

Deliverables

Deliverable Number	D24.2
Deliverable Title	Blueprint on implementing a DAAS platform
Lead Beneficiary	ESRF
Type	Report
Dissemination Level	Confidential, only for members of the consortium (including the Commission Services)
Due date of delivery	Month 18 (originally month 12 but extended by 6 months)

Deliverable report

D24.2

Blueprint on implementing a DAAS platform (*month 12*)

WP24

Demonstrator of a Photon Science Analysis Service (DaaS)

deliverable report

PROJECT DETAILS

PROJECT ACRONYM

CALIPSOplus

GRANT AGREEMENT NO:

xxxxxxxxx

START DATE

01/05/2017

PROJECT TITLE

Demonstrator of a Photon Science Analysis Service (DaaS)

CALL Horizon 2020-H2020-INFRAIA-2016-2017

INFRAIA-01-2016-2017: Convenient Access to Light Sources Open to Innovation, Science and to the World (CALIPSOplus)

DELIVERABLE DETAILS

WORK PACKAGE ID

WP24

WORK PACKAGE TITLE

WP24

WORK PACKAGE LEADER

Mirjam van Daalen (PSI)

Co-lead Andy Götz (ESRF)

DELIVERABLE ID

D24.2

ESTIMATED INDICATIVE PERSON MONTHS

12

NATURE

☒ R - Report

☐ P - Prototype

☐ D - Demonstrator

☐ O - Other

EXPECTED DATE

Month 12 - 31/05/2018 – extended to Month - 18
31/10/2018

DELIVERABLE TITLE

Blueprint on implementing a platform and manuals for the implementation at the different sites (Month 12)

DELIVERABLE DESCRIPTION

Blueprint on implementing a DAAS platform and manuals for the implementation at the different sites

PERSON RESPONSIBLE FOR THE DELIVERABLE

Aidan Campbell (ESRF)

DISSEMINATION LEVEL

☐ P - Public

☐ PP - Restricted to other programme participants & EC:

☐ RE - Restricted to a group:

☐ CO - Confidential, only for members of the consortium

REPORT DETAILS

VERSION

1

DATE

24/10/2018

FOR MORE INFO PLEASE CONTACT

STATUS

☐ Template

☐ Draft

☒ Final

☐ Released to EC

NUMBER OF PAGES

21

Aidan Campbell (ESRF)
aidan.campbell@esrf.fr
0041/56/3105674

DELIVERABLE REPORT AUTHOR(S)

Aidan Campbell (ESRF)
Andy Götz (ESRF)
Frank Schlutzen (DESY)
Stefan Egli (PSI)
Daniel Salvat (ALBA)

Table of Contents

PROJECT DETAILS	1
DELIVERABLE DETAILS	1
REPORT DETAILS	2
Table of Contents	3
Description of Work	5
1. Introduction	5
2. Blueprint architecture	6
Infrastructure requirements	6
Portals	7
Access Portal	7
Authorisation Database	7
Facility Portal	7
Resource Quota Database	7
AAI and Local Account	8
Data Catalogue	8
Service Catalogue	8
Orchestrator	8
Servers	8
3. User Portal	9
4. Authorisation and Authentication Infrastructure (AAI)	9
UmbrellaID	9
Site Specific Login	9
Authentication for Remote Desktop on Virtual Machines	9
5. Data catalogues	9
6. User Workflow	10
DAAS Portal	10
Step 6 - Remote Desktop	12
7. Orchestrator	14
Jupyter Notebooks	14
Preconfigured Virtual Machines	15
Image with all software installed	15
Image with software for each technique	15
Virtual machine(s) only for that user/team	15
Virtual machine(s) with GPU access	15
Remote Desktop	16
Guacamole Architecture	16
Security	17

8. Software Repository	17
Software Catalogue	18
9. Site Implementation Summary	18
Site Implementation Details	19
Conclusion	21

Description of Work

WP24: Demonstrator of a Photon Science Analysis Service (DaaS)

1. Introduction

The JRA2 work package of CALIPSOplus will produce a prototype of Data Analysis as a Service for synchrotron facilities. This deliverable presents a blueprint of the proposed architecture and implementation. The aim is to have a common approach while still taking into account the differences in the environment and ITC setup at the JRA2 partner sites.

In order to make synchrotrons more accessible to scientists and researchers, synchrotrons need to help users analyse their data by providing a data analysis service for new and experienced users. Some users do not even come to the synchrotrons and instead use a “mail-in” service which allows scientists and researchers to mail the sample to a synchrotron with instructions on how to perform the experiment. The results are then accessible through a web portal with all of the tools necessary to do the analysis. In addition the data are increasingly too big to export and therefore stay at the synchrotrons. Recently most synchrotrons have adopted data policies to curate and archive data for users. The obvious and necessary next step is to provide a remote data analysis service.

Many of the tools that users need to analyse experiment data are often limited and restricted so that the user must be at the facility. One of the goals of the JRA2 work package is therefore to provide these services online so that a user can use these tools remotely from anywhere. The long-term goal of this work package is to make synchrotrons more accessible to users in offering an easier and faster way to analyse their data remotely. Pooling know-how and solutions will allow us to foster collaboration between the JRA2 partner sites on development of common infrastructure architectures and data analysis software including developing, sharing and packaging of codes.

JRA2 will develop a common portal to be installed at each partner site with a common set of core functionalities. Each facility will adapt and enhance the portal with site specific services for their beamline portfolio, the different techniques and experiments carried out, how the user accesses their experiment data and the site specific infrastructure configurations.

