
# Exploiting GPUs for medical imaging applications with VIP and Dirac

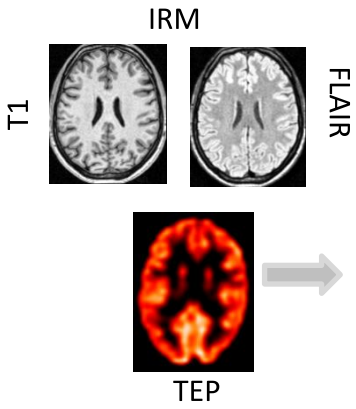
*Sorina Pop, Carole Lartizien, Pascal Wassong, Axel Bonnet, Thomas Grenier, Vanessa Hamar, Fabio Hernandez, Luisa Arrabito, Johan Bregeon, Pierre Gay, Andrei Tsaregorodtsev*

# Outline

- **Context**
  - The targeted medical applications
  - The Virtual Imaging Platform
  - Dirac
- Application deployment on GPUs
  - Docker container
  - Dirac SSH CE
  - VIP import
- Conclusions and perspectives

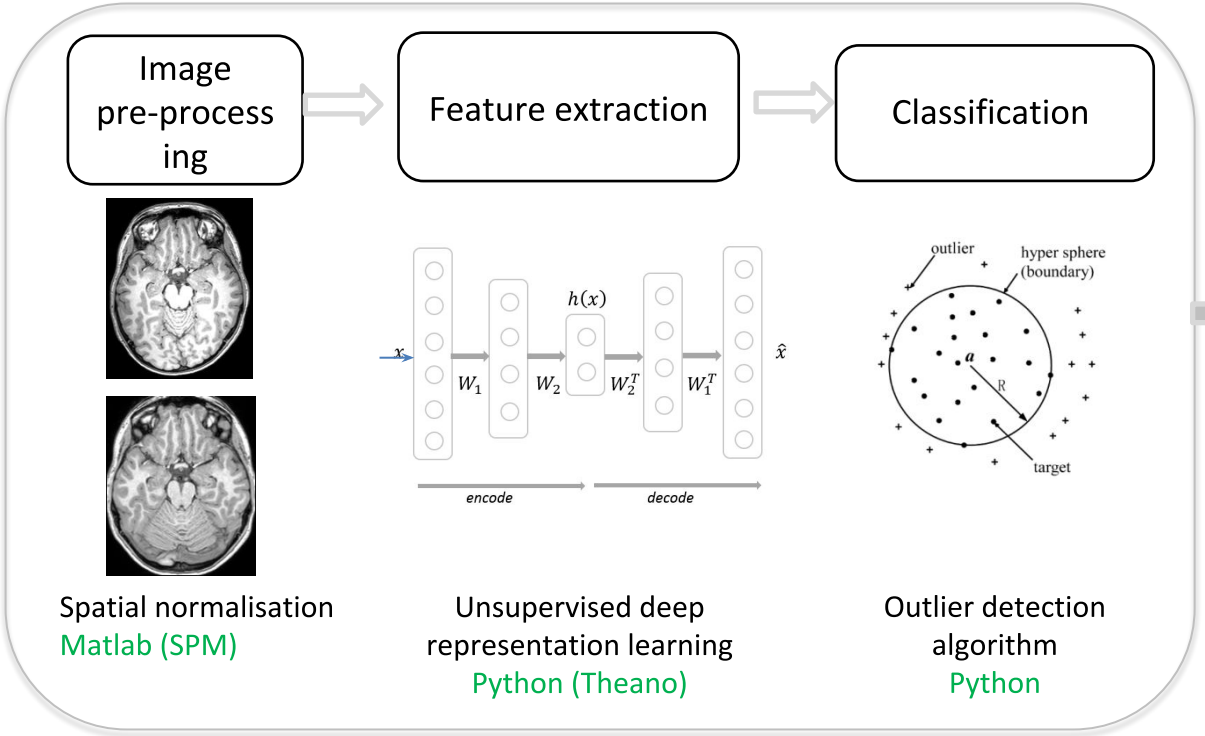
# A medical application to map brain pathologies based on multimodality neuroimaging and machine learning

