This project has received funding from the European Union’s Horizon 2020 Research and Innovation programme under Grant Agreement No 730872.

CALIPSOplus JRA2: Prototype of Data Analysis as a Service Platform

Aidan Campbell (ESRF)

Daniel Webster (PSI)

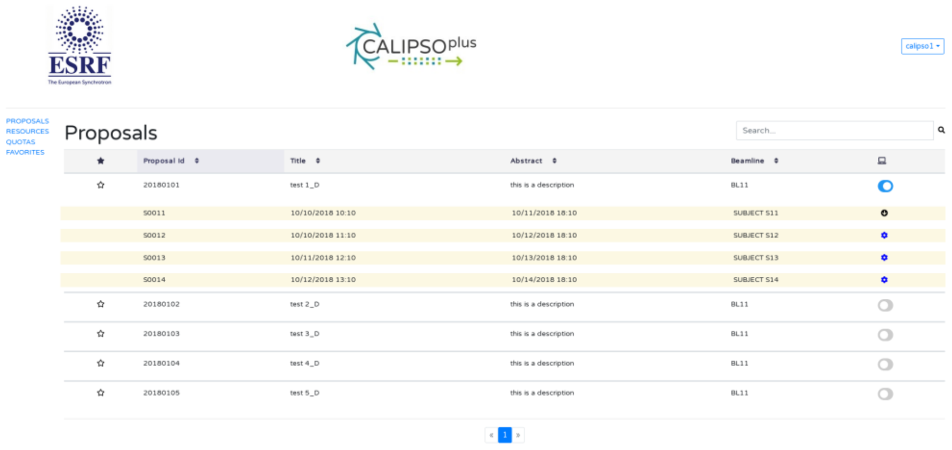
Motivation

- Light sources are generators of big volumes of complex scientific data and their users need assistance in analysing the scientific data.

- Our aim is to provide a remote Data Analysis As a Service (DAAS) portal and platform for users to:
  - Access their experimental data
  - Use pre-packaged Data Analysis software available at each institute
  - Access onsite computer resources to assist with data reduction and processing

CALIPSOPlus DAAS Portal

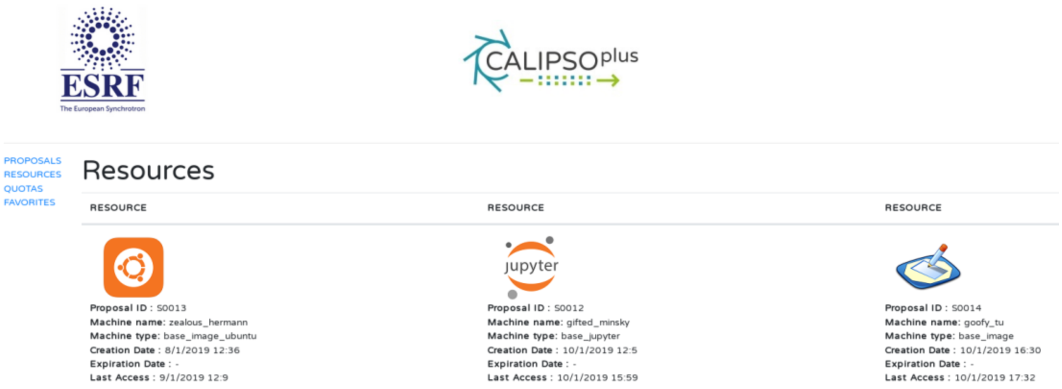
- Joint Project: ALBA/ESRF/PSI
- Written: Angular 7 / Django
- Common home page which connects user to institute portal
- Custom installation tested at: ESRF, ALBA, ELETTRA, PSI, DESY, SOLEIL, DLS
- Services:
  - Jupyter Notebooks
  - Containers
  - Virtual Machines



Proposals

★	Proposal Id	Title	Abstract	Beamline	
☆	20180101	test 1_D	this is a description	BL11	●
	S0011	10/10/2018 10:10	10/11/2018 18:10	SUBJECT S11	●
	S0012	10/10/2018 11:10	10/12/2018 18:10	SUBJECT S12	⚙
	S0013	10/11/2018 12:10	10/13/2018 18:10	SUBJECT S13	⚙
	S0014	10/12/2018 13:10	10/14/2018 18:10	SUBJECT S14	⚙
☆	20180102	test 2_D	this is a description	BL11	○
☆	20180103	test 3_D	this is a description	BL11	○
☆	20180104	test 4_D	this is a description	BL11	○
☆	20180105	test 5_D	this is a description	BL11	○