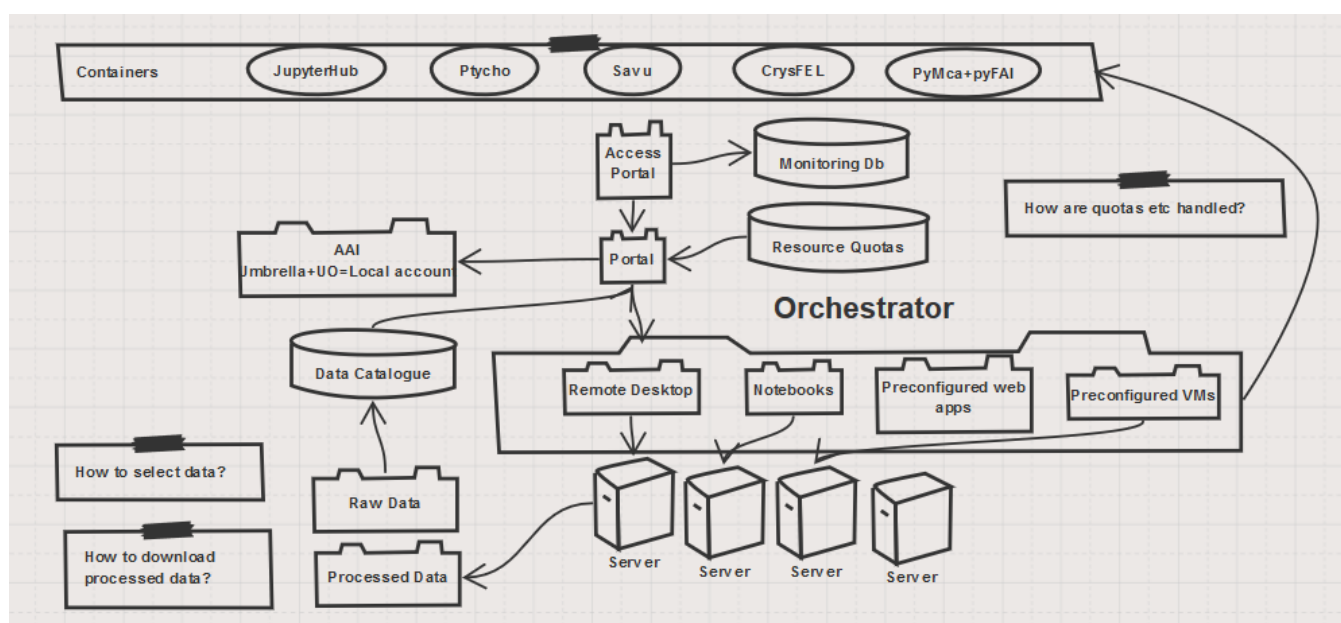
2. Blueprint architecture

At present, we plan to provide four main services which users can access remotely to perform their data analysis, these include:

1. Python notebooks,
2. Virtual machines,
3. Remote desktop applications,
4. Web applications for data analysis

The blueprint was prepared at a meeting in May 2018 (<https://indico.esrf.fr/indico/event/17/>) where the partner met to discuss requirements for the blueprint and to design the potential architecture and implementation of these services. The blueprint design was completed at a Hands-on meeting held at ALBA in October 2018 (<https://indico.esrf.fr/indico/event/18/>).

The resulting architecture is sketched below:



It was agreed that applications will be implemented as deployable packages, as pre-configured virtual machines or as containers. Virtual machines or containers provide encapsulated user environments, which can eventually be archived with the experimental data, thereby capturing valuable provenance data and strongly supporting reproducibility of the original experiment and data analysis workflows.

Infrastructure requirements

The infrastructure on which the DAAS services will run must be adequately dimensioned for the service to be a success. Even if the blueprint could run on a single computer it will require one or more clusters of machine to provide a useful service. The exact number of machines will depend on the use cases run at each site and the number of users. The standard set of use cases and their hardware requirements are summarised in this table:

Use Case	Technique	Software environment	Minimum h/w	Optimised h/w	HPC	Typical dataset(s)
CrystFEL	serial crystallography	Linux	1 x CPU	N x CPUs (N depends on dataset)	YES	10 to 100 TB

PyMca	spectroscopy	Linux, Windows, MacOS	1 X CPU	1 x CPU	NO	10s MB to 10s GB
PyFAI	diffraction	Linux, Windows, MacOS	1 x CPU	1 x GPU + 1 x CPU	NO	100s MB to 100s GBs
Savu	tomography	Linux	1 x CPU	N x CPUs + N x GPUs (N depends on dataset)	YES	10s GB to 100s GB
Ptycho	ptychography	Matlab	1 x CPU	1 x CPU	NO	10s GB to 100s GB

Portals

As part of the blueprint architecture, it was decided to have a central hub from which users can access the data analysis as a service portals at each facility. This will allow us to have one point of entry and the user to choose the appropriate facility depending on where they did their experiment or depending on the most suitable facility if the proposal is carried across multiple facilities.

Access Portal

Authorisation Database

We will have an access portal which can be viewed by anyone which can be used for both promoting the abilities of synchrotrons as well as providing access to our site specific portals. Attached to the access will be a authorisation database in order to prevent unauthorised access. Calispoplus has chosen UmbrellaID¹ as the main authentication mechanism. UmbrellaID allows anyone to make an account therefore we must add a database to make sure that only authorised users (those who carried out an experiment at the facility) with UmbrellaID accounts have access to the services offered by the DAAS portal.

Users at each synchrotron will have an account which will give them a much greater access to the access portal services than a public account created by students, teachers etc. This will also be relevant to synchrotron access as users that have carried out experiments at the ESRF would have no reason to access the Diamond portal in the UK.

In the future the Calipsoplus DAAS portal could be a thematic service in the EOSC for promoting common PaN services like the software catalog, tutorials, documentation, open data etc.

Facility Portal

The facility portal will be the portal implemented by each synchrotron with varying services offered but the core functionality will be offered by most if not all sites.

Resource Quota Database

A resource quota database must be connected to the portal in order to make sure that the user is only making use of a number of resources proportional to their experiment/proposal. The importance of a resource quota is to make sure that a single user or a small group of users are not using all of the resources and limiting access for others. A resource quota will allow us to control the number of CPUs, RAM, storage and time allocated to each user via the DAAS portal which will also depend on what they are required to do. A user carrying out

¹ <https://www.umbrellaid.org/>

computational heavy simulations will require more resources and this will be accommodated for, but not every user will need this or be given additional resources.

For Jupyterhub, the initial plan is to give each user the same amount of resources e.g storage, RAM and CPU which they can use. In a second stage another Jupyterhub instance which will allow for Notebooks to be created with more resources for some users who require this. This additional instance will also support GPUs if available.

AAI and Local Account

At each facility portal, the user will be able to have access to the portal providing they have authenticated via the common authentication mechanism (UmbrellaID) and they have the appropriate access. However, facilities will have the option to give users the ability to access a facility portal by using the facility credentials they will have created for them. This will give us more flexibility in extending our services to support UmbrellaID while also providing access to engineers and administrators who may not have an UmbrellaID account with the appropriate rights.