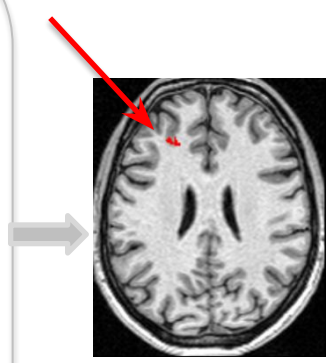
## Scientific context




**MR (T1/FLAIR) and TEP database**:

- 80 normal subjects
- 25 epilepsy patients

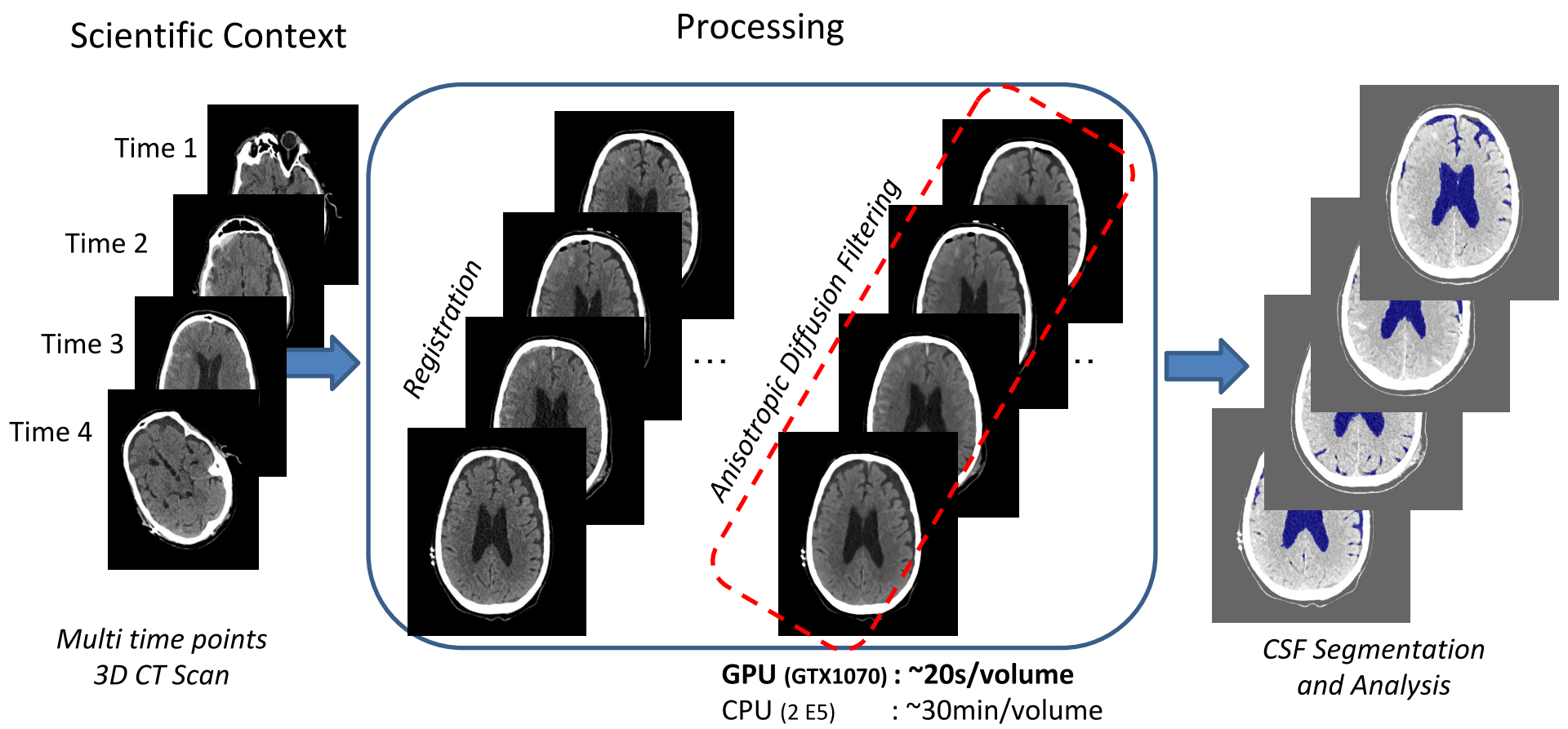


Crédits: Carole Lartizien, PhD work of Zara Alaverdyan (Oct 2015-Oct 2018)

[Alaverdyan MIDL 2018]

# Longitudinal study of Cerebrospinal fluid volume changes in CT images




Manet 2018 “CONVERSION OF POST-TRAUMATIC EXTERNAL HYDROCEPHALUS TO NORMAL PRESSURE HYDROCEPHALUS. AN ILLUSTRATIVE CASE”

# Virtual Imaging Platform (VIP)

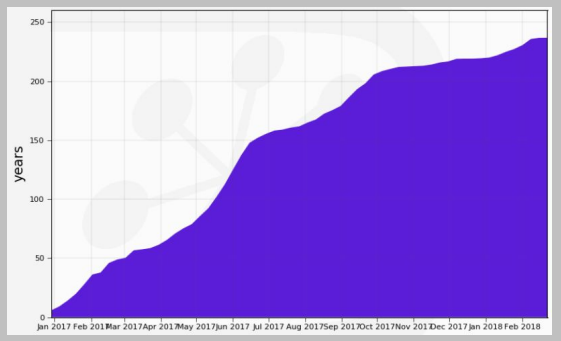
https://vip.creatis.insa-lyon.fr

## Web portal

Application as a service
File transfer to/from grid