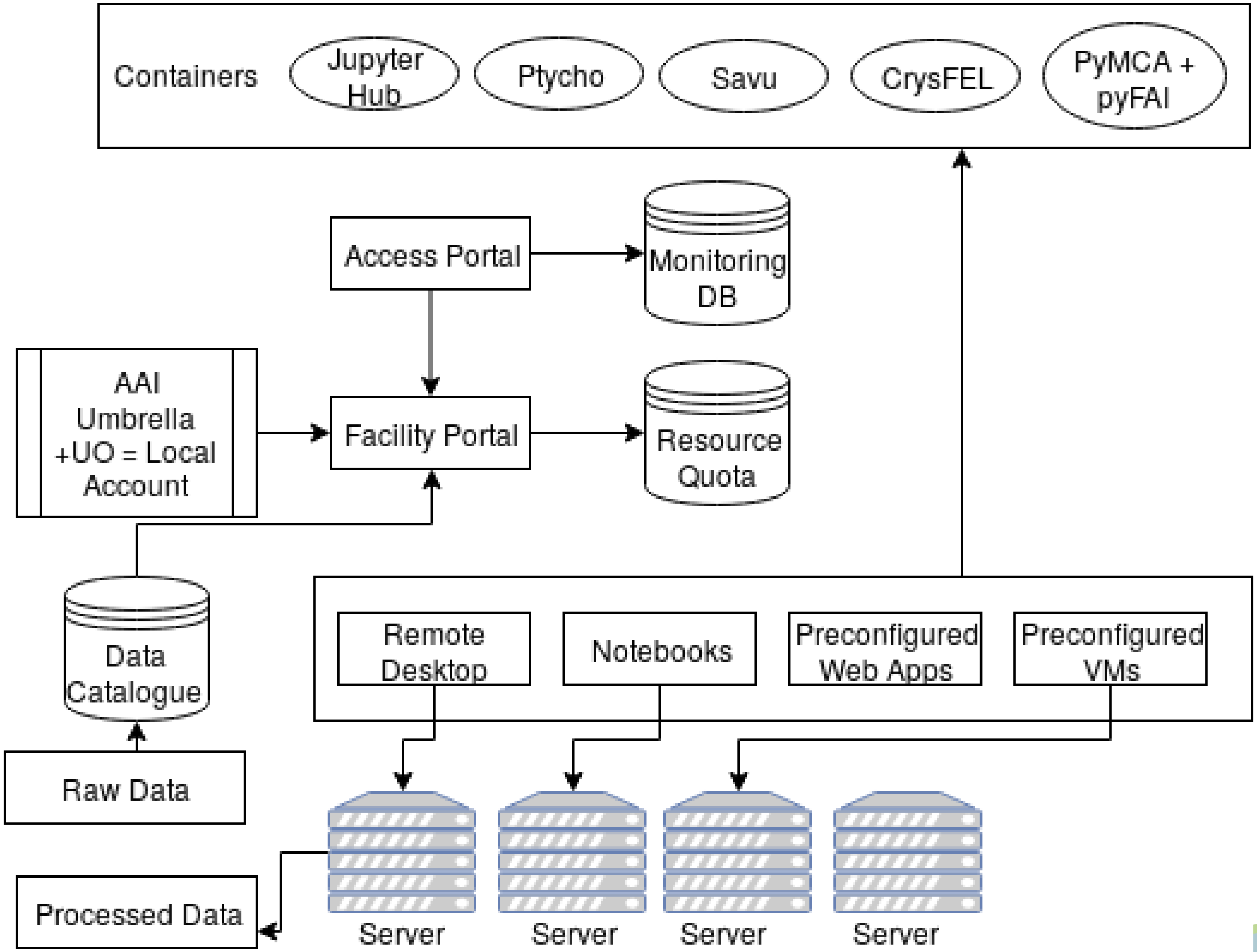
« 1 »



Resources

RESOURCE	RESOURCE	RESOURCE
Proposal ID : S0013 **Machine name:** zealous_hermann **Machine type:** base_image_ubuntu **Creation Date :** 8/1/2019 12:36 **Expiration Date :** - **Last Access :** 9/1/2019 12:9	**Proposal ID :** S0012 **Machine name:** gifted_minsky **Machine type:** base_jupyter **Creation Date :** 10/1/2019 12:5 **Expiration Date :** - **Last Access :** 10/1/2019 15:59	**Proposal ID :** S0014 **Machine name:** goofy_tu **Machine type:** base_image **Creation Date :** 10/1/2019 16:30 **Expiration Date :** - **Last Access :** 10/1/2019 17:32

CALIPSOPlus DAAS Portal Architecture*

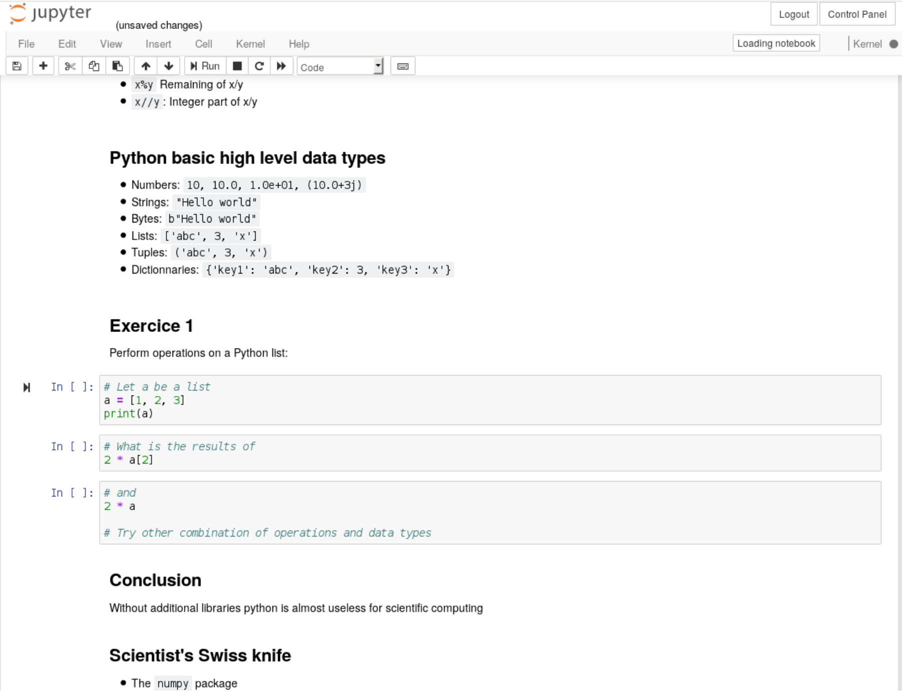


See paper https://icalepcs2019.vrws.de/papers/wepha057.pdf

Jupyter Notebooks (ESRF)

