SWAN: service for web based analysis

D. Castro, E. Tejedor, D. Piparo, P. Mato
E. Bocchi, J. Moscicki, M. Lamanna

https://swan.web.cern.ch

May 24th, 2018
ABP-CWG meeting #25

Introduction

Motivation

> Analysis only with a web browser

- Available everywhere and at anytime
- Integrated with other analysis ecosystems : ROOT, R, Python, ...

> Easy to use (but powerful)

- No local installation and configuration needed

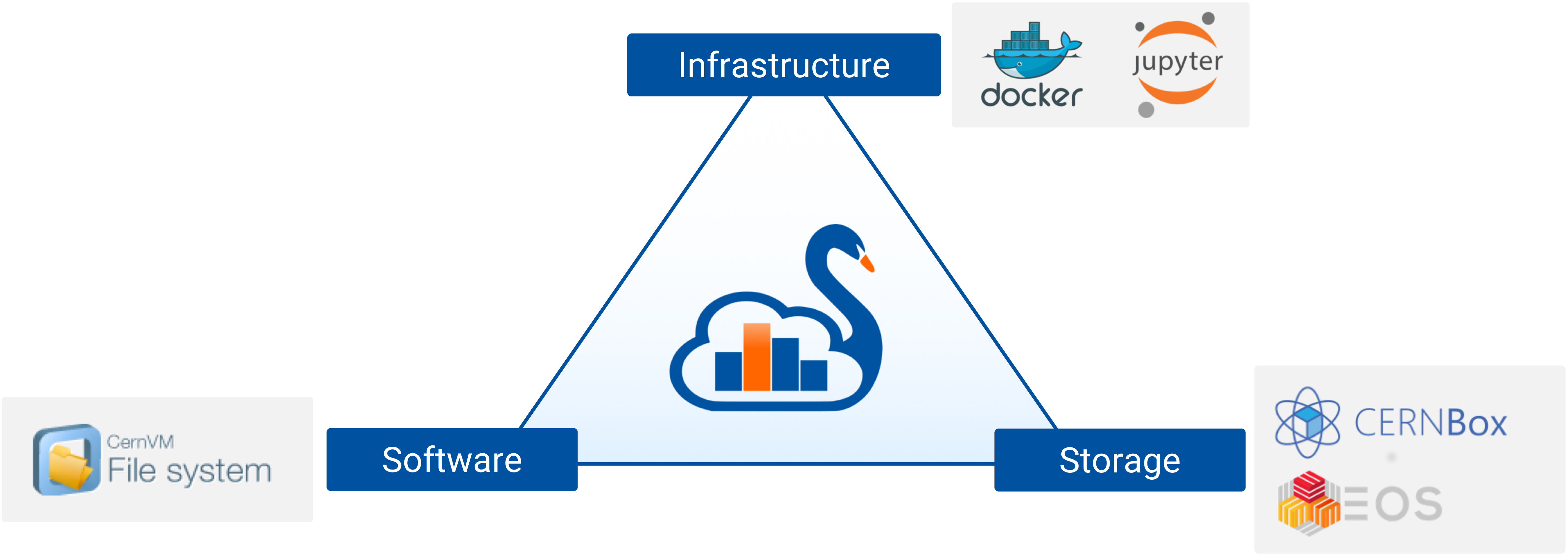
> Create easily sharable scientific results: plots, data, code

> Integration with CERN resources

- Access software, user/experiments data, mass processing power

SWAN

Integrating services



Infrastructure

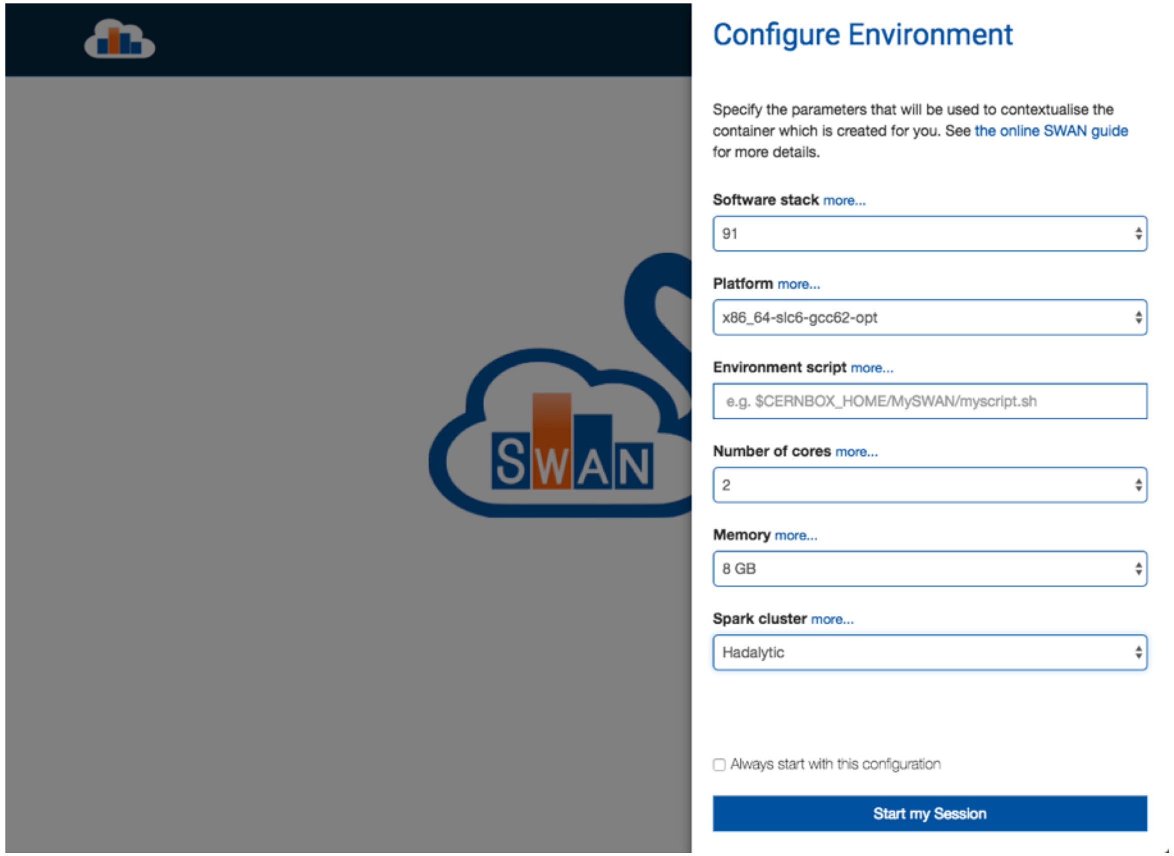
Software

Storage

Jupyter - The Notebook as Interface

- A web-based interactive interface and platform that combines code, equations, text and visualisations
  - Ideal for sharing/collaboration
  - … In a nutshell: an “interactive shell opened within the browser”
- Many supported languages (kernels)
  - In SWAN: Python, ROOT C++, R and now **Octave**
- Interactive, usually lightweight computations
- Very useful for some use cases
  - Final steps of an analysis, Exploration, Teaching, documentation and Reproducibility

SWAN Interface



Configure Environment

Specify the parameters that will be used to contextualise the container which is created for you. See the online SWAN guide for more details.

