APACHECON
EUROPE

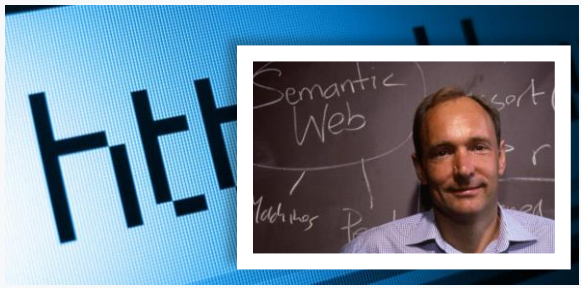
OPEN SOURCE BIG DATA TOOLS ACCELERATING PHYSICS RESEARCH AT CERN

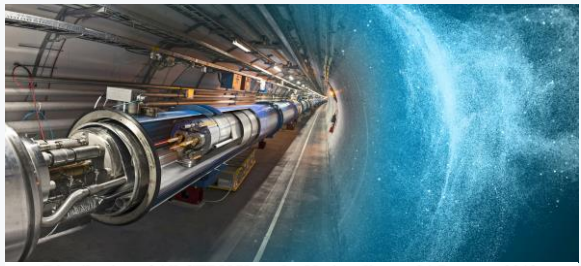
Prasanth Kothuri, CERN

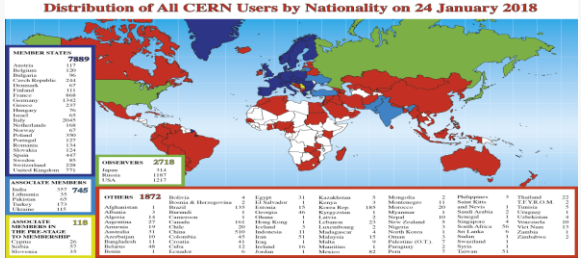
APACHECON
EUROPE Oct. 22nd - 24th
2019

CERN

- CERN - European **Laboratory** for Particle Physics
- The place where the **Web** was born
- Home of the **Large Hadron Collider** and 4 big Experiments:
  - ATLAS ~ CMS ~ LHCb ~ ALICE
- 22 member states + 6 associate members + **world-wide** collaborations
  - ~3000 Members of Personnel
  - ~12,000 users
  - ~1000 MCHF yearly budget







The Large Hadron Collider (LHC)

Largest machine in the world

27km, 6000+ superconducting magnets

Fastest racetrack on Earth

Protons circulate 11245 times/s (99.9999991% the speed of light)

Emptiest place in the solar system

High vacuum inside the magnets

Hottest spot in the galaxy

During Lead ion collisions create temperatures 100 000x hotter than the heart of the sun;

CERN Data Centre

- Physics data are aggregated in the CERN Data Centre, where initial data reconstruction of physics events is performed
- A remote extension of the CERN data centre is hosted in Budapest, Hungary. It provides the extra computing power required to cover CERN's needs.

~ **12.5 PB per month**

~ **2 PB accessed every day**

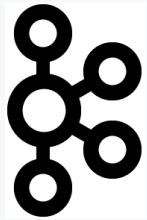
~ 4 megawatt of electricity

	Meyrin Data Centre	Wigner Extension	TOTAL
Servers	11 500	3 500	15 000
Processor cores	174 300	56 000	230 300
Disks	61 900	29 700	91 600 (280 PB capacity)
Tape Cartridges			32 200 (~ 400 PB capacity)

Data at Scale @ CERN

- **Physics data** – Today we use WLCG to handle it
  - Optimised for physics analysis and concurrent access
  - ROOT framework - custom software and data format
  - Early stage experimental work ongoing to use Spark for physics analysis

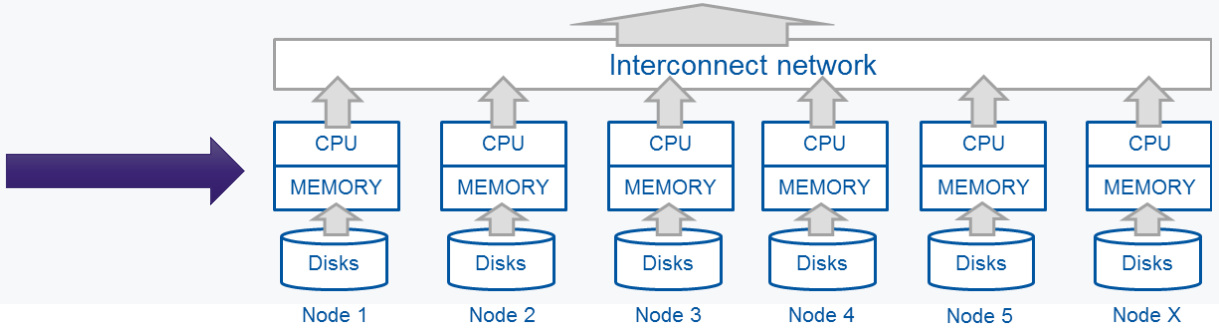
- **Infrastructure data**
  - Accelerators and detector controllers
  - Experiments Data catalogues (collisions, files etc.)
  - Monitoring of the WLCG and CERN data centres
  - Systems logs