- Currently
  - 40+ users
  - Limited hardware resources

- Near Future
  - 100+ users
  - SLURM
    - Scalability
    - GPU Access
  - More GPUs available



- `x%y` Remaining of x/y
- `x//y` : Integer part of x/y

Python basic high level data types

- Numbers: `10, 10.0, 1.0e+01, (10.0+3j)`
- Strings: `"Hello world"`
- Bytes: `b"Hello world"`
- Lists: `['abc', 3, 'x']`
- Tuples: `('abc', 3, 'x')`
- Dictionnaries: `{'key1': 'abc', 'key2': 3, 'key3': 'x'}`

Exercice 1

Perform operations on a Python list:

```
In [ ]: # Let a be a list
        a = [1, 2, 3]
        print(a)

In [ ]: # What is the results of
        2 * a[2]

In [ ]: # and
        2 * a

        # Try other combination of operations and data types
```

Conclusion

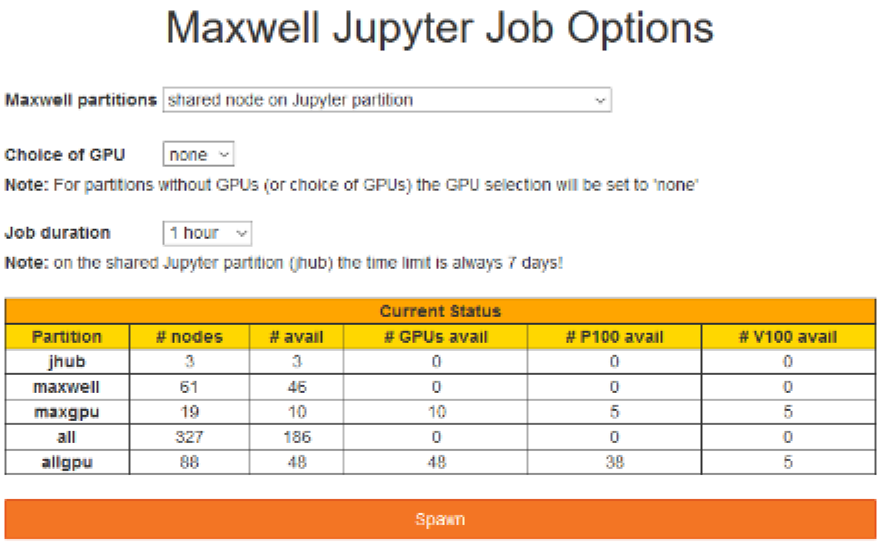
Without additional libraries python is almost useless for scientific computing

Scientist's Swiss knife

- The `numpy` package

Jupyter Notebooks (DESY)

- 2018 - 2019
  - 180+ users
  - Typically 60-120
  - concurrent sessions
  - SLURM
    - Scalability
    - GPU Access
  - 3 AMD dedicated nodes
    - Other partitions can
    - be used too



Maxwell Jupyter Job Options

Maxwell partitions shared node on Jupyter partition

Choice of GPU none

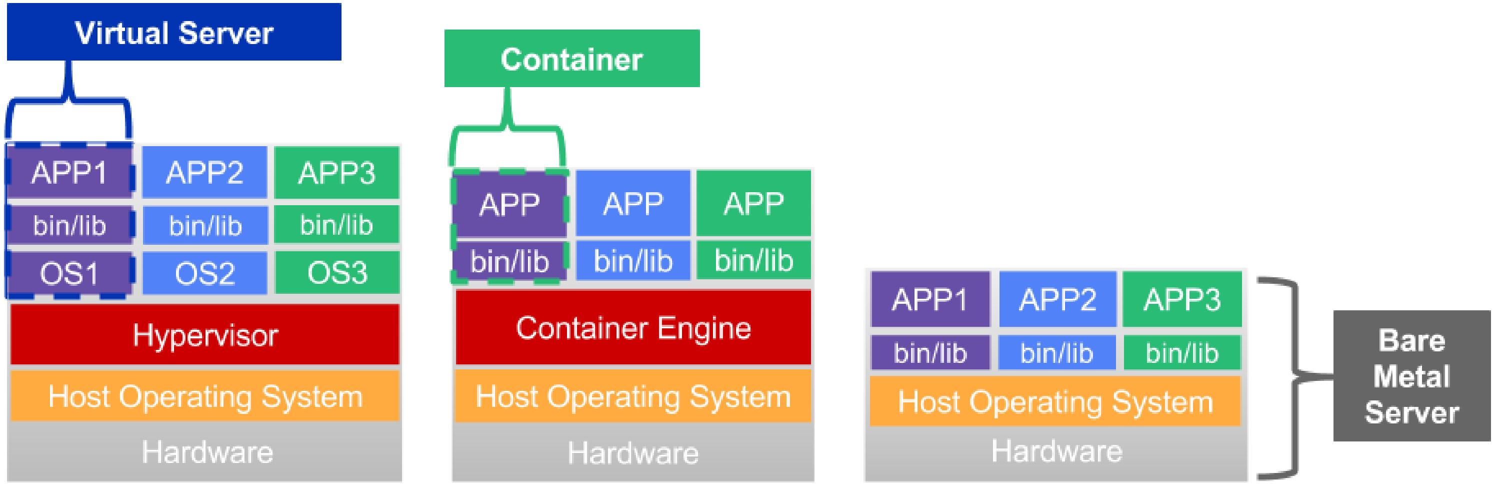
Note: For partitions without GPUs (or choice of GPUs) the GPU selection will be set to 'none'

Job duration 1 hour

Note: on the shared Jupyter partition (jhub) the time limit is always 7 days!