Software stack more...
91

Platform more...
x86_64-slc6-gcc62-opt

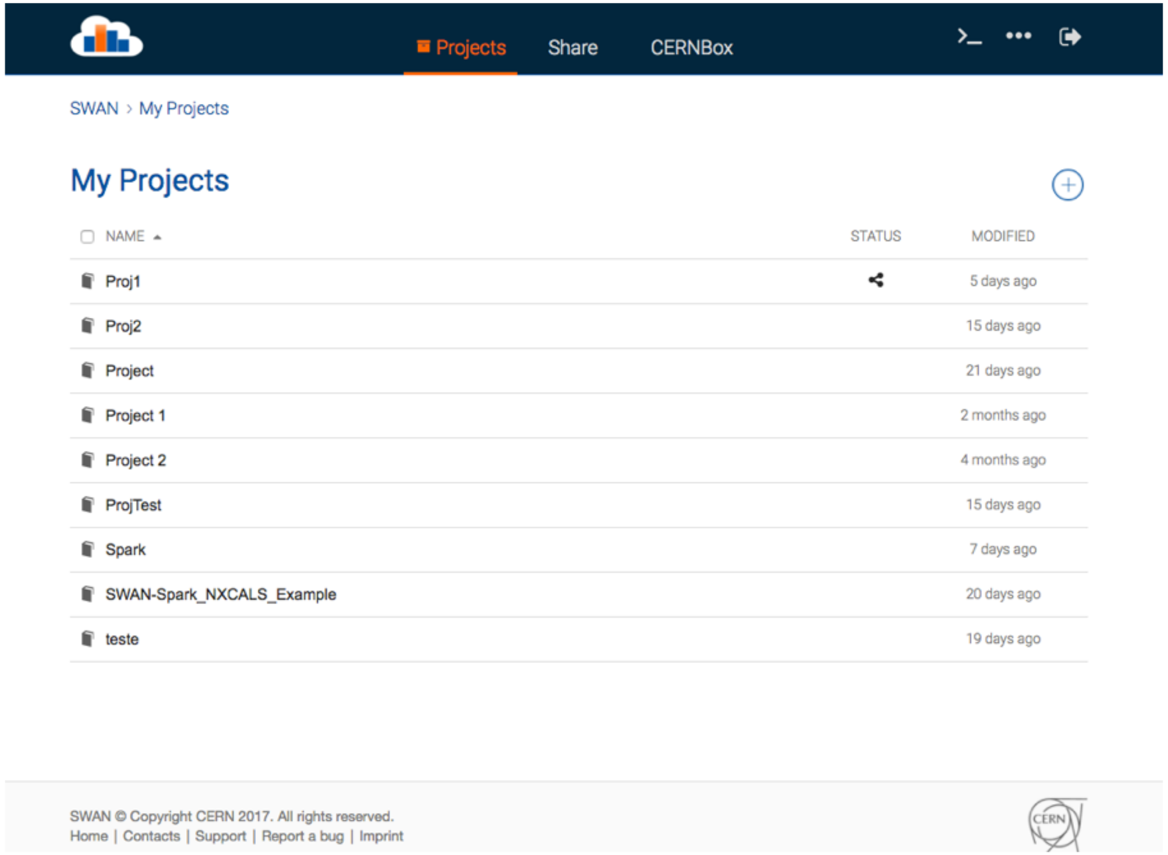
Environment script more...
e.g. $CERNBOX_HOME/MySWAN/myscript.sh

Number of cores more...
2

Memory more...
8 GB

Spark cluster more...
Hadalytic

□ Always start with this configuration



SWAN › My Projects

My Projects

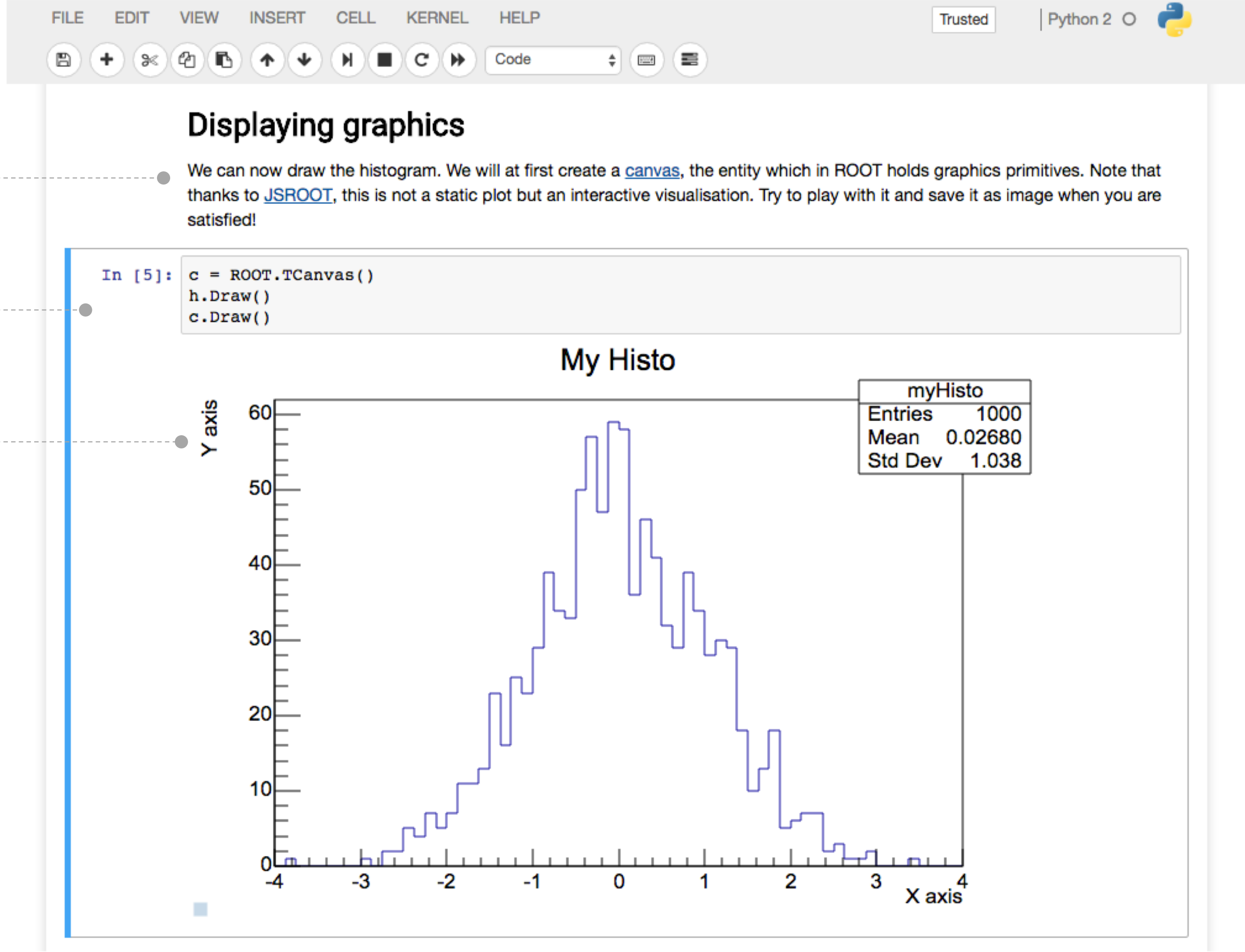
□ NAME ▲	STATUS	MODIFIED
Proj1	⋖	5 days ago
Proj2		15 days ago
Project		21 days ago
Project 1		2 months ago
Project 2		4 months ago
ProjTest		15 days ago
Spark		7 days ago
SWAN-Spark_NXCALS_Example		20 days ago
teste		19 days ago