Data Catalogue

The facilities data catalogue will also be linked to the facility portal as the user will have to specify which dataset(s) they wish to have access to either on a virtual machine or for use on JupyterLab. The user will enter some details about the dataset such as the proposal number, the name of the beamline, session, date of experiment etc and the data catalogue API will be able to find the data and return it to the user in the most efficient way. This data will be mounted directly by the virtual machine they have access to or by the JupyterLab container. This operation will remain the same for the user no matter if it is stored on a hard drive at the facility. Data that is archived is not restored automatically. It will require a manual request via to the data portal to restore the data. The user will need to specify where the data has been restored to access it.

Service Catalogue

The DAAS portal provides a mechanism for specifying different services provided and which beamlines need which services.

Orchestrator

The orchestrator will coordinate containerised services started via the portal for multiple users . The goal is to provide remote desktop, pre-configured virtual machines, Notebooks and pre-configured web applications in our portal. The user will be able to select which service they wish to use.

Servers

Each of these services will require servers in order to host them. In the case of Jupyter Notebooks, they will be hosted on a Kubernetes cluster which will allow us to balance many containers across multiple servers however it will be Kubernetes who will decide which container to run on which server. With this setup, we will be able to add and remove servers depending on the number of users and their requirements.

As part of the services we supply, we will also be able to provide containers with pre-configured software such as CrystFEL and Ptycho installed. We will still need to test graphical applications in containers to see if containers are a viable option. We will use Docker and Singularity containers. We will also need to take into account not only performance but also hardware requirements as virtual machines are much more resource intensive.

One of the many decisions made was to use containers instead of virtual machines for software such as JupyterHub, PyMCA and PyFAI which are python based and can be used in a Jupyter Notebook/Jupyter Lab. In order to best manage Jupyterlab instances, we will be running JupyterHub which will make a new Lab instance when it is requested by a new user. When configuring JupyterHub, we provide pre-configured Jupyterlab images so that additional software such as the ones listed above can be installed during setup and used by the users immediately.

Containers allow us to run multiple Notebooks/Labs on a single computing node which allows us to less resources than if we used a separate virtual machine for each instance.

3. User Portal

As each facility has its own installation for providing services, its possible that some facilities can offer different services to each of its users. It was decided not to have a single common portal in which users can log on and authenticate and then be redirected to the portal created by the facility they wish to access due to the limited services it would offer i.e. only authentication.

The user will be guided through each stage with instructions and they will be asked for the necessary information in order to carry out their request. This includes requests for proposal data to find the correct dataset(s) and provide the user with the appropriate software to be installed on a virtual machine or container.

The implementation of the portal is a web application written in Angular 6 which relies heavily on components and services which allow for new data analytics services to be added once developed as well as remove old services without affecting other services which are provided.

4. Authorisation and Authentication Infrastructure (AAI)

AAI enables users to use their “home” organisation credentials to log in and access data and services needed from multiple providers. Service providers are able to control and manage the rights of their users and create a different access level for each user as well as for groups based on projects.²

UmbrellaID

UmbrellaID is an Identity Provider (IdP) for the users of the (European) large neutron and photon facilities³. UmbrellaID allows each user to have a unique identity which will not only allow access to each of the synchrotrons involved in this project by using the same username and password, but also allow access to research data and dedicated computing resources.

UmbrellaID will be used in this project to allow access to a common web portal after which the user will choose which synchrotron resources they want to use without the need to re-authenticate.

It is also possible that some facilities may not yet have configured all of their services to be accessed using UmbrellaID. An alternative authentication mechanism (e.g. EduGain) or a site specific login will therefore be necessary until all of these services support UmbrellaID. This will depend on each individual synchrotron.

Site Specific Login

Each synchrotron will be able to implement additional authentication methods to their local site portal however UmbrellaID will be the preferred authenticator for the accessing the portal.

Authentication for Remote Desktop on Virtual Machines

In order to increase the security and reduce the risk of unauthorised access of remote desktop on virtual machines, the remote desktop application will only create the connection with the virtual machine if the signed in user is a member of a group with access to that machine. This will allow for multiple users to use the same virtual machine thus giving access to administrators, researchers and support teams.

5. Data catalogues

A very important issue is that the user is able to access all of the data relating to their proposal. The default granularity supported will be at the level of Proposal. More detailed data searches for datasets will be done via the data portal and not via the common DAAS portal. It is up to the facility data portal to make a link to the DAAS portal services.

² <https://www.elixir-europe.org/services/compute/aai>

³ <https://umbrellaid.org/>

The critical issue of making data accessible in an efficient way via a local mount. The type of mount will depend on the local facility file system(s).

6. User Workflow

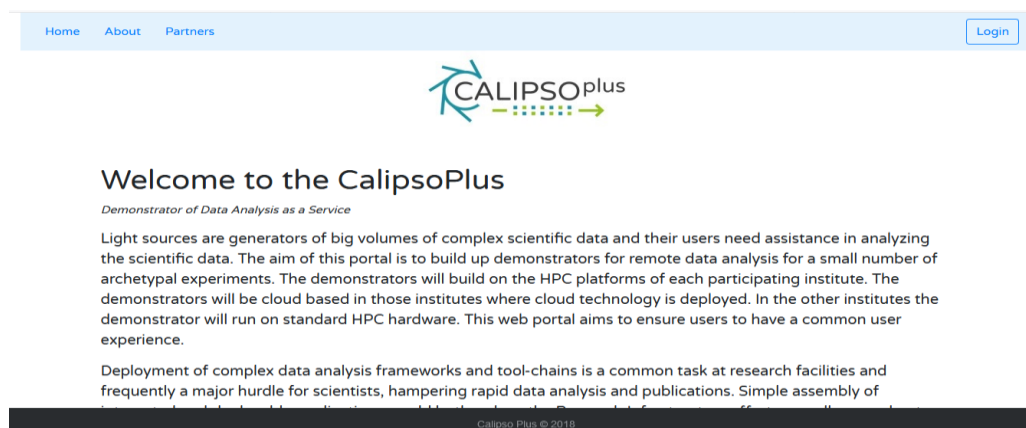
The typical user workflow is as follows:



DAAS Portal


The DAAS portal is the main entry point for users. The following screenshots demonstrate the user experience.

