Integration of DAX pipeline with XNAT

Dipankar Bachar, David Meunier, Aissam Rahmani

Brainhack Marseille

December 2 - December 4 2020

Plan :

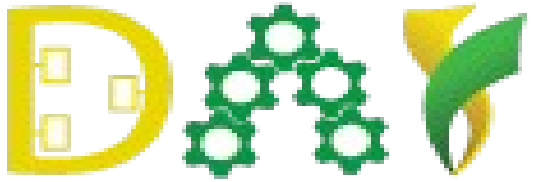
- What is DAX ? How it works ?
- Installing/Setup DAX

- DAX Pipelines
  - Spiders
  - Processors

- DAX command lines

What is DAX ?

Distributed Automation on XNAT (DAX), is a python package developed at Vanderbilt University, Nashville, USA

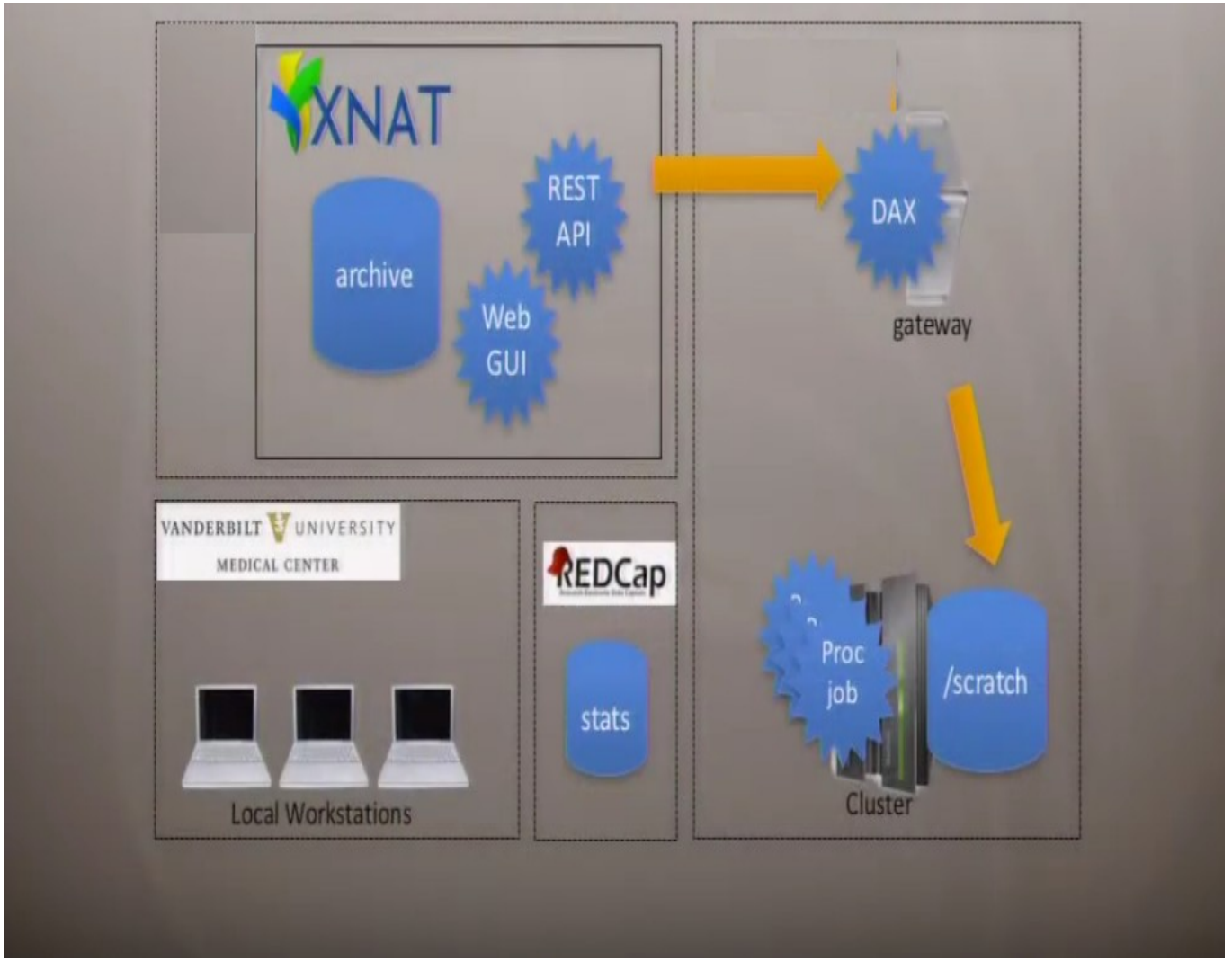


- ***DAX* allows you :**
  - **To extract project/data information from XNAT via scripts (Xnat_tools/Xnat...)**
  - **To create pipelines (spiders/processors) to run image processing jobs on your data**
  - **To run these jobs on the calculation clusters and to upload the results back to xnat**
  - **To interact with XNAT via python using commands in XnatUtils.**

How it works ?

