        writeLine("serverLog" , "Output of *cmd is: *Out");
        #writeLine("serverLog" , "SEDA 2.1 generation is OK");
        msiDataObjUnlink( "objPath=*obj_path++++forceFlag=" , *Status );
        writeLine("serverLog" , "Removed *obj_path from the collection");

}
```

The vitam.re rule file 2/2

```
if(*COMM.user_user_name == "rods")
{
        *obj_path = *DATAOBJINP.obj_path ;
        *user = *COMM.user_user_name ;
        *cmd = "vitam.sh" ;
        *par = *obj_path ;
        writeLine("serverLog" , "*user stored object *obj_path");
        msiModAVUMetadata( "-d" , *obj_path , "add" , "ARCHIVED" , "False" , "Bool" );
        writeLine("serverLog" , "Set ARCHIVED metadata to False on *obj_path");
        msiExecCmd( *cmd , *par , "null" , "null" , "null" , *Result );
        msiGetStdoutInExecCmdOut( *Result , *Out );
        *x_request_id_line = elem ( split( *Out , "\r" ), 5) ;
        *x_request_id = elem ( split( *x_request_id_line , " " ), 1);
        msiModAVUMetadata( "-d" , *obj_path , "add" , "X-Request-Id" , *x_request_id , "String" );
        writeLine("serverLog" , "Set X-Request-Id metadata to *x_request_id on *obj_path");
        msiModAVUMetadata( "-d" , *obj_path , "add" , "SENT" , "True", "Bool" );
        writeLine("serverLog" , "Set SENT metadata to True on *obj_path");
        msiSleep( "10" , "0" );
        *cmd2 = "get.sh" ;
        *par2 = *obj_path ;
        msiExecCmd( *cmd2 , *par2 , "null" , "null", "null" , *Result2 );
        msiGetStdoutInExecCmdOut( *Result2 , *Out2 );
        writeLine("serverLog" , "Output of *cmd2 is: *Out2");
        writeLine("serverLog" , *Out2 like "\*<ReplyCode>OK</ReplyCode>\*");
        writeLine("serverLog" , "*obj_path successfully archived in Vitam");
        msiModAVUMetadata( "-d" , *obj_path , "set" , "ARCHIVED" , "True" , "Bool" );
        writeLine("serverLog" , "Set ARCHIVED metadata to True on *obj_path");
        msiDataObjUnlink( "objPath=*obj_path++++forceFlag=" , *Status );
        writeLine("serverLog" , "Removed *obj_path from the collection");
```

Our POC is a success:)



List of microservices used

- MsiSetACL
- MsiExecCmd
- MsiGetStdoutInExecCmdOut
- MsiDataObjUnlink
- MsiModAVUMetadata
- MsiSleep

Useful links

- Dynamic PEPs
- API Ingest External VITAM
- Resip GitHub issues here and there
- Discussions on the iRODS forum here and there
- Issue GitHub iRODS micro service plugin curl