Step 1 - Welcome screen




Step 2 - Login

[Home](#) [About](#) [Partners](#) [Login](#)





Click on the logo bellow to sign in with your UmbrellaID



Or
Click [here](#) to sign in with your local account

Calipso Plus © 2018

Step 3 - Select Proposal



calipso3

PROPOSALS
RESOURCES
QUOTAS
FAVORITES

Proposals

Search...

★	Proposal Id	Title	Abstract	Beamline	
★	20180301	test 1 C	this is a description1	BL13	<input type="button" value="Desktop"/>
★	20180302	test 2 C	this is a description2	BL13	<input type="button" value="Desktop"/> Desktop Jupyter
☆	20180303	test 3 C	this is a description3	BL13	<input type="button" value="Desktop"/>
☆	20180304	test 4 C	this is a description4	BL13	<input type="button" value="Desktop"/>
☆	20180305	test 5 C	this is a description5	BL13	<input type="button" value="Desktop"/>
☆	20180306	test 6 C	this is a description6	BL13	<input type="button" value="Desktop"/>
☆	20180307	test 7 C	this is a description7	BL13	<input type="button" value="Desktop"/>

Calipso Plus © 2018

Step 4 - Start / Monitor service



calipso3

PROPOSALS
RESOURCES
QUOTAS
FAVORITES

Proposals

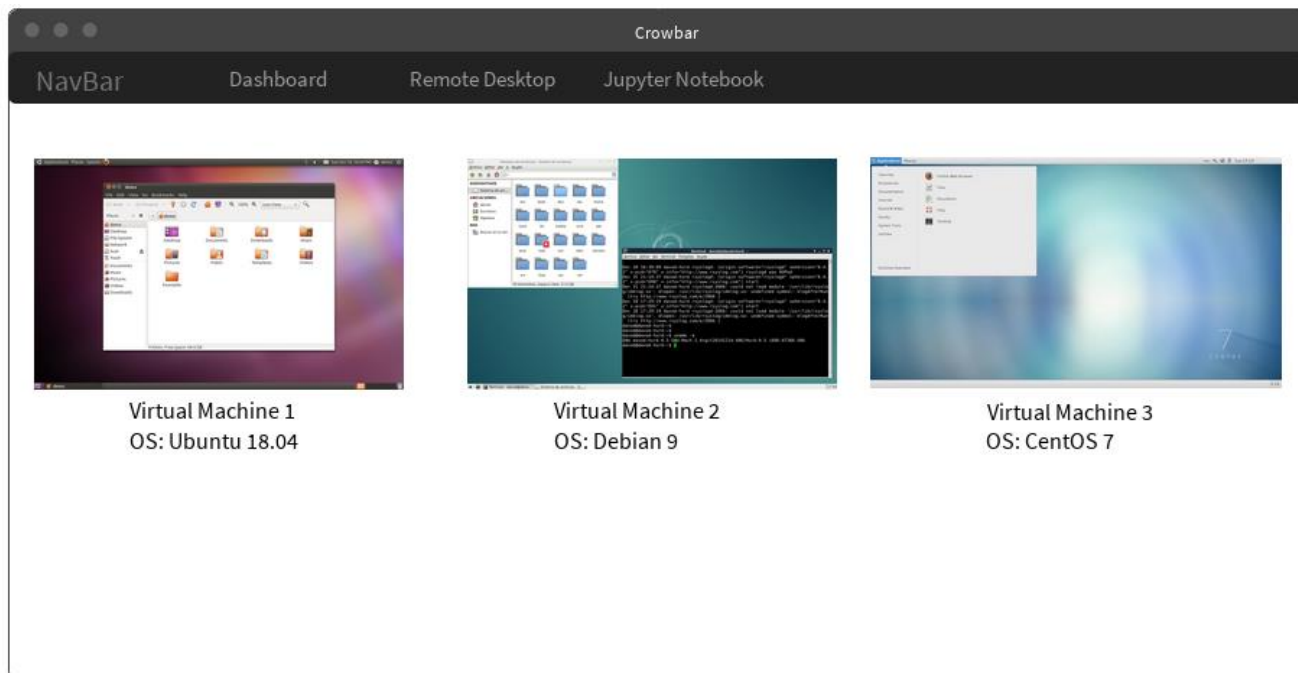
Search...

★	Proposal Id	Title	Abstract	Beamline	
☆	20180301	test 1 C	this is a description1	BL13	<input type="button" value="Stop"/> <input type="button" value="Desktop"/>
☆	20180302	test 2 C	this is a description2	BL13	<input type="button" value="Stop"/> <input type="button" value="Jupyter"/>
☆	20180303	test 3 C	this is a description3	BL13	<input type="button" value="Wait"/>
☆	20180304	test 4 C	this is a description4	BL13	<input type="button" value="Desktop"/>
☆	20180305	test 5 C	this is a description5	BL13	<input type="button" value="Desktop"/>
☆	20180306	test 6 C	this is a description6	BL13	<input type="button" value="Desktop"/>
☆	20180307	test 7 C	this is a description7	BL13	<input type="button" value="Desktop"/>