SWAN © Copyright CERN 2017. All rights reserved.
Home | Contacts | Support | Report a bug | Imprint

Text

Code

Graphics



Displaying graphics

We can now draw the histogram. We will at first create a canvas, the entity which in ROOT holds graphics primitives. Note that thanks to JSROOT, this is not a static plot but an interactive visualisation. Try to play with it and save it as image when you are satisfied!

In [5]:
```
c = ROOT.TCanvas()
h.Draw()
c.Draw()
```

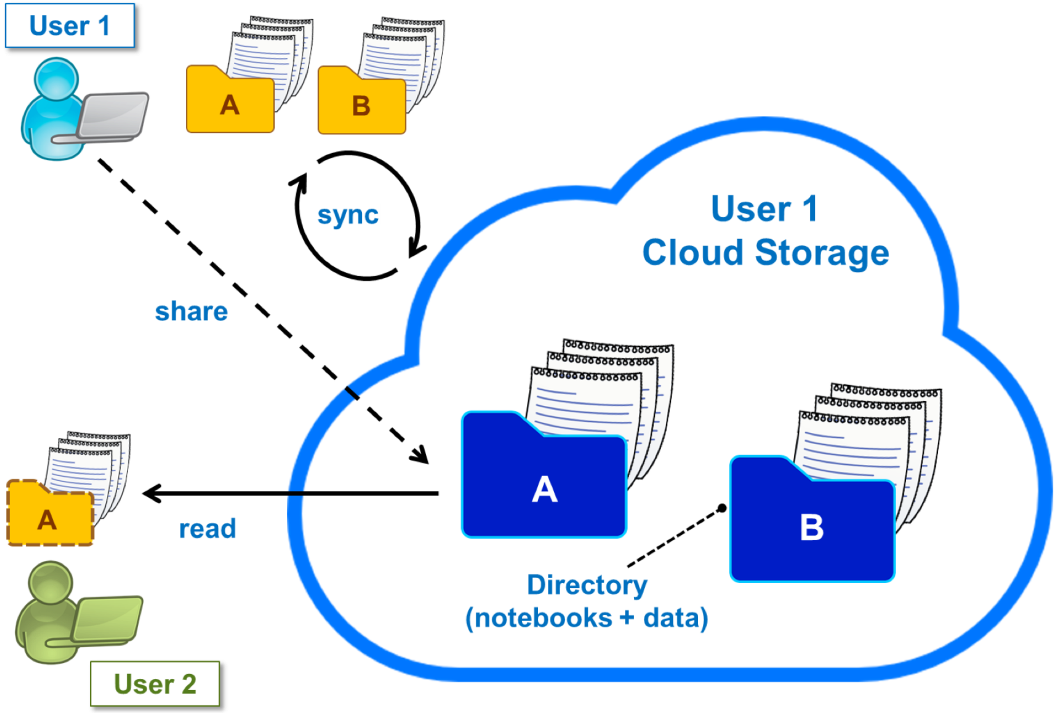
Storage

> Uses EOS mass storage system

- All experiment data potentially available

> User personal space, synchronized through CERNBox

- All files synced across devices, the cloud and other users

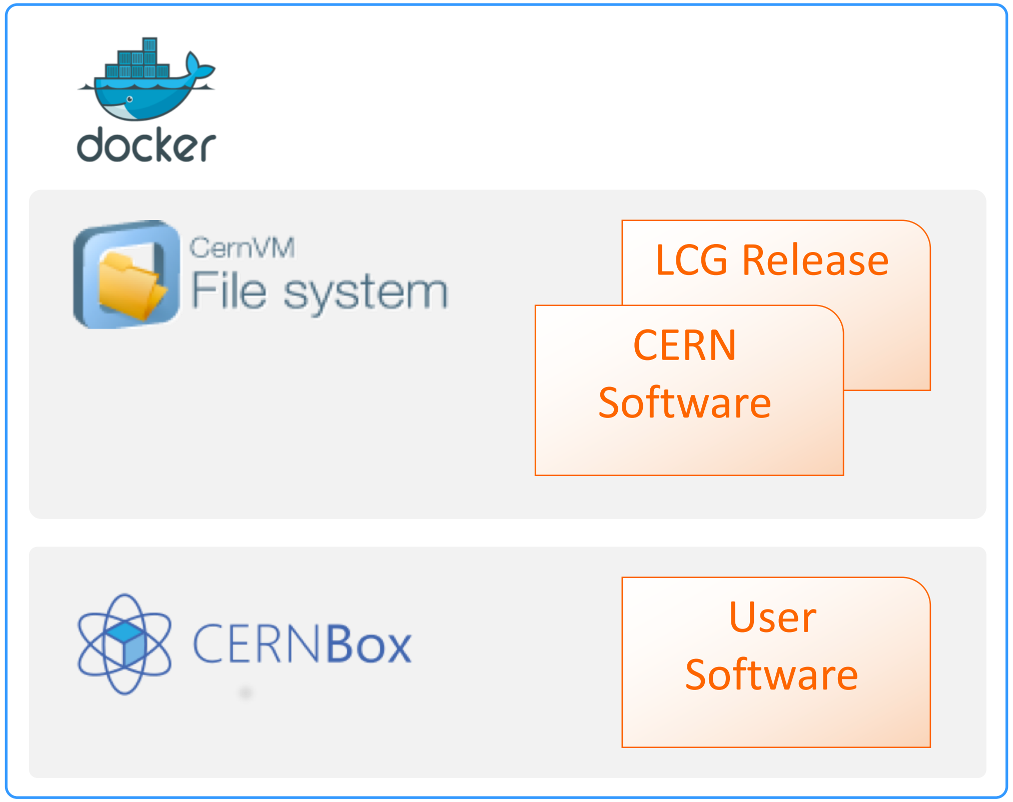


EOS CERNBox

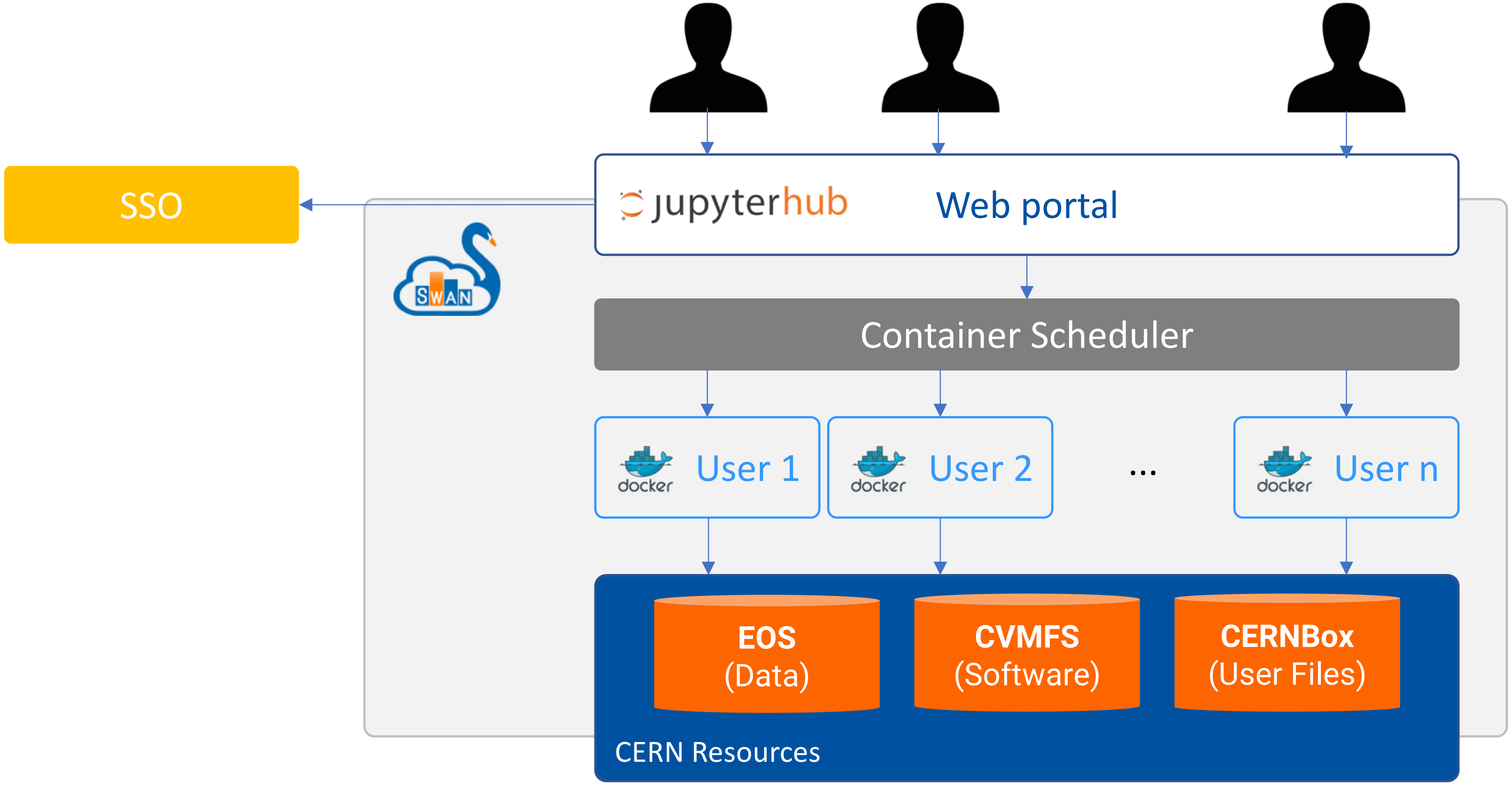
Software