## Infrastructure

Supported by EGI Infrastructure
Uses biomed VO (~65 sites in Europe and beyond)
230 cumulated CPU years utilized by VIP applications in 1 year

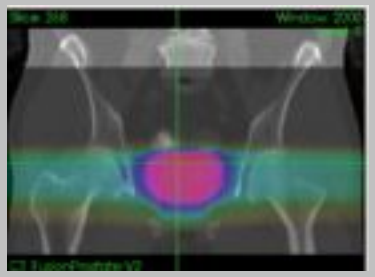



France-Grilles

DIRAC

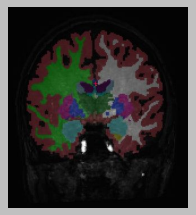
## Scientific applications

### Cancer therapy simulation



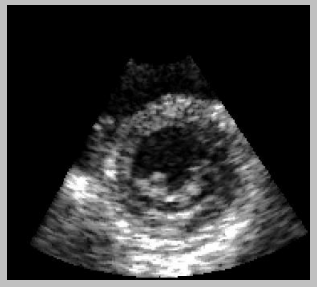
Prostate radiotherapy plan simulated with GATE(L. Grevillot and D. Sarrut)

### Neuro-image analysis



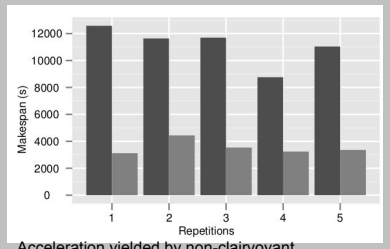
Brain tissue segmentation with Freesurfer

### Image simulation



Echocardiography simulated with FIELD-II (O. Bernard *et al*)