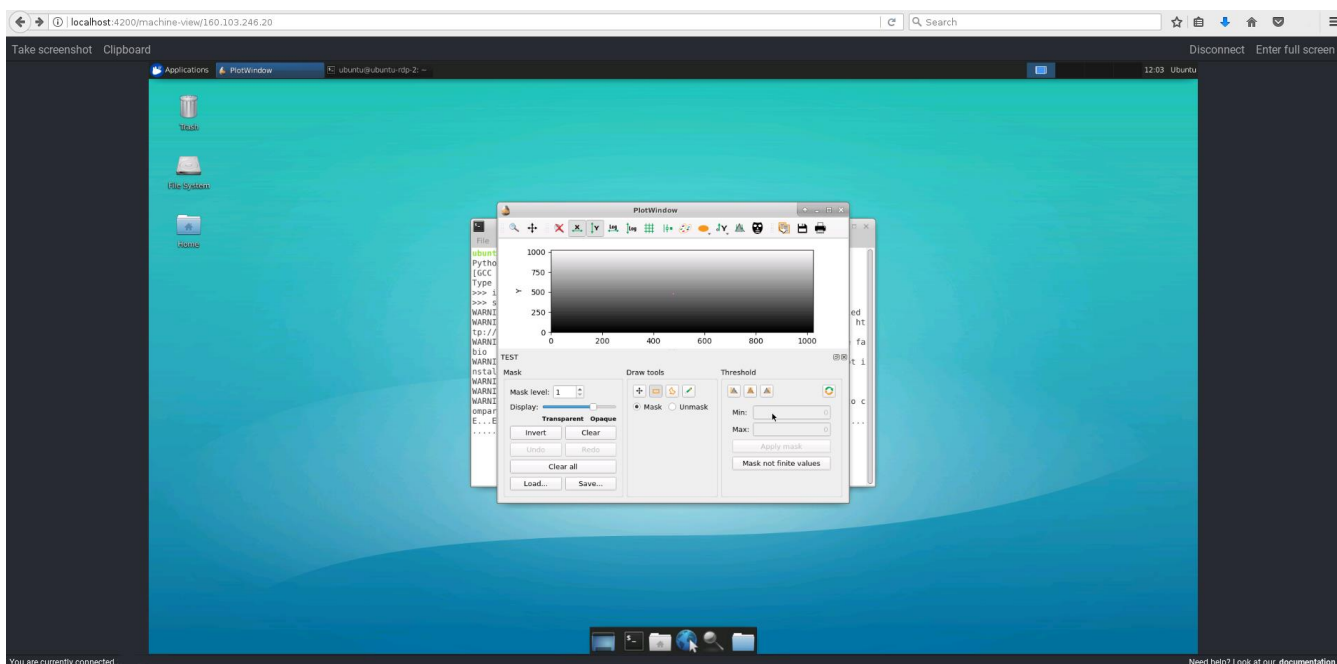
Calipso Plus © 2018

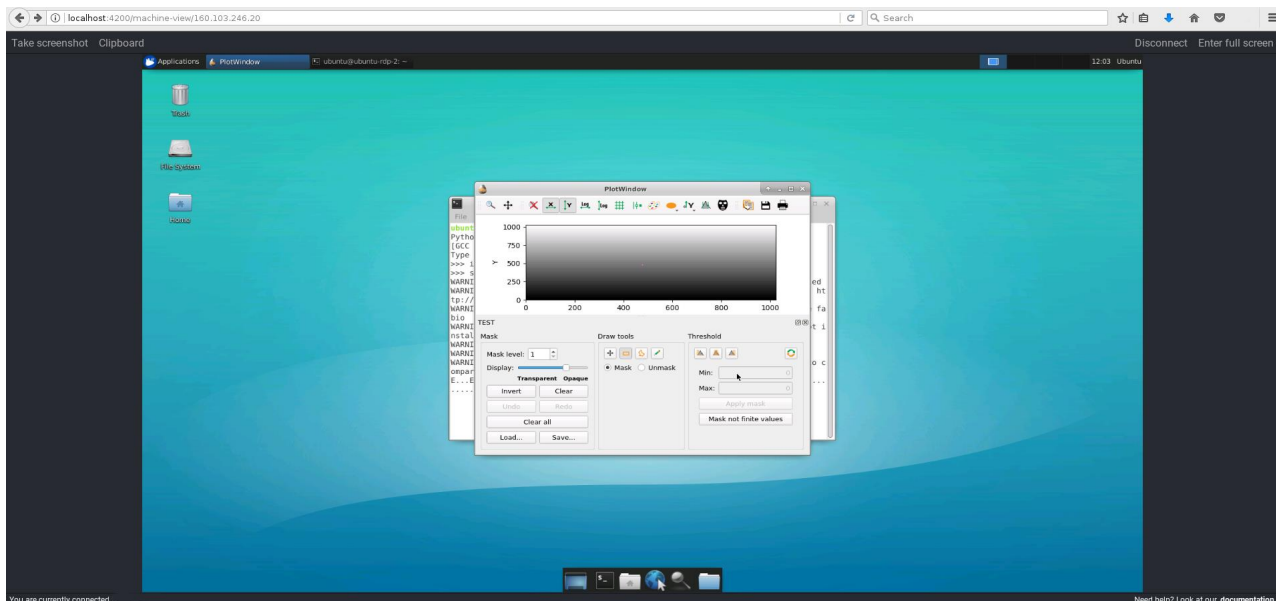
Step 6 - Remote Desktop

If the user selects Remote Desktop, they will be brought to the following screen with a list of virtual machines they can access.





Once checks have been carried out to ensure the user had access to this virtual machine, the browser will immediately show the GUI of the virtual machine as shown below:





Step 6 - Resources used





calipso3

Resources

PROPOSALS
RESOURCES
QUOTAS
FAVORITES

RESOURCE	RESOURCE	RESOURCE
<p>Proposal ID : 20180301</p> <p>Machine name: youthful_borg</p> <p>Machine type: base_image</p> <p>Creation Date : 24/10/2018 7:24</p> <p>Expiration Date : -</p> <p>Last Access : 24/10/2018 9:25</p> <p>ENTER</p>	<p>Proposal ID : 20180302</p> <p>Machine name: youthful_cray</p> <p>Machine type: base_jupyter</p> <p>Creation Date : 24/10/2018 7:24</p> <p>Expiration Date : -</p> <p>Last Access : -</p> <p>ENTER</p>	<p>Proposal ID : 20180303</p> <p>Machine name: wonderful_dubinsky</p> <p>Machine type: base_jupyter</p> <p>Creation Date : 24/10/2018 7:24</p> <p>Expiration Date : -</p> <p>Last Access : -</p> <p>ENTER</p>