> Software distributed through CVMFS
  - ”LCG Releases” - pack a series of compatible packages
  - Software used by researchers is available

> Possibility to install other libraries in user local storage



Architecture



SSO

jupyterhub Web portal

Container Scheduler

docker User 1

docker User 2

…

docker User n

EOS
(Data)

CVMFS
(Software)

CERNBox
(User Files)

CERN Resources

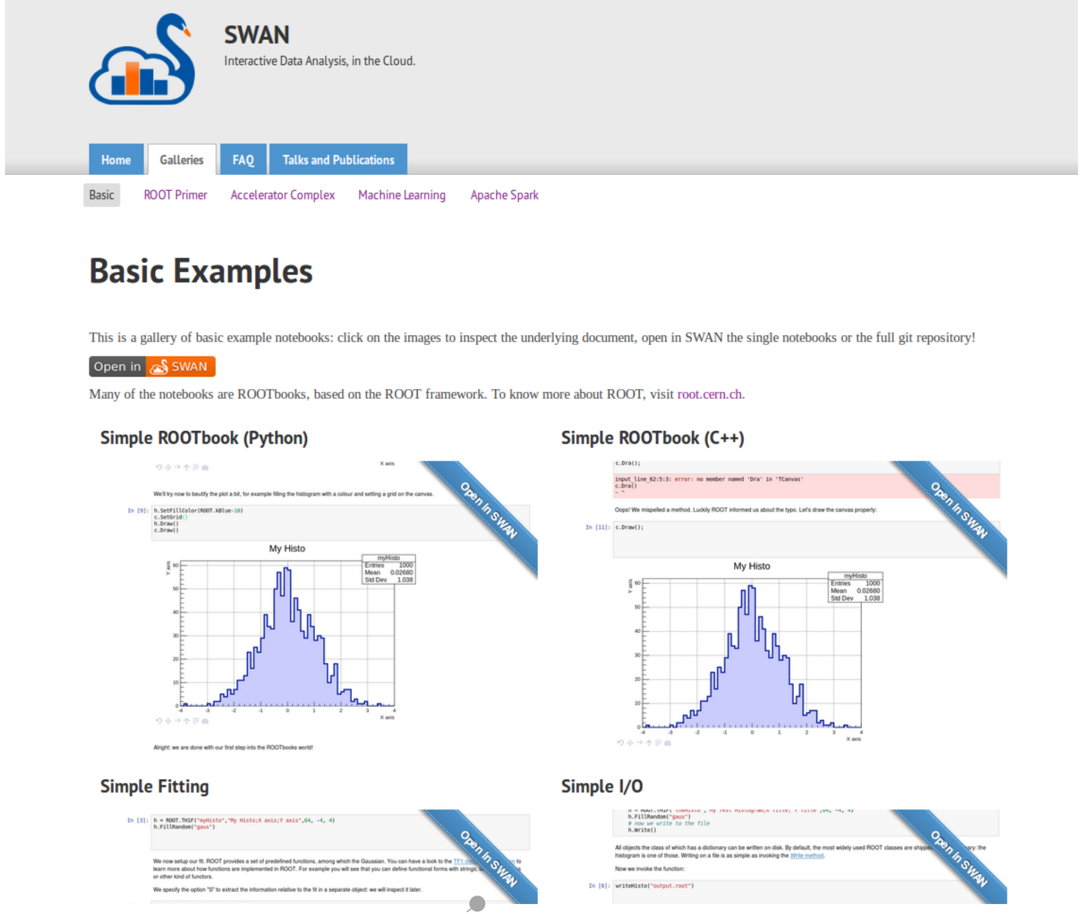
SWAN and the community

SWAN user community

> SWAN development is guided by our user community
  - New features (libs, kernels, ...) are requested by users from their real usage needs

> Gallery of examples
  - Made in collaboration with our users
  - Almost 50 notebooks in 7 categories

Example notebooks at **swan.web.cern.ch**



SWAN

Interactive Data Analysis, in the Cloud.