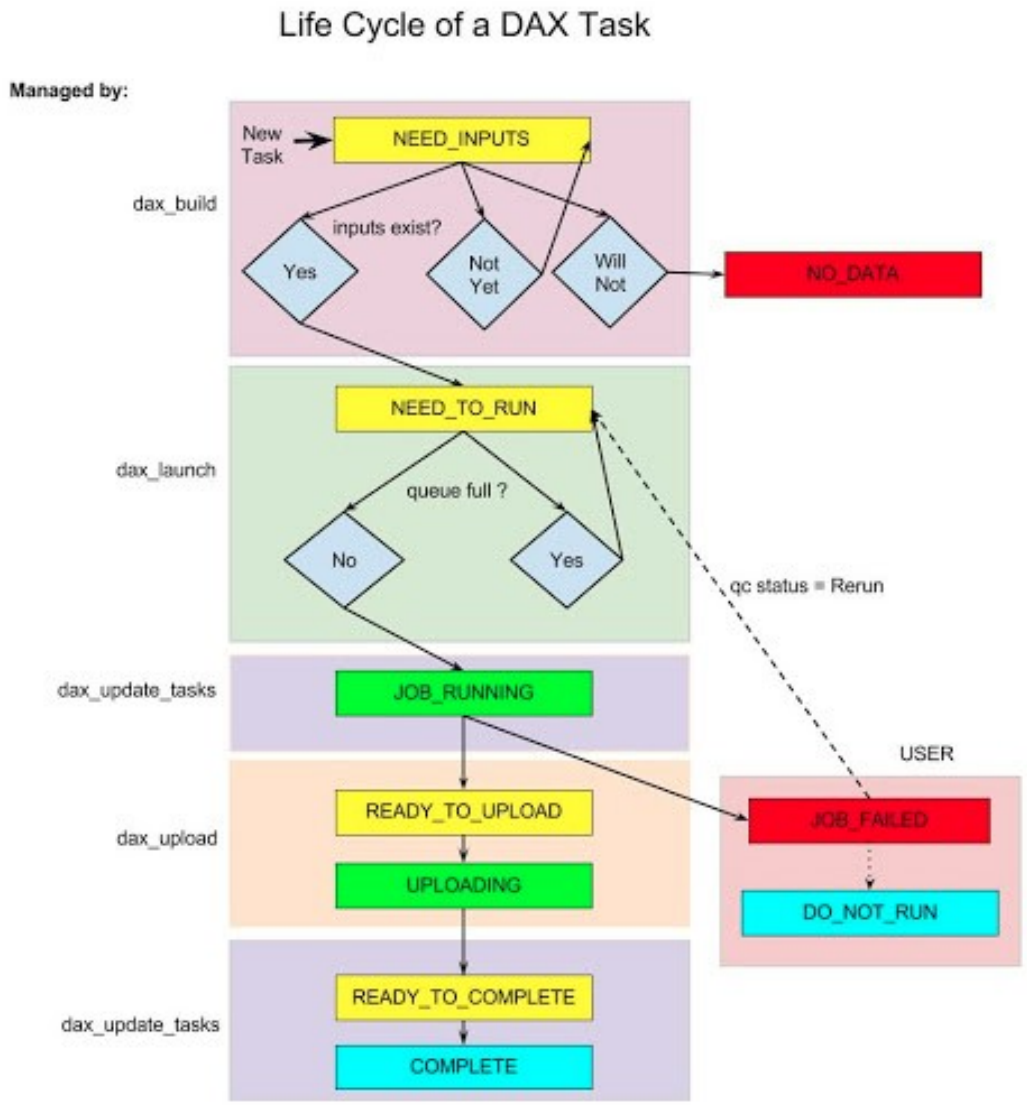
DAX communicates with XNAT via API REST, identify and download the session data into the computational cluster

Launch image processing tasks on Cluster (Spiders)



How it Works ?

After the calculation is finished in the cluster, upload the result back to XNAT server



Installing/Setup DAX

- **Dax can be installed in Python3**

```
$ pip install dax
```

It's available on github at the address
https://github.com/VUIIS/dax

- **To setup Dax with your Xnat server, run the following**

```
(dax) $ dax setup
 ########## DAX_SETUP ##########
Script to setup the ~/.dax_settings.ini files for your dax installation.

Warning: daxnetrc is empty. Setting XNAT login:
 Please enter your XNAT host: <xnat_host_url>
 Please enter your XNAT username: <username>
 Please enter your XNAT password: <password>
 --> Good login.

 Starting to config the dax_settings.ini file:
   -  Section: admin
      Do you want to set/modify the section [admin] in the settings file? [yes/no] no
   -  Section: cluster
      Do you want to set/modify the section [cluster] in the settings file? [yes/no] no
   -  Section: dax_manager
      Do you want to set/modify the section [dax_manager] in the settings file? [yes/no] no

0 error(s) -- dax_setup done. ########## END ##########
```

Installing/Setup DAX

- **Verify Installation**

```
$ XnatCheckLogin --host <xnat_host_url>
```

➢ Should provide ‘Good Login’

```
================================================================
Checking your settings for XNAT in xnatnetrc file:
Logins found for <http://10.164.4.26:8080/test>. Checking connection ...
  Connecting to host <http://10.164.4.26:8080/test> with user <admin>...
    --> Good login.
================================================================
```

DAX Pipelines

How do we define a Pipeline in DAX?