Step 7 - Quotas



calipso3

Quotas

PROPOSALS
RESOURCES
QUOTAS
FAVORITES

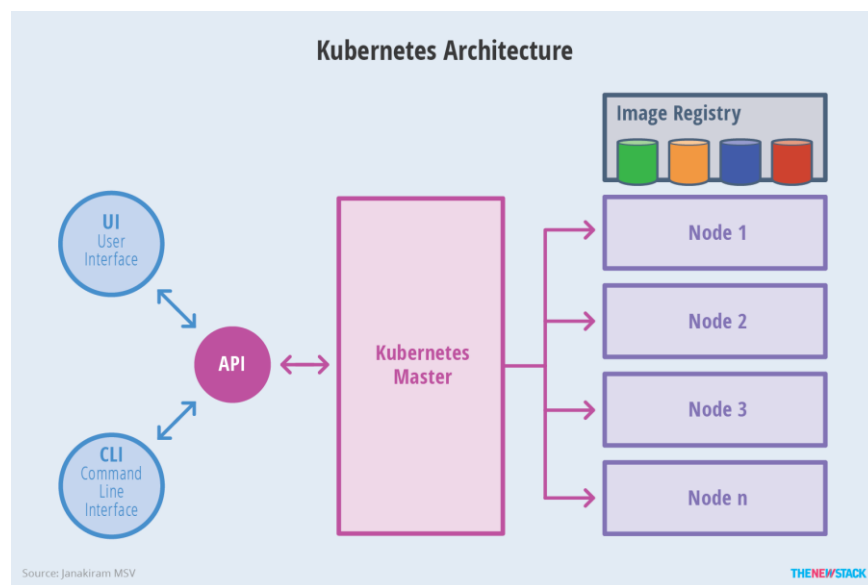
Your default quota	Your used quota	Your available quota	Default image
<p>machines : 5</p> <p>cpus : 10</p> <p>memory : 30G</p> <p>storage: 80G</p>	<p>machines : 3</p> <p>cpus : 3</p> <p>memory : 9G</p> <p>storage: 15G</p>	<p>machines : 2</p> <p>cpus : 7</p> <p>memory : 21G</p> <p>storage: 65G</p>	<p>-</p> <p>cpus : 1</p> <p>memory : 3G</p> <p>storage: 5G</p>

7. Orchestrator

The Orchestrator for the DAAS blueprint is Kubernetes⁴. From our blueprint, it has been agreed to use the orchestrator for all container-based services like Jupyter Notebooks and desktop containers..

Kubernetes is an open-source system initially developed by Google which manages containerised applications in a clustered environment.⁵ At a basic level, it is a system for running and coordinating containerised applications across a cluster of machines while able to completely manage the life cycle of the application by having the ability to use method which offer high predictability, availability and scaling. We are able to scale up or down our cluster and services within the cluster depending on user need.

The benefits of using Kubernetes is that we are able to dynamically allocate persistent volumes which will store the Notebook(s) of each user through interacting with OpenStack Cinder. We are also able to heavily control the



number of resources for each user such as the number of CPUs, the RAM for each Notebook etc.

In term of infrastructure, we are also able choose how the containers are deployed on the cluster including allocating the containers on a node with a GPU if it is needed for a specific task/experiment. This can be expanded to allow us to choose the scheduling and deployment method of each container so that the number of containers is roughly balanced across each node or we could decide to fill each node before deploying on other nodes.

Kubernetes will allow us to constantly expand or shrink the cluster depending on how many containers we need including adding heterogeneous hardware. Kubernetes will be able to balance the containers using a wide range of hardware specifications.

Jupyter Notebooks

JupyterHub, a multi-user Hub, spawns, manages, and proxies multiple instances of the single-user Jupyter notebook server. JupyterHub can be used to serve notebooks to a class of students, a corporate data science group⁶, or a scientific research group⁶.

Jupyter Notebooks also come with the additional challenge of ensuring that the python libraries our users will need are installed on each Notebook image. Luckily Jupyterhub has released a Docker image that we can deploy using Kubernetes that has many of the scientific libraries installed but it will not have software created at the

⁴ <https://kubernetes.io/>

⁵ <https://www.digitalocean.com/community/tutorials/an-introduction-to-kubernetes>

⁶ <https://jupyterhub.readthedocs.io/en/stable/>

ESRF such as *silx* or CrystFEL for DESY. By modifying the Dockerfile, we are able to add any additional software which will be installed when the image is loaded into a container.

Jupyterhub also gives us the ability to use Jupyterlab in addition to Jupyter Notebooks. This is easily configurable in the Jupyterhub configuration file and will allow our users to use multiple Notebooks at the same time.

Preconfigured Virtual Machines

At present, some sites offer users to make use of virtual machines in which they can analyse their results from their experiment. They currently make use of this through Secure Shell (SSH).

The blueprint proposes to optionally provide this service to our users however it is necessary to redevelop the architecture of this service in order to provide virtual machines to our users as quickly as possible with the required software already installed.

Due to the amount of time required to create a virtual machine, install the necessary software or download the required data from an experiment, a number of virtual machines will be created and left idle on our infrastructure in order to allow quick access for potential users. Although this will take up valuable resources, this provides not only quicker access for users but also a much better user experience. By running a simple program, we are able to detect when the number of idle virtual machines is below a certain threshold determined by each site and more virtual machines can be created. This list of idle virtual machines will now be referred to as a virtual machine bank.

It could also be possible to create the virtual machine on-the-fly depending on the purpose as well as the size of the dataset that has to be transferred to it. We will also be able to access the Python OpenStack Nova API to deploy virtual machines if additional virtual machines are requested by that user. This will likely depend on the user quota for resources as we do not want a small group of users to use all of our resources for hours or days.

The software required for each virtual machine will depend on the beamline used to get the results. There are multiple approaches that can be taken in order to provide all of the software needed. These include:

Image with all software installed

In order to meet the requirements for all users and for all experiments, we are able to create an image with all of the software needed for all beamlines. Using this image, we are able to create pre-configured virtual machines however, this method will put additional strain on our resources as the virtual machines will require more storage in order to store all of the software.

Image with software for each technique

By creating an image for each beamline, we are able to overcome the problem of storage and resources required as each virtual machine will be smaller and thus faster to create. This method is likely to work in most cases however, it could mean that some users will require additional software and they will have to put in requests for the additional software to be installed incase multiple beamlines are used within a single proposal.

Virtual machine(s) only for that user/team

In order to minimise the amount of software required for each virtual machine and reduce the resource requirements, we could allow users the ability to install the software that they need themselves. This would offer the most flexibility to the user but it could increase security vulnerabilities if not implemented correctly. One step to minimise this risk would be for the user to install additional software to their own user directory and update the path variable. This could be difficult for some users and will have to be discussed further.

Virtual machine(s) with GPU access

Some applications (PyFAI and PyNx for example) need access to GPU's in order to run efficiently.

Remote Desktop

By providing virtual machines as a service to our users, we are also limiting them to Secure Shell for applications with a Graphic User Interface. By providing a remote desktop feature, users who are less familiar or less comfortable with the terminal will still be able to analyse their results.