Basic Examples

This is a gallery of basic example notebooks: click on the images to inspect the underlying document, open in SWAN the single notebooks or the full git repository!

Many of the notebooks are ROOTbooks, based on the ROOT framework. To know more about ROOT, visit root.cern.ch.

Simple ROOTbook (Python)

We'll try now to beutify the plot a bit, for example filling the histogram with a colour and setting a grid on the canvas.

```
In [9]: h.SetFillColor(ROOT.kBlue-10)
        c.SetGrid()
        h.Draw()
        c.Draw()
```

Alright: we are done with our first step into the ROOTbooks world!

Simple ROOTbook (C++)

```
c.Dra();
input_line_62:5:3: error: no member named 'Dra' in 'TCanvas'
c.Dra()
~ ^
```

Oops! We mispelled a method. Luckily ROOT informed us about the typo. Let's draw the canvas properly:

```
In [11]: c.Draw();
```

Simple Fitting

```
In [3]: h = ROOT.TH1F("myHisto","My Histo;X axis;Y axis",64, -4, 4)
        h.FillRandom("gaus")
```

We now setup our fit. ROOT provides a set of predefined functions, among which the Gaussian. You can have a look to the TF1 cl[illegible] to learn more about how functions are implemented in ROOT. For example you will see that you can define functional forms with strings, [illegible] or other kind of functors.

We specify the option "S" to extract the information relative to the fit in a separate object: we will inspect it later.

Simple I/O

```
h.FillRandom("gaus")
# now we write to the file
h.Write()
```

All objects the class of which has a dictionary can be written on disk. By default, the most widely used ROOT classes are shipp[illegible]ary: the histogram is one of those. Writing on a file is as simple as invoking the Write method.

Now we invoke the function:

```
In [6]: writeHisto("output.root")
```

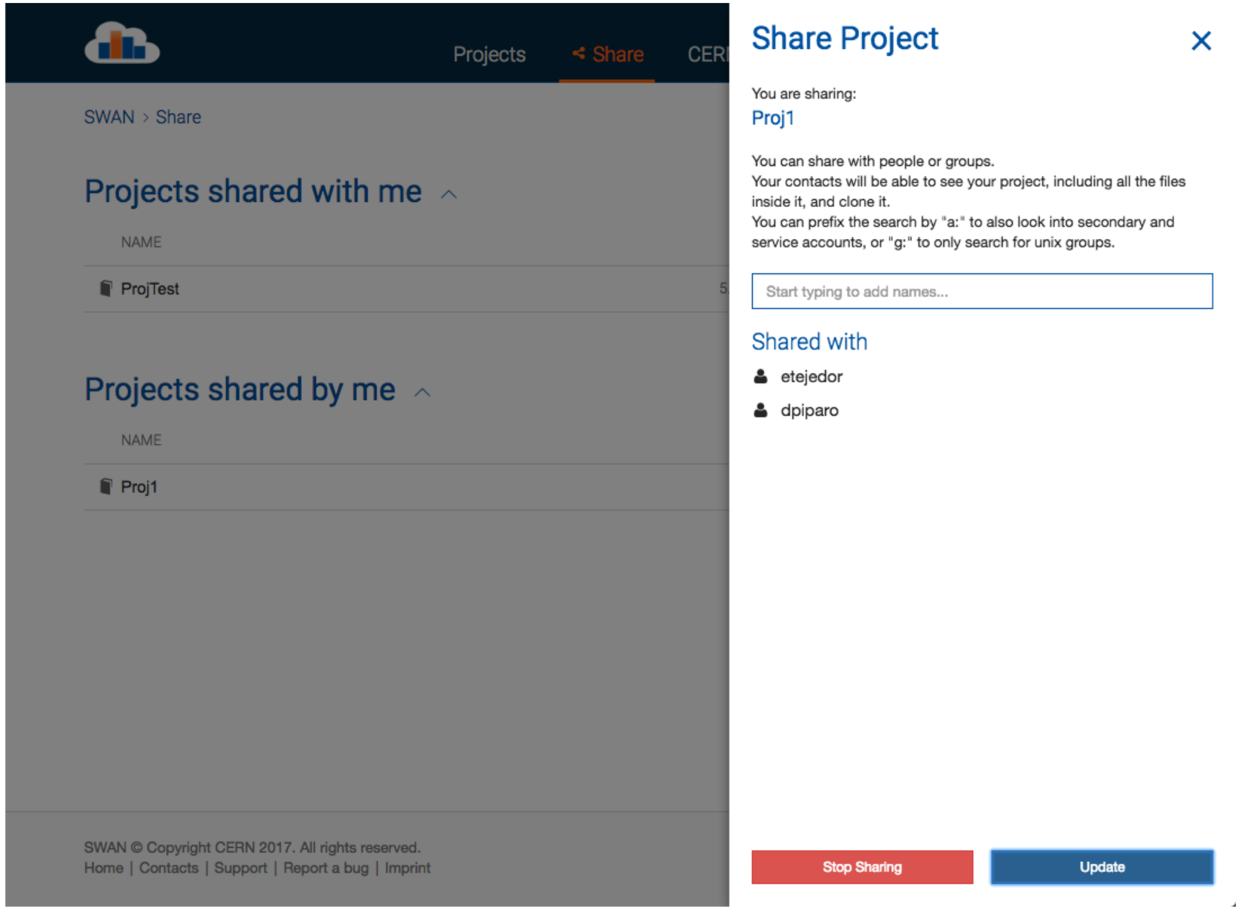
Access with only a click

Recent developments

Sharing made easy

> Sharing from inside SWAN interface

> Users can share “Projects”
  - Special kind of folder that contains notebooks and other files, i.e. input data



Projects shared with me

NAME
ProjTest

Projects shared by me

NAME
Proj1

SWAN © Copyright CERN 2017. All rights reserved.
Home | Contacts | Support | Report a bug | Imprint

Share Project

You are sharing:
Proj1

You can share with people or groups.
Your contacts will be able to see your project, including all the files inside it, and clone it.
You can prefix the search by "a:" to also look into secondary and service accounts, or "g:" to only search for unix groups.