Current Status					
Partition	**# nodes**	**# avail**	**# GPUs avail**	**# P100 avail**	**# V100 avail**
jhub	3	3	0	0	0
maxwell	61	46	0	0	0
maxgpu	19	10	10	5	5
all	327	186	0	0	0
allgpu	88	48	48	38	5

Compute Options: VM, Container, Bare Metal



Virtual Server

APP1 | APP2 | APP3

bin/lib | bin/lib | bin/lib

OS1 | OS2 | OS3

Hypervisor

Host Operating System

Hardware

Container

APP | APP | APP

bin/lib | bin/lib | bin/lib

Container Engine

Host Operating System

Hardware

APP1 | APP2 | APP3

bin/lib | bin/lib | bin/lib

Host Operating System

Hardware

Bare Metal Server

Pre-installed software

Pre-installed software

Sudo privileges

© 2016 Kumulus Technologies

Containers

- Computer that scientists and users can access with pre-installed software entirely in the web
- Scientists can:
  - Request a specific Linux OS (Ubuntu, CentOS, Debian etc)
  - Install their own software
  - Access their data (NFS)
  - Do analysis on site hardware from home/university
- Developers can:
  - Create/update container images with Github
  - Containers will update automatically on portal

Virtual Machines

- Computer that scientists and users can access with pre-installed software entirely in the web
- Scientists can:
  - Request a specific OS (including Windows)
  - Can’t install their own software (security risks)
  - Access their data
  - Do analysis on site hardware from home/university
- Use Cases tested:
  - PyMca (ESRF), pyFAI (ESRF), PtychoShelves (PSI), Savu (DLS), CrysFEL (DESY)

Future Developments

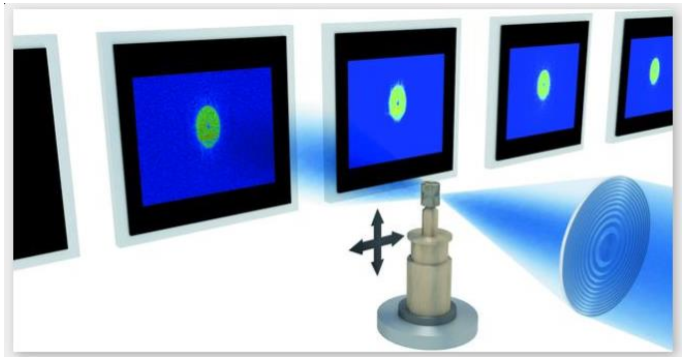
- Integrate experiment data within the portal
  - Custom plug-ins required
- Building tailored Notebooks for beamlines
- Create containers/virtual machines without needing an experiment
- Write report on needs of Calipsoplus community wrt European Open Science Cloud
- Integrate results with PaNOSC and ExPaNDS DAAS portal

PSI Deployment

- This project represents an ideal opportunity to prove emergent technologies
  - Containerisation and orchestration thereof
  - Microservice Architectures
  - “DevOps” methodologies
- PSI chose to deploy the portal on Red Hat OpenShift
  - Enterprise-grade Kubernetes distribution
- CALIPSOplus portal had to be deployed as a Microservice Architecture
  - A wealth of experience gained for PSI in this methodology