### Modeling and optimization of distributed computing systems



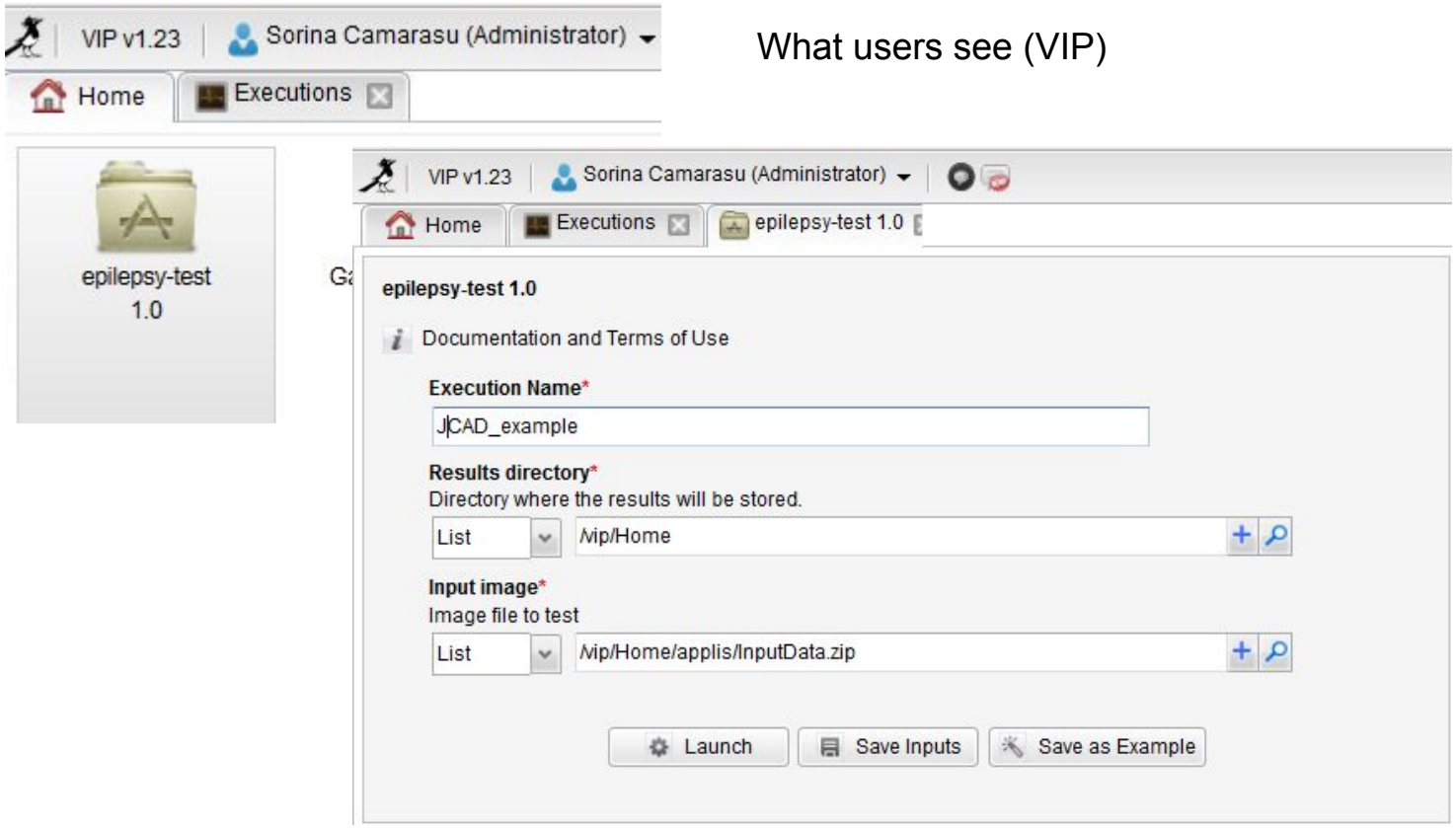

Acceleration yielded by non-clairvoyant task replication (R. Ferreira da Silva *et al*)