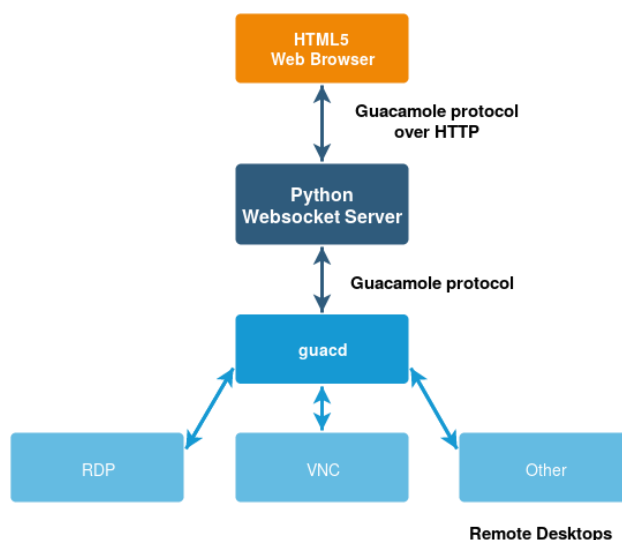
We are currently developing software which will provide this functionality by using Apache Guacamole which can connect to a virtual machine without a specific client. This allows us to access the virtual machine in a web browser thus providing all of our services in a single web portal. Sites which have other remote desktop solutions (NX, FastX, Citrix) have the option of integrating these technologies. They will have to sort out issues how to run this containers etc. locally.

Guacamole allows access one or more desktops from anywhere remotely, without having to install a client, particularly when installing a client is not possible. By setting up a Guacamole server, we can provide access to any other computer on the network from virtually any other computer on the internet, anywhere in the world. Even mobile phones or tablets can be used, without having to install anything.

Apache Guacamole is an HTML5 web application that provides access to desktop environments using remote desktop protocols such as RDP or VNC⁷.

In order to use Guacamole for our existing infrastructure, we are not using the web application provided but building our own custom web application using the Guacamole API which is provided by Apache.

Guacamole Architecture



Guacd is a daemon process which runs in the background which dynamically loads support for desktop protocols and connects them to remote desktops based on instructions received from the web application.

The guacd process will listen to TCP connections from the web application and load the remote desktop protocol when connecting the the desktop. Once the connection has been made, guacd will act as an in-between transmitting messages between the web application and the remote desktop.

DockerHub offers a guacd image which can easily be downloaded and started on our infrastructure in a container which allows for the communication between the web application and the remote desktop. This requires us to pass the ip address of the virtual machine from the Python Websocket to the guacd server who

⁷ <https://guacamole.apache.org/doc/gug/preface.html>

will create the connection. The number of guacd containers can easily be scaled and reduced depending on the number of users at a single time.

At present, a single guacd server is running in a Docker container however we will soon move this to Kubernetes in order to provide consistency with our JupyterHub cluster.

In order to communicate with guacd from the browser, a websocket server was created so that an open connection could be maintained for message passing between the web application and guacd. As soon as the connection is opened, the user authentication is checked and we are able to specify which protocol to use (RDP, VNC etc) as well as the height and width of the terminal to be drawn on the web app.

Security

To provide security for the virtual machines and to make sure that users do not have unauthorised access to experiment data, only the virtual machines which the user can access are shown on the web application. If there are three virtual machines the user can access, the user will be given the choice of which virtual machine to connect to from these three virtual machines.

After the user has made their choice, another check is carried out to make sure that the user has permission to access this virtual machine in order stop users from guessing the hostnames of other virtual machines and gaining unauthorised access.

We will also be able to different credentials for each virtual machine ensuring that even if the user is able to gain access to the virtual machine through RDP, they are not able to get past the initial login screen.

Users will need to agree to a set of **Conditions of Usage** in order to ensure that the DAAS service is used for *bona fide* applications related to the experiments they have conducted.

8. Software Repository

All software developed for the implementation of the proposed architecture is being published on github.com under permissible licenses at <https://github.com/calipsoplus>.

Currently available software products

- crowbar-auth: <https://github.com/Calipsoplus/crowbar-auth>
 - Python Flask server implementation which handles user authentication and retrieves information on virtual machines which each user can access.
- calipsoplus-backend: <https://github.com/Calipsoplus/calipsoplus-backend>
 - CalipsoPlus, django backend
- calipso-docker-demo: <https://github.com/Calipsoplus/calipso-docker-demo>
 - Dockerized CalipsoPlus Demo
- crowbar-guacamole: <https://github.com/Calipsoplus/crowbar-guacamole>
 - Guacamole implementation which allows for communication with a virtual machine using RDP
- crowbar: <https://github.com/Calipsoplus/crowbar>
 - Web application written in Angular 6 to access Synchrotron services such as virtual machines, remote desktop and JupyterHub
- calipsoplus-frontend: <https://github.com/Calipsoplus/calipsoplus-frontend>
 - CalipsoPlus, angular frontend
- calipsoplus-local-login-mock: <https://github.com/Calipsoplus/calipsoplus-local-login-mock>
 - Django Mock Login

The user experience is an important aspect of the facility portal in order to provide ease of use and reduce the volume of support required to complete a task. For this reason, we aim to make the facility portal as easy to use as possible while also providing everything the user needs.

Software Catalogue

The PaNdata software catalogue⁸ is a database of software used mainly for data analysis of photon and neutron experiments. It allows scientific institutes to upload information about the software that they have created for others to download although the website does not contain the binaries for the software.

9. Site Implementation Summary

This section summarises which sites will implement which parts of the blueprint described above. The components are linked to different levels.