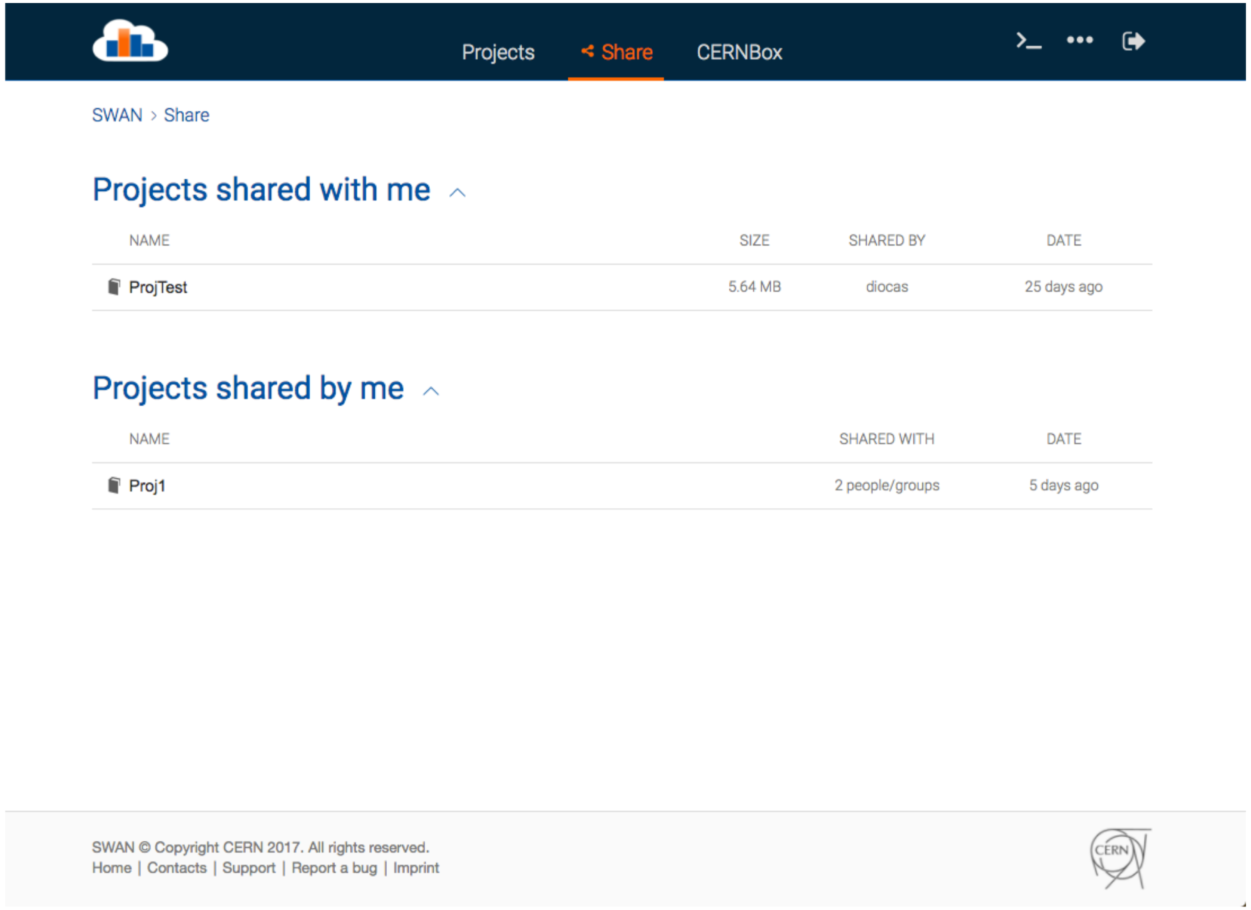
Start typing to add names...

Shared with
- etejedor
- dpiparo

Sharing made easy

> Users can open shared Projects
  - Inspect notebooks in view-mode

> Users can clone a shared Project
  - Jupyter doesn't allow concurrent editing



Projects shared with me

NAME	SIZE	SHARED BY	DATE
ProjTest	5.64 MB	diocas	25 days ago

Projects shared by me

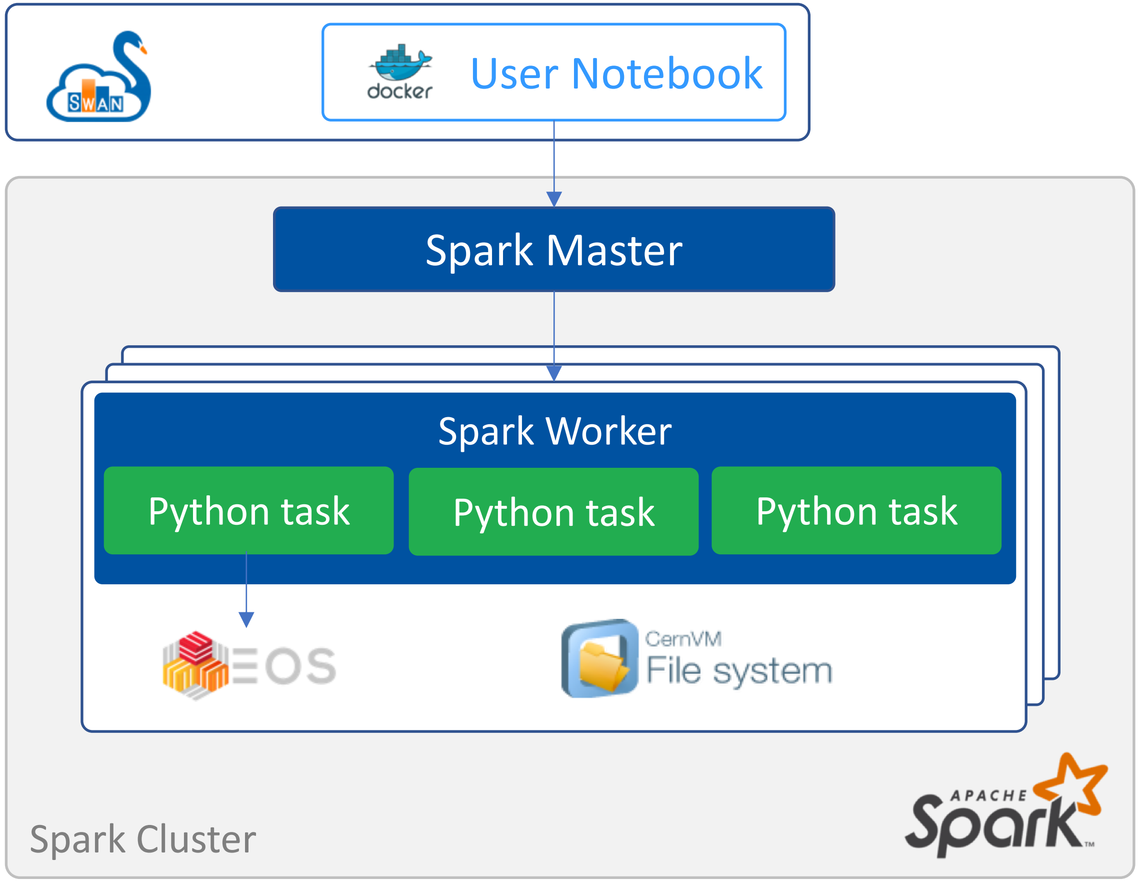
NAME	SHARED WITH	DATE
Proj1	2 people/groups	5 days ago