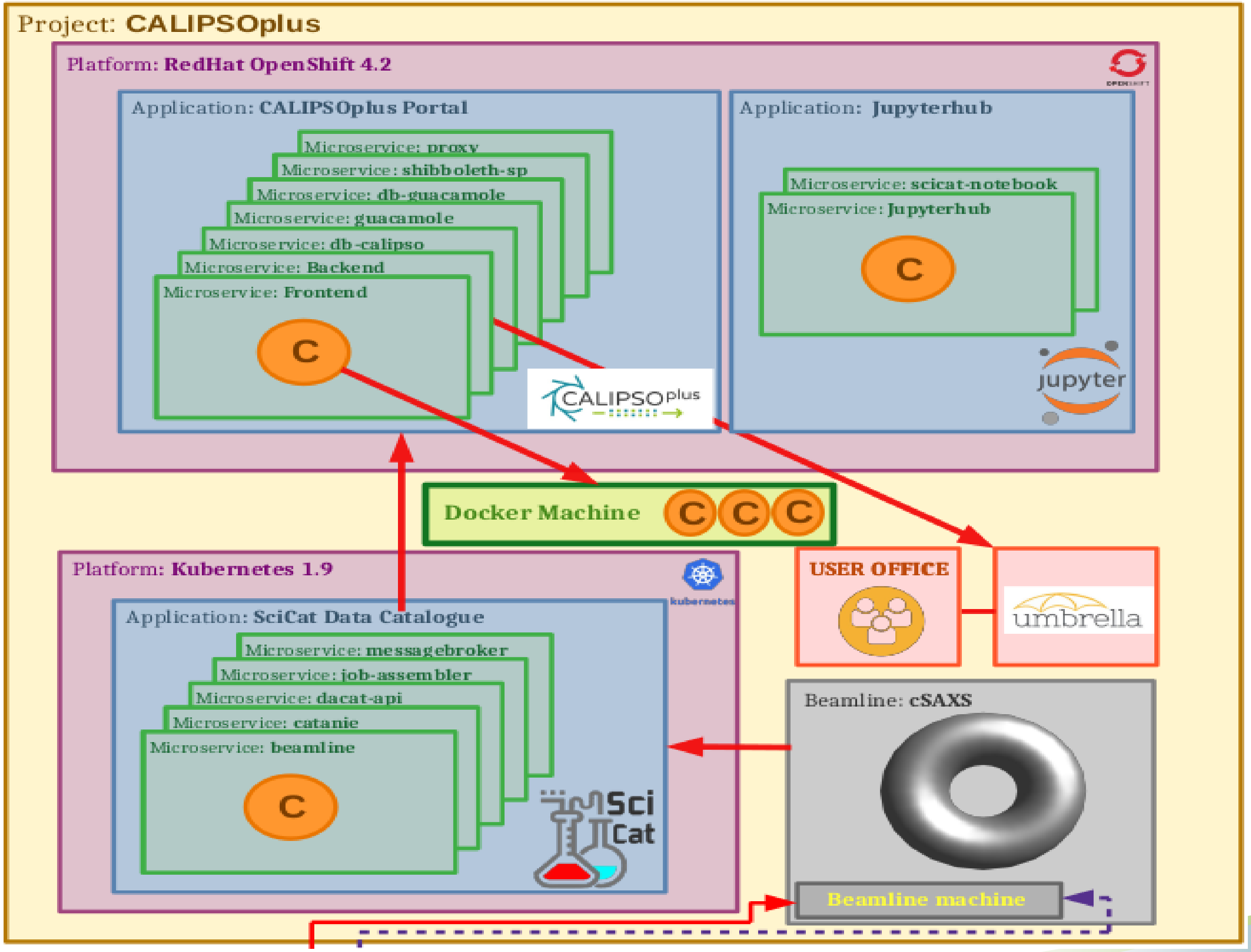
PSI Use Case

- We engaged our cSAXS Beamline for our CALIPSOplus use case
  - ***cSAXS*** – Coherent Small-Angle X-ray Scattering - uses Ptychography (among other techniques) for image reconstruction
  - ***Ptychography***: computationally generate images by processing coherent interference patterns
  - The application we are using for this is ***PtychoShelves***
    - ***PtychoShelves***, a versatile high-level framework for high-performance analysis of ptychographic data



  - Paper can be found here: **https://scripts.iucr.org/cgi-bin/paper?zy5001**
    - *(Will paste into chat window)*