The following levels defined are:

- **Level 0 – mandatory**
- **Level 1 - optional**

Component	Level	ESRF	PSI	ALBA	DESY	ELETTRA	SOLEIL	DLS	HZDR
hypervisor	0	KVM	vSphere (KVM)	XenServer	Open Stack	KVM (OpenStack planned)	?	VMWare OpenStack	KVM
cloud platform	0	Open Stack	None (VMWare Suite)	None - Open Stack future	Open Stack	none yet	?	OpenStack	Openstack
operating system	0	Debian	RedHat7	CentOS	Centos	CentOS	CentOS	RH/CentOS	CentOS
orchestrator	0	Kubernetes	Kubernetes (so far only for Web applications)	Kubernetes [+ Rancher future]	Kubernetes	Kubernetes (planned)	Kubernetes	Kubernetes	?
Calipso+ portal	0	YES	planned	YES	YES	YES (planned)	YES	YES (planned)	YES
Jupyter notebook service	0	YES	Yes	No	YES	YES (planned)	YES	YES	YES
data catalog api	0	icat	SciCat	ICAT (planned)	planned	iCAT (planned)	ICAT	ICAT (ISPyB planned)	Under construction

⁸ <https://software.pan-data.eu/>

								d)	
local data access	0	GPFS NFS	GPFS/NFS/SMB	NFS4	various	NFS SMB	NFS	GPFS NFS CIFS	GPFS NFS
UmbrellaID AAI	0	YES	Done for DUO	YES	YES (uo)	YES	YES	YES (via UKAMF)	YES
local user AAI	0	YES	YES	YES	YES	YES	YES	YES	YES
GPU	1	YES	YES	YES	YES	YES	YES	YES	YES
Calipso+ use cases	0	YES	Ptycho Shelves	?	YES	PyMca, Ptycho ?	YES Savu	YES	?
Virtual Machine service	1	YES	VMWare	Citrix XenDesktop	yes, but not for photon users	Proxmox	YES	YES (via STFS IRIS)	YES
Remote Desktop service	1	Guacamole	NoMachine	Guacamole + CitrixXenDesktop	FastX	Guacamole	Guacamole	NoMachine	Citrix, NoMachine, later: Guacamole
Docker	0	YES	YES	YES	YES	YES	YES	YES	YES
Singularity	1	YES	No, but interested	No, but interested	YES	No, but interested		YES	YES

Site Implementation Details

The blueprint is in the process of being implemented at some of the sites. This section lists the current situation at those sites where implementation of the blueprint has been started at the time of the submission of the blueprint deliverable (October 2018).

ESRF:

At the ESRF, we are also using an OpenStack cluster to create pre-configured virtual machines and provide remote desktop capabilities with these virtual machines. The virtual machines can be adapted to run remote desktop by installing the Remote Desktop Protocol and exposing the port.

At the ESRF, the user will be presented with a dashboard showing multiple services that can be used such as access to a virtual machine, remote desktop access or a Jupyter Notebook that they can access and perform the necessary actions.

We are focusing strictly on RDP even though other protocols such as VNC (and Xorg in the future) are supported. We need to ensure that OpenGL / WebGL are supported for the production service. At present, we are currently building the custom web application using Angular 6 for the front end and building a websocket server in Python in order to maintain a constant connection with the guacd server. This will be replaced with the JRA2 DAAS portal in the future.

The user will also be able to enter their own ESRF credentials thus bypassing the common portal. This will mostly be used by our engineers for maintenance, upgrading of software, addition of services as well as by scientists who work solely for the ESRF. Users and ESRF staff will also be able to authenticate using Umbrella on the ESRF portal if they want.

We have not yet implemented a scalable version of Jupyter Notebooks on our Kubernetes cluster but the Notebooks allow the user to use both text and write code in Python to perform data analysis using tools such as SILX and PyFAI and view the results in the form of a graph using Matplotlib.

DESY:

Jupyter instances can be launched in a number of ways. DESY offers services to deploy notebook attached to a HPC cluster hosting all photon science data. The deployment is done with a common batch scheduler (slurm) which permits for certain classes of users to request enhanced resources for rapid data analysis. For less latency demanding tasks, notebooks can be launched to commodity servers either on openstack (kubernetes deployed) or a general purpose batch-farm (using HTcondor as a scheduler). In the latter cases data need to be staged or made accessible via remote protocols.

All photon science users also have the possibility to access a fully graphical desktop environment and execute any of the use cases, either in a light-weight client or via a web-browser. This readily also allows to execute custom jupyter kernels independent of JupyterHubs. The environment also allows for collaborative work in a single desktop environment.

Jupyter instances can be launched in a number of ways. DESY offers services to deploy notebook attached to a HPC cluster hosting all photon science data. The deployment is done with a common batch scheduler (slurm) which permits for certain classes of users to request enhanced resources for rapid data analysis. For less latency demanding tasks, notebooks can be launched to commodity servers either on openstack (kubernetes deployed) or a general purpose batch-farm (using HTcondor as a scheduler). In the latter cases data need to be staged or made accessible via remote protocols.

ALBA:

We are currently hosting a Kubernetes infrastructure in pilot status, used mainly for Continuous Integration, testing and packaging on Docker containers by the Controls Systems Section. After a first testing period, there are plans to upgrade this setup, or even include other extra management layers, such as Rancher.

ALBA is also providing remote data analysis to the MIRAS Beamline users, infrared microscopy beamline, based on the commercial solution Citrix Xen Desktop (VDI - Virtual Desktop Infrastructure) and providing Unscrambler X and Opus software licensed solutions, for reprocessing the datasets acquired during the experiment, which are mounted automatically. This infrastructure, optimized for 3D remote graphics encryption, is currently running on Windows VMs, but has been successfully tested for Linux environments as well.

Several Jupyter instances are available installed at the ALBA Citrix Xen Desktops central service, for testing purposes.

There are additional data analysis services available for staff users and also remotely through Citrix XenApp technology, publishing around 20 Beamline specific applications such as pyMCA, in addition to around 10 scientific calculation applications such as Matlab, Mathematica or Maxima.

ALBA runs an HPC cluster composed by 12 CentOS nodes (2,4 Tflops on 216 CPU cores) and an Nvidia Tesla P100 GPU. It's controlled by Slurm workload manager, which launches jobs with more than 15 installed applications using a BeeGFS scratch of 40TB. There are plans under execution to have its capacity increased with additional nodes and also several Nvidia Pascal GPUs.

Conclusion

From this blueprint, we can see that the proposed architecture and implementation of providing Data Analysis as a Service for synchrotron facilities provide a common user experience and services while still allowing the freedom for each synchrotron to implement site-specific functionality depending on their hardware and experimental procedures.

Initial prototypes and tests show that the main issues required to provide a DAAS service for synchrotrons have been taken into account. There are multiple factors like data access, performance, compute resources which can only be done at the facility infrastructure level. Ultimately the service needs to be tested by the users. This will be done through the user organisations.

Each synchrotron must now begin to add UmbrellaID authentication to their infrastructure and package site specific software for their users.