## Users

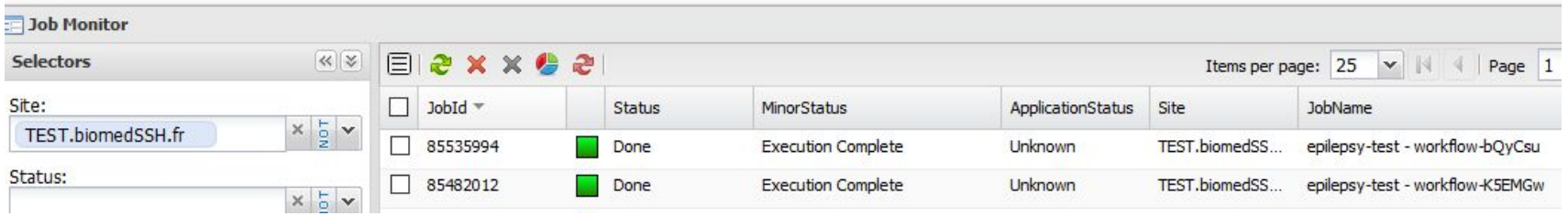
**1000+** registered users in October 2018
**44** publications since 2011

# Application execution on VIP

What users see (VIP)



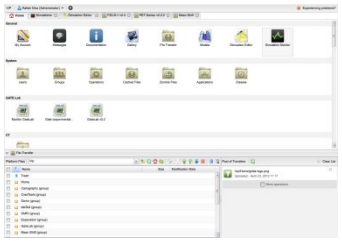

What users DON'T see (Dirac)



# VIP architecture

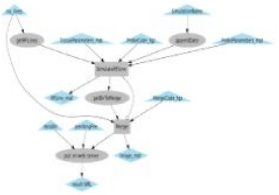
User

0. Login
1. Send input data



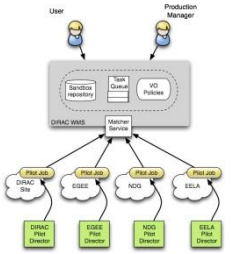
Web Portal

3. Launch workflow



Workflow Engine
(Moteur + GASW)

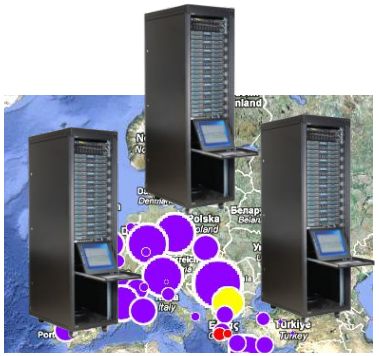
4. Generate and
submit task



Pilot Manager
(DIRAC)