PSI Architecture



Project: **CALIPSOplus**

Platform: **RedHat OpenShift 4.2**

Application: **CALIPSOplus Portal**

Microservice: **proxy**

Microservice: **shibboleth-sp**

Microservice: **db-guacamole**

Microservice: **guacamole**

Microservice: **db-calipso**

Microservice: **Backend**

Microservice: **Frontend**

C

Application: **Jupyterhub**

Microservice: **scicat-notebook**

Microservice: **Jupyterhub**

C

jupyter

Docker Machine C C C

Platform: **Kubernetes 1.9**

kubernetes

Application: **SciCat Data Catalogue**

Microservice: **messagebroker**

Microservice: **job-assembler**

Microservice: **dacat-api**

Microservice: **catanie**

Microservice: **beamline**

C

SciCat

USER OFFICE

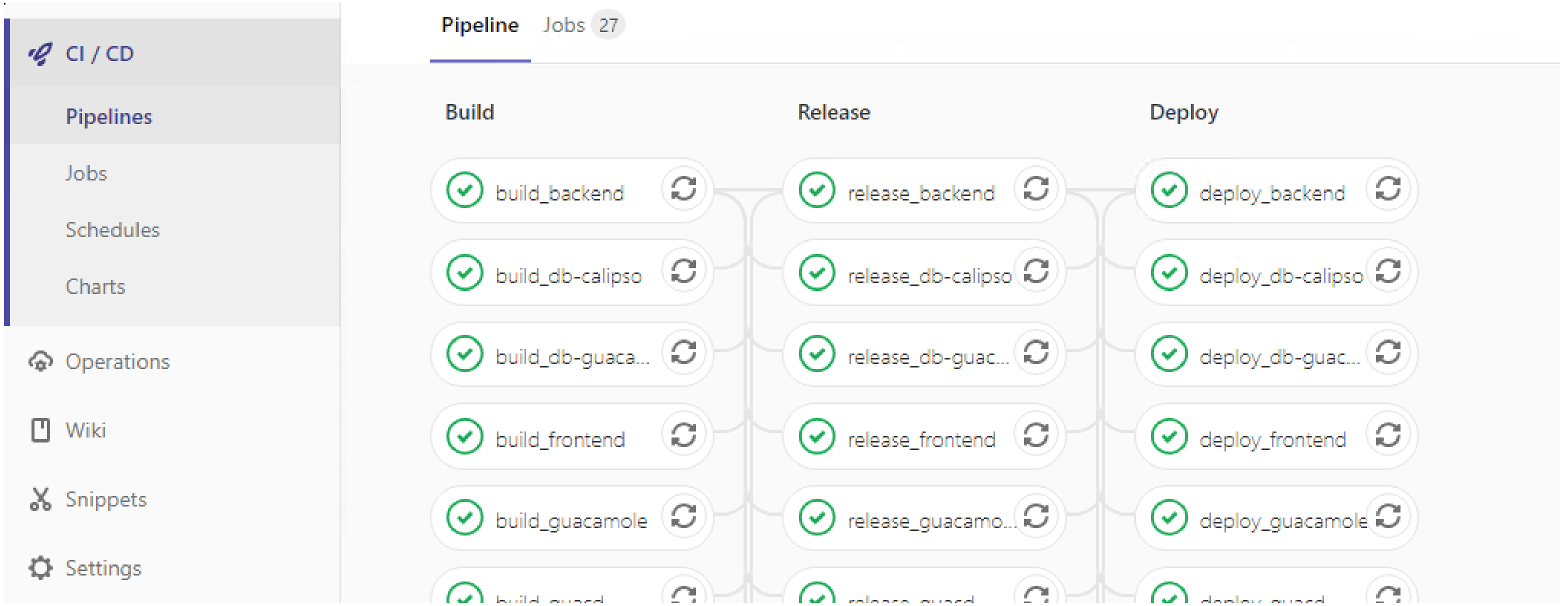
umbrella

Beamline: **cSAXS**

Beamline machine

Deploying the Application

- We deploy CALIPSOplus as a Microservice Architecture via "Continuous Integration/Continuous Deployment" (CI/CD) from GitLab:

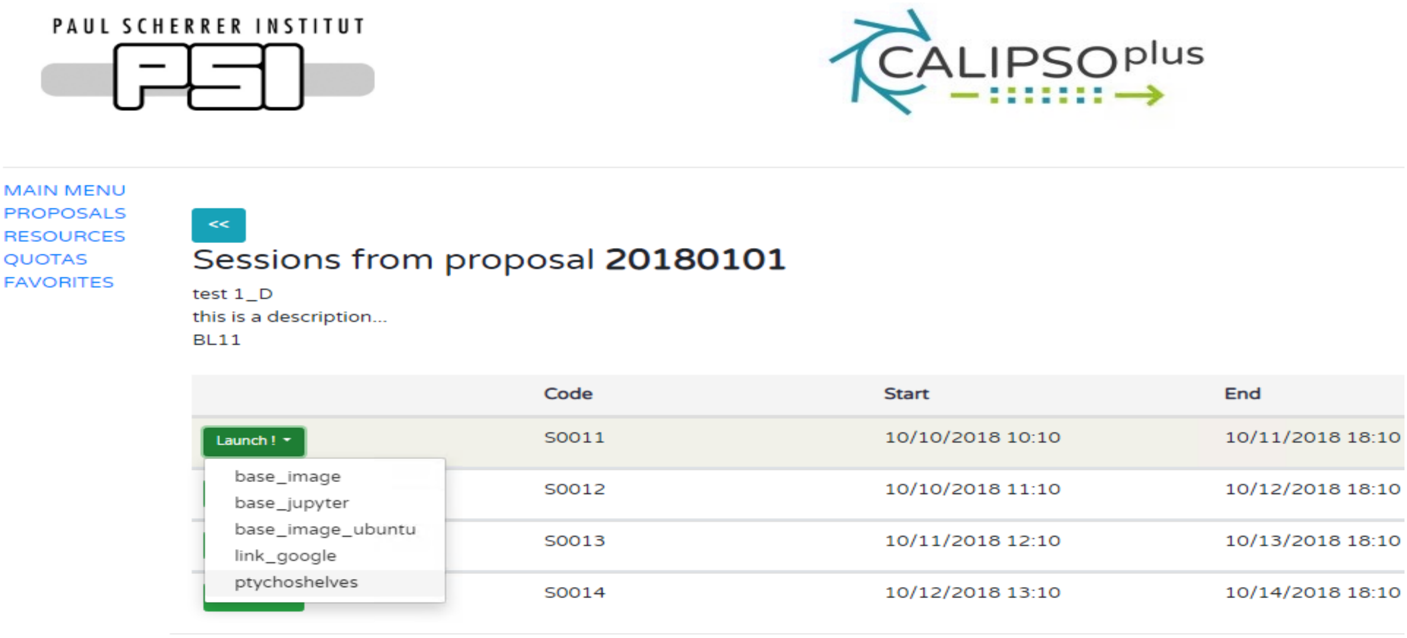


Build	Release	Deploy
build_backend	release_backend	deploy_backend
build_db-calipso	release_db-calipso	deploy_db-calipso
build_db-guaca...	release_db-guac...	deploy_db-guac...
build_frontend	release_frontend	deploy_frontend
build_guacamole	release_guacamo...	deploy_guacamole
build_guard	release_guard	deploy_guard

- A new build is triggered, released, and deployed automatically to our OpenShift instance, upon committing new code
- This is "**DevOps**" in action

Demonstration - Screenshots

- We select the proposal we are interested in, launch our chosen container on top of this data, and connect to it via RDP or VNC



Sessions from proposal **20180101**

test 1_D
this is a description...
BL11

	Code	Start	End
Launch !	S0011	10/10/2018 10:10	10/11/2018 18:10
	S0012	10/10/2018 11:10	10/12/2018 18:10
	S0013	10/11/2018 12:10	10/13/2018 18:10
	S0014	10/12/2018 13:10	10/14/2018 18:10