SWAN © Copyright CERN 2017. All rights reserved.
Home | Contacts | Support | Report a bug | Imprint

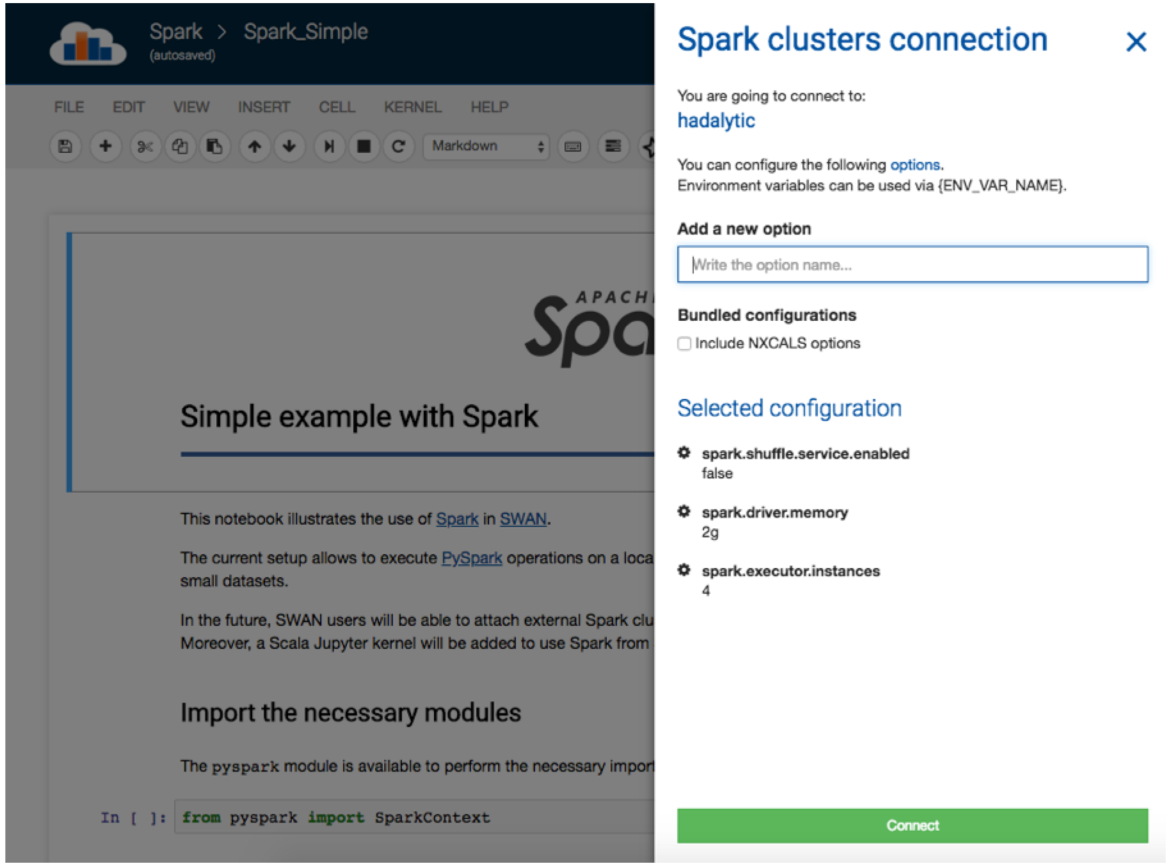
Integration with Spark

- One of the features requested by the community
  - Team from the Beams department

- Allow users to connect to CERN Spark Clusters to submit jobs



Integration with Spark



Simple example with Spark

This notebook illustrates the use of Spark in SWAN.

The current setup allows to execute PySpark operations on a loca
small datasets.

In the future, SWAN users will be able to attach external Spark clu
Moreover, a Scala Jupyter kernel will be added to use Spark from

Import the necessary modules

The pyspark module is available to perform the necessary impor

```
In [ ]: from pyspark import SparkContext
```

Spark clusters connection

You are going to connect to:
hadalytic

You can configure the following options.
Environment variables can be used via {ENV_VAR_NAME}.

Add a new option

Write the option name...

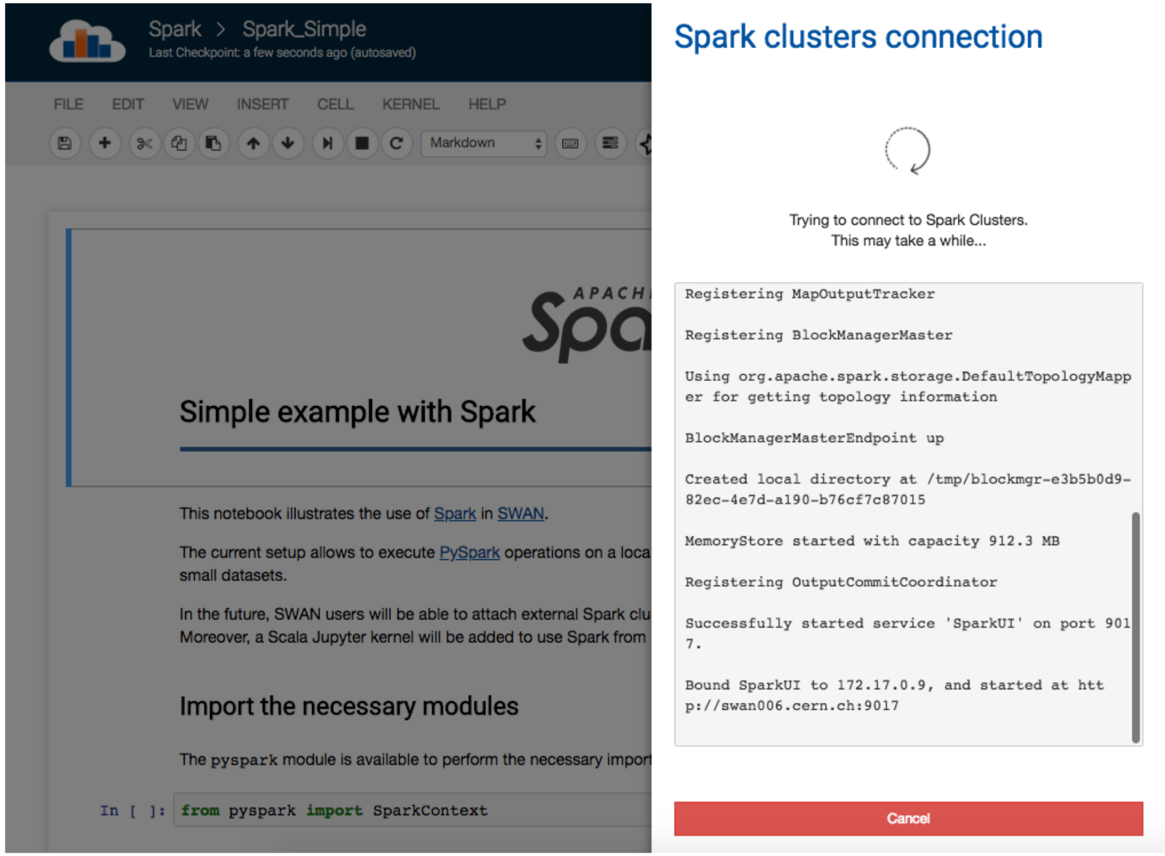
Bundled configurations

□ Include NXCALS options

Selected configuration

- **spark.shuffle.service.enabled**
  false
- **spark.driver.memory**
  2g
- **spark.executor.instances**
  4

Connect



Simple example with Spark

This notebook illustrates the use of Spark in SWAN.