6. Get task

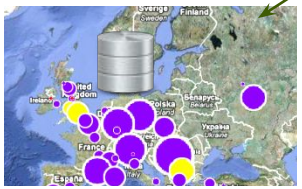
5. Submit
pilot jobs



Computing site

7. Get files
8. Execute
9. Upload results

2. Transfer
input files

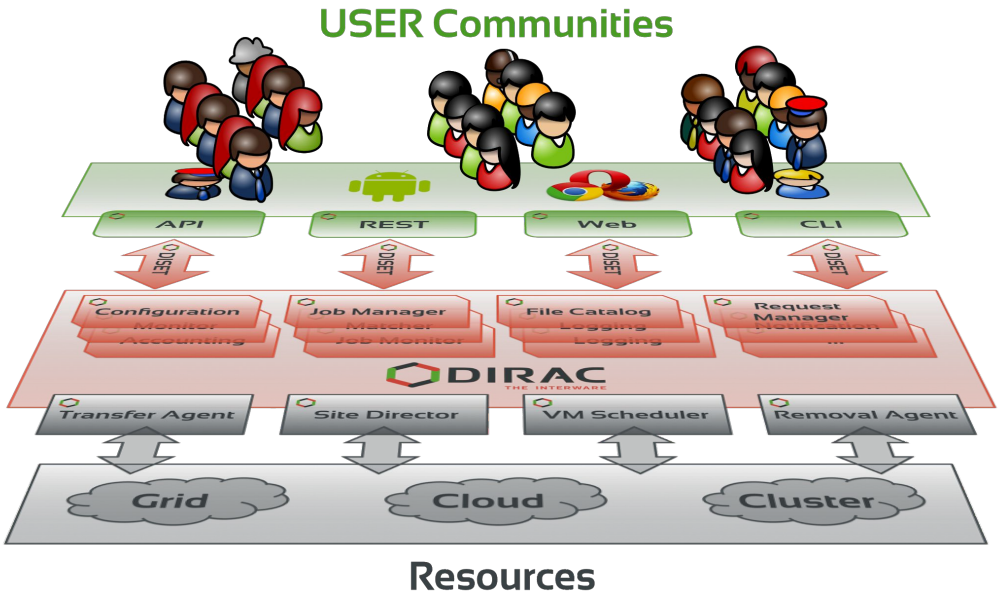


Storage
Element

DIRAC
THE INTERWARE

# DIRAC: the interware

- A software framework for distributed computing
- A complete solution to one (or more) user community
- Builds a layer between users and resources



# A few examples of DIRAC usages

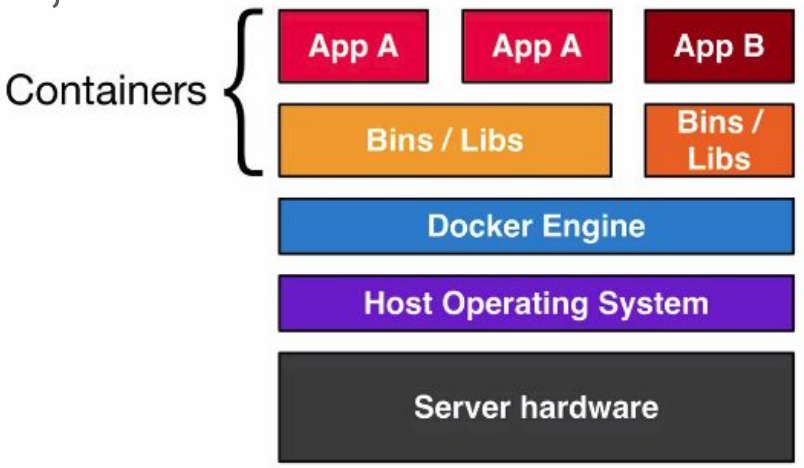
- Sending jobs to “the Grid”
  - e.g., the biomed VO for VIP
- Interfacing with different sites
  - Different computing elements and batch systems
  - e.g., individual clusters and GPUs
- More examples and details on the poster
  - “Dirac Interware for scientific applications”

# Outline

- Context
  - The targeted medical applications
  - The Virtual Imaging Platform
  - Dirac
- **Application deployment on GPUs**
  - Docker container
  - VIP import
  - Dirac SSH CE
- Conclusions and perspectives

# Docker containers

- A container = an entire runtime environment
  - An application + all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package
  - By containerizing the application platform and its dependencies, differences in OS distributions and underlying infrastructure are abstracted away
- Docker has become synonymous with container technology because its success, but
  - Container technology is not new
  - Other containers exist (Singularity)
- DockerHub
  - Image discovery and distribution
  - https://hub.docker.com






Credits: https://stackoverflow.com

# A Docker container for our application

- Prepare the Dockerfile
  - Use an existing nvidia image having cuda and cuDNN already installed (nvidia/cuda:7.5-cudnn5-devel-centos7)
  - Install and configure anaconda, theano and keras
  - Bring in code source with git clone (or “ADD” local files)
- Build the image
  - docker build -t feature-extraction .
- Use nvidia-docker
  - docker runtime enabling access to the GPU
- Start the container using the nvidia runtime
  - docker run --runtime=nvidia -it feature-extraction