- A Linux "system" will now be at our disposal, and our scientific application is already mapped-in and ready to run

Demonstration - Screenshots

- We can then run our MATLAB package, and get our results:

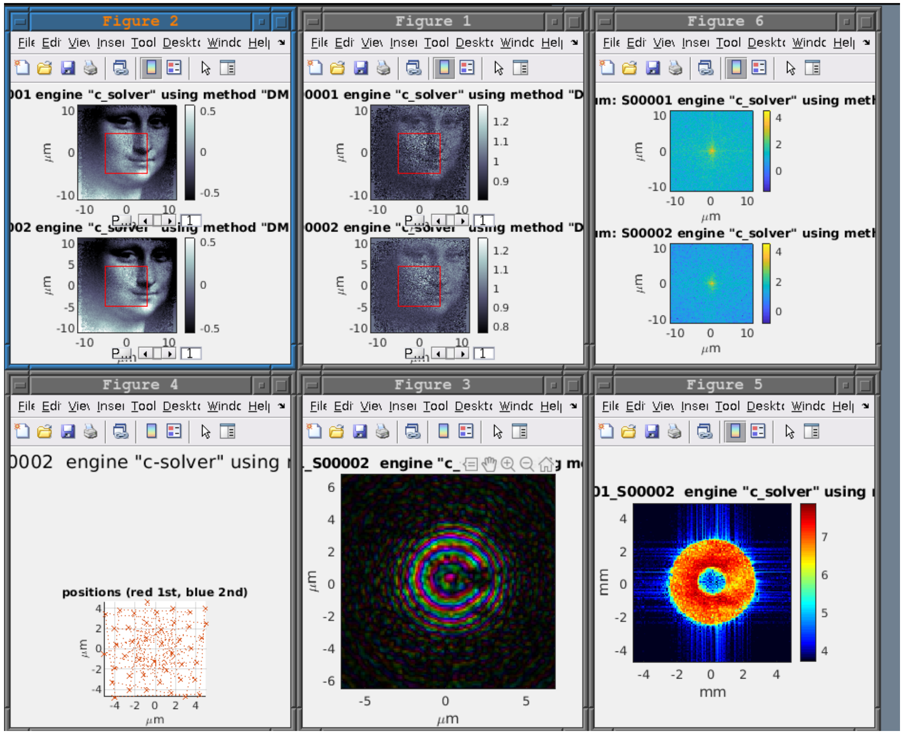
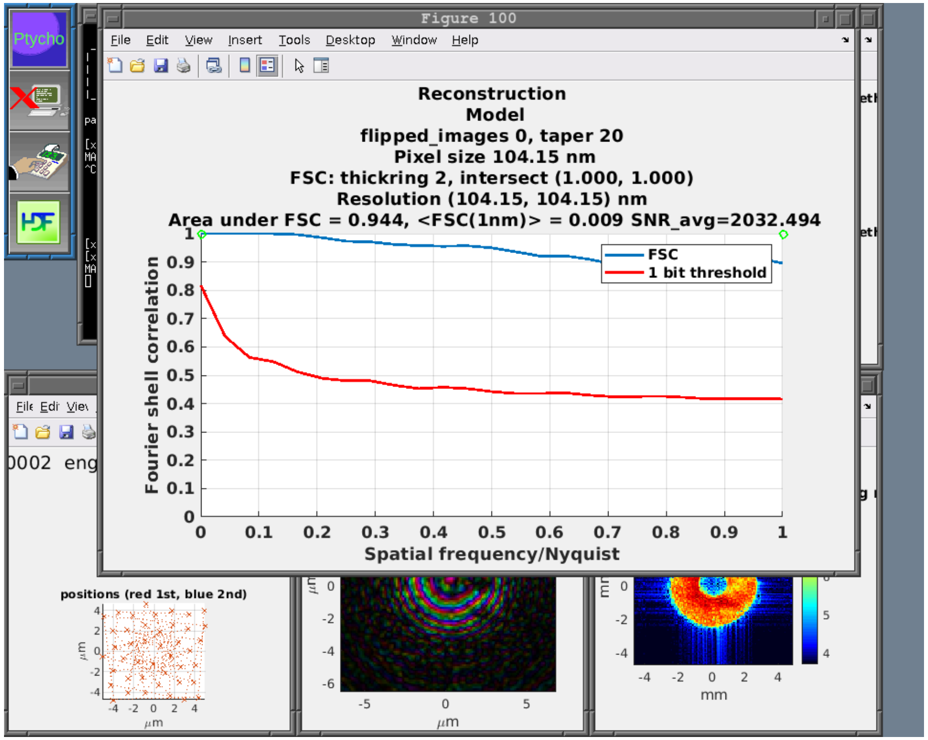


Figure 2
Figure 1
Figure 6
Figure 4
Figure 3
Figure 5
001 engine "c_solver" using method "DM
002 engine "c_solver" using method "DM
0001 engine "c_solver" using method "D
0002 engine "c_solver" using method "D
m: S00001 engine "c_solver" using meth
m: S00002 engine "c_solver" using meth
0002 engine "c-solver" using
S00002 engine "c
01_S00002 engine "c_solver" using
positions (red 1st, blue 2nd)
μm
mm



Ptycho
Figure 100
File Edit View Insert Tools Desktop Window Help
Reconstruction
Model
flipped_images 0, taper 20
Pixel size 104.15 nm
FSC: thickring 2, intersect (1.000, 1.000)
Resolution (104.15, 104.15) nm
Area under FSC = 0.944, <FSC(1nm)> = 0.009 SNR_avg=2032.494
FSC
1 bit threshold
Fourier shell correlation
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Spatial frequency/Nyquist
positions (red 1st, blue 2nd)
μm
mm

Live Demo – please standby..