DAX pipelines are described by two files

- Spider

  The Spider is a python script. The purpose of the srcript is to define the pipeline that you want to run, and specify the data you need from XNAT and then prepare the processed data to upload to XNAT.

- Processors

  The Processor is a YAML text file defines the Environment, Inputs, Commands, and Outputs of the pipeline.

DAX Pipelines

❑ **How to write a spider?**

Naming conventions for Spider:

Any spider should be name: **Spider_ProcName_vX_Y_Z.py** where the ProcName will refer to the process you are running.

Each spider was divided into three functions:

- **pre_run**() : downloads the data from XNAT. Use XnatUtils methods to interact with XNAT.
- **run**() : runs the process (matlab or others).
- **finish**() : copies the data from your output folder to a Upload folder specify in your configuration file (.bashrc). To upload, you will need to use dax_upload.

You can follow this link to know how to write it from A to Z
https://github.com/VUIIS/dax/wiki/Writing-Spider

DAX Pipelines

❑ How to write a processor?

All processor YAML files should start with these two lines:

```
---
moreauto: true
```

The primary components of a processor YAML file are :

inputs outputs command attrs

The **inputs** section defines the files and parameters to be prepared for the pipeline.

The **outputs** section defines a list files or directories to be uploaded to XNAT upon completion of the pipeline

An Example

```
---
moreauto: true
inputs:
  default:
    container_path: MRIQA_v1.0.0.simg
  xnat:
    scans:
      - name: scan_t1
        types: MPRAGE
        resources:
          - resource: NIFTI
            ftype: FILE
            varname: t1_nifti
outputs:
  - path: stats.txt
    type: FILE
    resource: STATS
  - path: report.pdf
    type: FILE
    resource: PDF
  - path: DATA
    type: DIR
    resource: DATA
command: >-
  singularity
  run
  --bind $INDIR:/INPUTS
  --bind $OUTDIR:/OUTPUTS
  {container_path}
  --t1_nifti /INPUTS/{t1_nifti}
attrs:
  walltime: '36:00:00'
  memory: 8192
```

DAX Pipelines

❑ How to write a processor?

The **command** field defines a string template that is formatted using the values from **inputs**.
The command should represent the spider command you want to run. Each attribute specified in **{}** marks should correspond to an input specified in **inputs**.

The **attrs** section defines miscellaneous other attributes including cluster parameters. These values replace tags in the job template.

The **jobtemplate** is a text file that contains a template to create a batch job script.

An Example

```
---
moreauto: true
inputs:
  default:
    container_path: MRIQA_v1.0.0.simg
  xnat:
    scans:
      - name: scan_t1
        types: MPRAGE
        resources:
          - resource: NIFTI
            ftype: FILE
            varname: t1_nifti
outputs:
  - path: stats.txt
    type: FILE
    resource: STATS
  - path: report.pdf
    type: FILE
    resource: PDF
  - path: DATA
    type: DIR
    resource: DATA
command: >-
  singularity
  run
  --bind $INDIR:/INPUTS
  --bind $OUTDIR:/OUTPUTS
  {container_path}
  --t1_nifti /INPUTS/{t1_nifti}
attrs:
  walltime: '36:00:00'
  memory: 8192
```

DAX command lines

- **Xnatquery** is a tool to query objects on XNAT for each level. You can see which projects you have access to and see the hierarchy of data on your project.

  e.g $ Xnatquery -p projectID #To get all the subjects in this project

- **Xnatinfo** is the tool to get fast statistics information on a project (number of subjects/sessions/scans/assessors and the status of the assessors).

  e.g $Xnatinfo ProjectID

- **XnatCheck** is a quick way to check directly on your terminal if there is the resource you just created on all your project.

  e.g $Xnatdownload -p PID -d /tmp/downloadPID ……

- **Xnatdownload** will download all the resources that you asked for in a directory

- **Xnatupload** will create subject/experiment/scan/resources for a project on XNAT and upload the data into the project from a folder.

DAX command lines

- **Xnatreport** gives a report on one or more projects. It provides a report for the full project. No filters are applied in this tool. You can specify and output format to only print a subset of fields.

  e.g $Xnatreport -p PID # Report of a project

- **XnatCheckLogin** XnatCheckLogin allows the user to check that environment variables are
- set appropriately. It will let you know in a few seconds if your logins are good or not.

- **BIDSMapping** BIDSMapping tool allows the user to create, update or replace rules/mapping
- at the project level on XNAT. These rules are essential as they entail the link between scan
- type or series description on XNAT to the BIDS datatype, task type and repetition time. XnatToBids
- function uses these mapping at the project to transform XNAT data into the BIDS compliant
- data with BIDS filenames and folder structure.