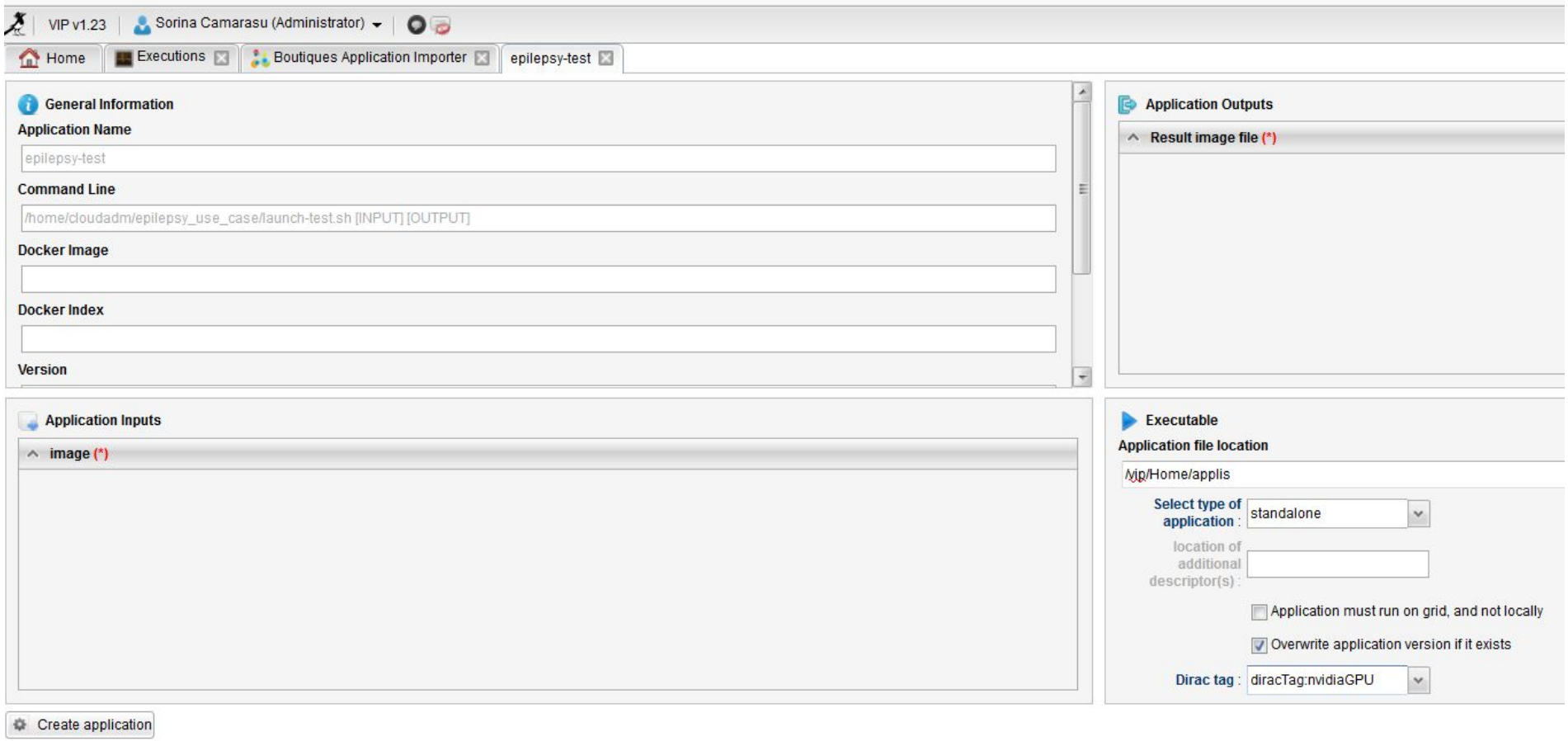
# Boutiques

- Describe, publish, integrate and execute command-line applications **across platforms**
  - facilitate application porting
  - import and exchange of applications
- Use of Linux containers to facilitate application installation and sharing
- https://github.com/boutiques

```
{
    "name": "epilepsy-test",
    "tool-version": "1.0",
    "description": "Run Epilepsy test, using launcher script",
    "command-line": "launch-train.sh [INPUT] [OUTPUT]",
    "schema-version": "0.5",
    "container-image": {
        "type": "docker",
        "image": "feature-extraction"
    },
    "inputs": [
        {
            "id": "image",
            "name": "Input image",
            "type": "String",
            "description": "Image file to test",
            "value-key": "[INPUT]",
            "list": false,
            "optional": false,
            "default-value": ""
        }
    ],
    "output-files": [
        {
            "id": "result",
            "name": "Result image file",
            "description": "Result image file",
            "value-key": "[OUTPUT]",
            "path-template": "[INPUT].tar.gz"
        }
    ]
}
```

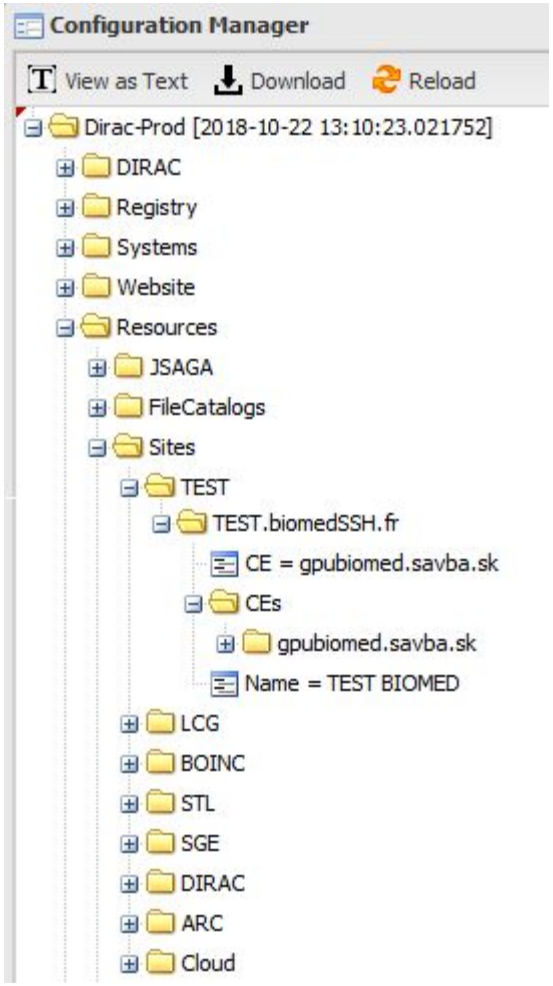
Boutiques JSON descriptor to define the command-line, inputs and outputs of the application

# Automatic import into VIP with Boutiques



# Dirac Resource Configuration

- Add computing resource
  - Currently SSH Computing Element
- Configure resource
  - CEType=SSH
  - Name, IP address, public key...
- Use Dirac Tag
  - Tag: this resource can receive tasks which need that Tag
  - RequiredTag: ONLY jobs requesting that tag are allowed on this resource
- Submit a Dirac job
  - Tags = “NvidiaGPU”;



# Outline

- Context
  - The targeted medical applications
  - The Virtual Imaging Platform
  - Dirac
- Application deployment on GPUs
  - Docker container
  - VIP import
  - Dirac SSH CE
- **Conclusions and perspectives**

# Conclusions

- How to facilitate GPU usage for the processing of medical data and efficient machine learning approaches ?
  - VIP users can access applications as a service
  - DIRAC allows for transparent job execution on distributed infrastructures such as grid and clouds
  - Docker containers to automate the deployment
- Challenges and future work
  - Finish application integration in production
  - Handling of large data volumes
  - Integration of multiple GPUs from various disparate sources

Thank you for your attention!
Question?