- We will now show a live demo of the portal in action

Conclusions

- CALIPSOplus JRA has been very useful in bringing together sites to collaborate and share a prototype portal for providing Data Analysis as a Service
- Feedback from users was positive and demonstrated the need for such services
- Jupyter service is being generalised at most sites
- Other services (containers+VM) are under test
- Main difficulty encountered in providing DAAS services in production is the lack of data analysis policy at all the sites (a survey has been prepared to get feedback from sites)
- Future developments will be in EOSC with PaNOSC+ExPaNDS

Acknowledgments

- **ESRF**:
- A.Campbell, A.Götz, A.Roux, J.Kieffer, T.Vincent, A.Solé, A.de Maria, M.Retegan, V.Favre-Nicolas, B.Roussel
- **ALBA**:
- D.Salvat, A.Camps, D.Sanchez
- **Elettra**:
- G.Kourousias, I.Andrian, D.Palmisano
- **Paul Scherrer Institute**:
- M.van Daalen, S.Egli, D.Webster, A.Ashton
- **DESY:**
- J. Reppin, F.Schluenzen
- **Diamond Light Source**
- T.Schoonjans
- **SOLEIL:**
- M.Ounsy, G.Viguier
- **Helmholtz Zentrum Dresden Rossendorf:**
- B.Schramm, M.Grobosch

Thank You

...questions?

Backup slides

- Extra slides which can be used in case there are questions on certain details

Jupyter Notebook Prototypes

- SLURM (DESY, ESRF)
  - Interactive notebook server created using a scheduler
  - GPU support
  - 12 hour session
  - Cannot install new software (except by pip install --user)
- SudoSpawner
  - Single machine with GPU support
  - Very limited hardware resources
- Kubernetes (ALBA, PSI)
  - Load balanced notebooks (more computing resources)
  - Unlimited session time
  - Can install new software (apt-get, pip, etc) temporarily
  - Notebook is culled after X hours of inactivity
  - Multiple custom notebook images

Virtual Machines

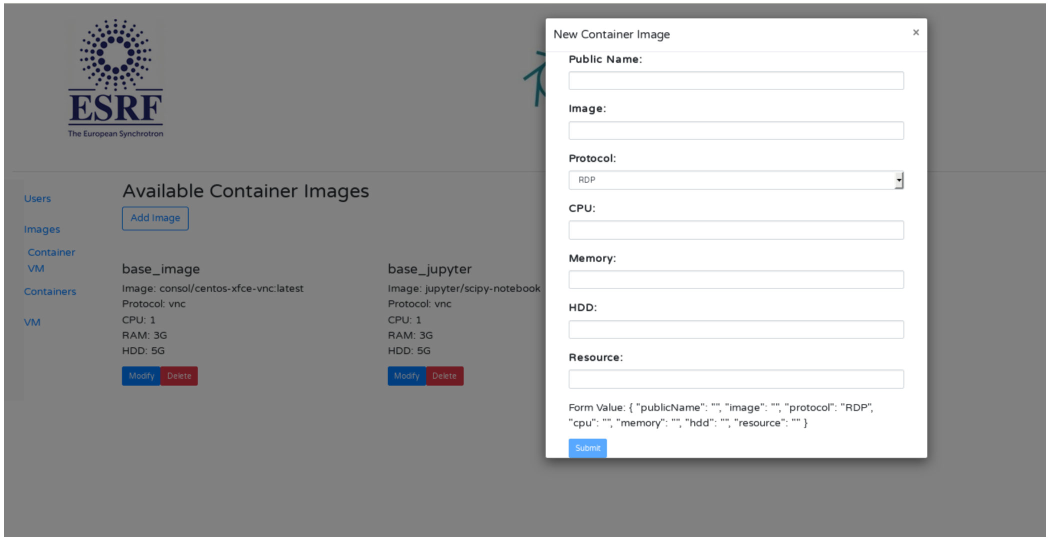
- Virtual machines can be created using multiple systems used by each site.
- ESRF:
  - KVM
- Others:
  - OpenStack
  - KVM
  - Citrix
  - Vsphere / VMWare
- Creating a container can take time
  - Maintain “bank” of virtual machines available at all times

Containers

- Previously
  - Limited to Docker
  - Limited to a single machine (not scalable)
  - Apache Guacamole
  - NFS for data access
- Now
  - Kubernetes
    - Scalability
    - Orchestration
  - Apache Guacamole but in the same browser
  - NFS for data access

Administration

- View resource usage for Containers and Virtual Machines
- Upload new container and vm images
- Manage all containers and virtual machines
- Manage all users



Available Container Images

base_image
Image: consol/centos-xfce-vnc:latest
Protocol: vnc
CPU: 1
RAM: 3G
HDD: 5G

base_jupyter
Image: jupyter/scipy-notebook
Protocol: vnc
CPU: 1
RAM: 3G
HDD: 5G

New Container Image

Public Name:

Image:

Protocol: RDP

CPU:

Memory:

HDD:

Resource:

Form Value: { "publicName": "", "image": "", "protocol": "RDP", "cpu": "", "memory": "", "hdd": "", "resource": "" }