The current setup allows to execute PySpark operations on a loca
small datasets.

In the future, SWAN users will be able to attach external Spark clu
Moreover, a Scala Jupyter kernel will be added to use Spark from

Import the necessary modules

The pyspark module is available to perform the necessary impor

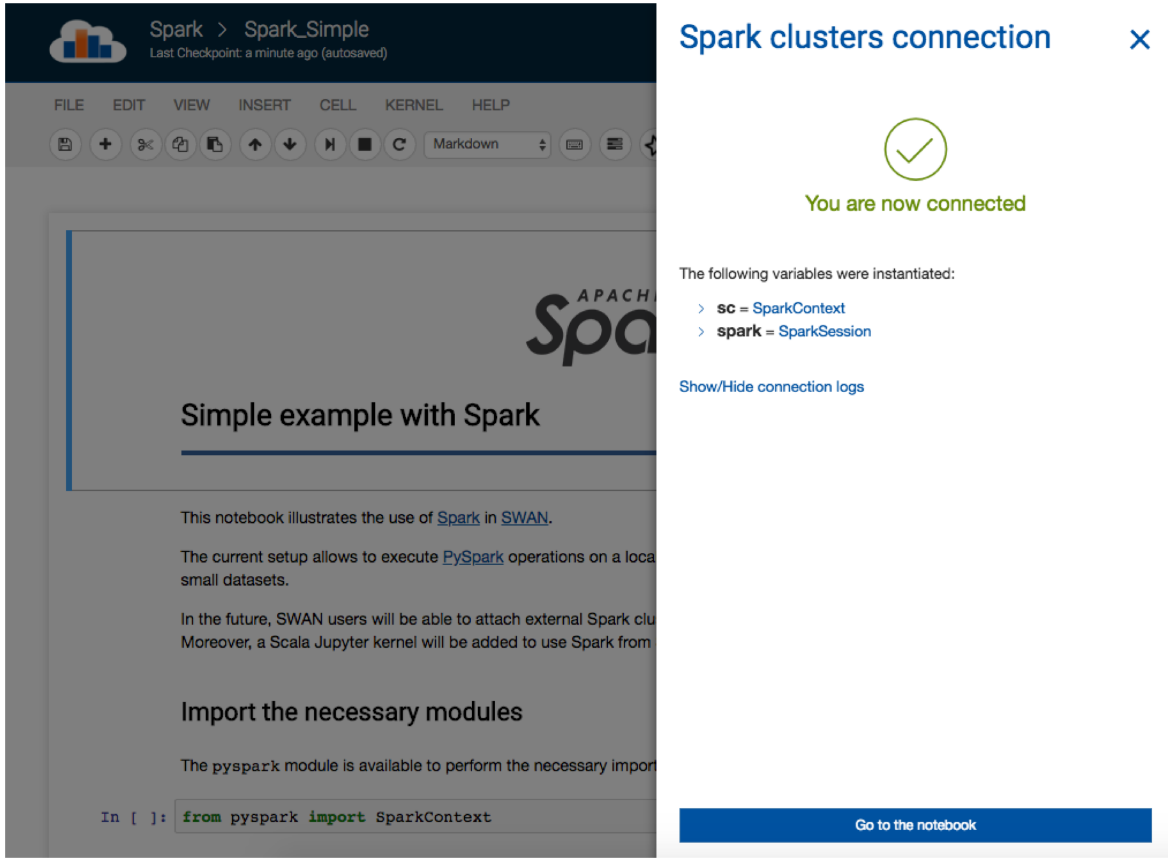
```
In [ ]: from pyspark import SparkContext
```

Spark clusters connection

Trying to connect to Spark Clusters.
This may take a while...

```
Registering MapOutputTracker

Registering BlockManagerMaster

Using org.apache.spark.storage.DefaultTopologyMapp
er for getting topology information

BlockManagerMasterEndpoint up

Created local directory at /tmp/blockmgr-e3b5b0d9-
82ec-4e7d-a190-b76cf7c87015

MemoryStore started with capacity 912.3 MB

Registering OutputCommitCoordinator

Successfully started service 'SparkUI' on port 901
7.

Bound SparkUI to 172.17.0.9, and started at htt
p://swan006.cern.ch:9017
```

Cancel



Simple example with Spark

This notebook illustrates the use of Spark in SWAN.

The current setup allows to execute PySpark operations on a loca
small datasets.

In the future, SWAN users will be able to attach external Spark clu
Moreover, a Scala Jupyter kernel will be added to use Spark from

Import the necessary modules

The pyspark module is available to perform the necessary impor

```
In [ ]: from pyspark import SparkContext
```

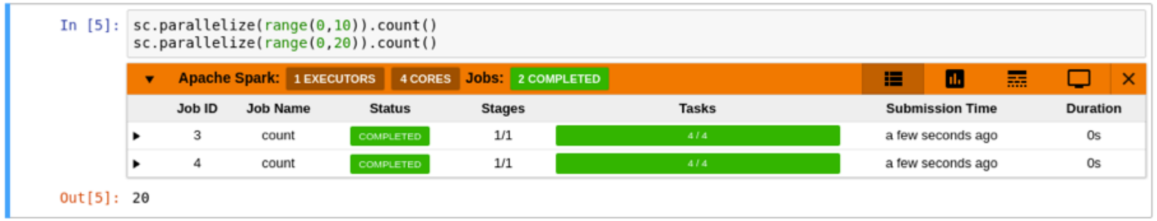
Spark clusters connection

You are now connected

The following variables were instantiated:

- sc = SparkContext
- spark = SparkSession

Show/Hide connection logs



```
In [5]: sc.parallelize(range(0,10)).count()
        sc.parallelize(range(0,20)).count()
```

	Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
▶	3	count	COMPLETED	1/1	4 / 4	a few seconds ago	0s
▶	4	count	COMPLETED	1/1	4 / 4	a few seconds ago	0s

```
Out[5]: 20
```

Demo

Video

Conclusion

Conclusion

- SWAN is a CERN service that provides Jupyter Notebooks on demand
- SWAN promotes a cloud based analysis model where users can do analysis only with their browser
- SWAN federates CERN services for software, storage and infrastructure so that users can find what they need in the service
- SWAN fosters collaboration and results sharing between scientists
- SWAN is an Interface for Mass Processing Resources (Spark)

SWAN: service for web based analysis

Thank you