Hadoop and Spark for big data analytics

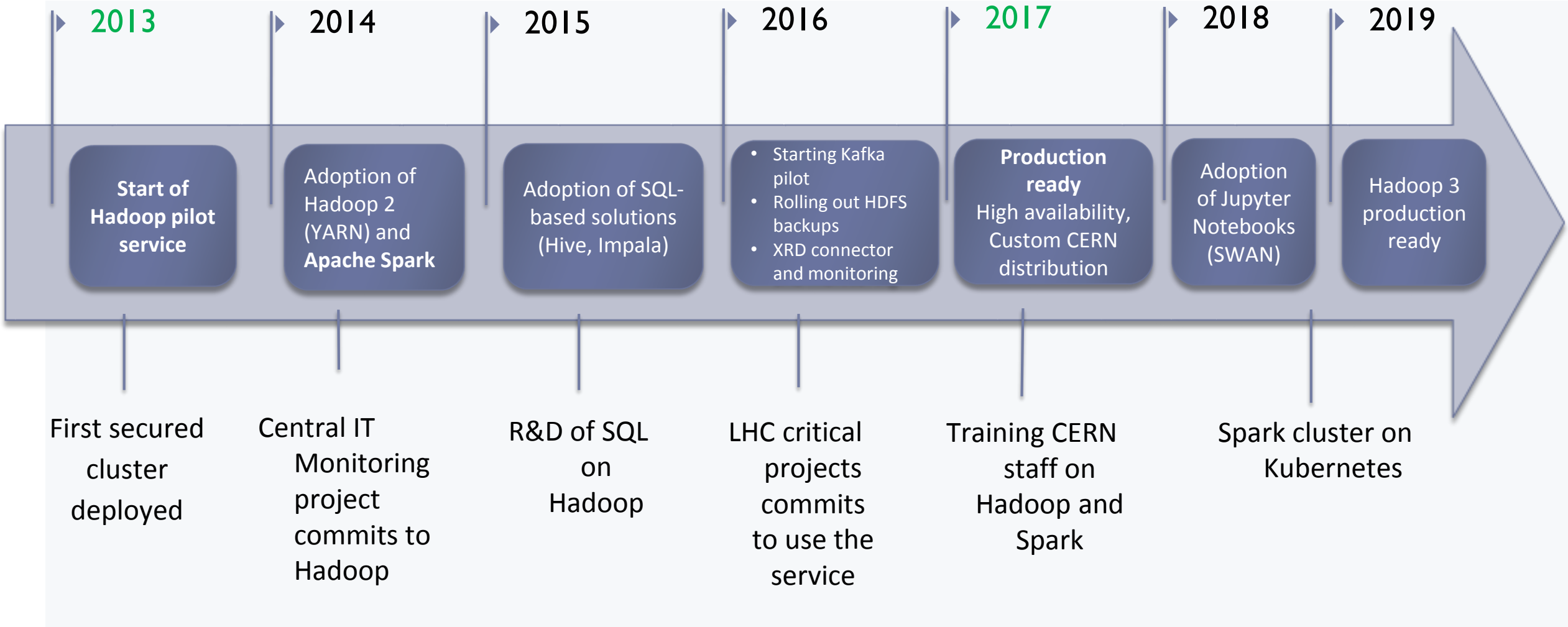
- Distributed systems for data processing
  - Storage and multiple data processing interfaces
  - Can operate at scale by design (shared nothing)
  - Typically on clusters of commodity-type servers/cloud
  - Many solutions target data analytics and data warehousing
  - Can do much more: stream processing, machine learning
- Already well established in the industry and open source

Scale-out data processing



APACHECON EUROPE

CERN Hadoop and Spark Service - Timeline



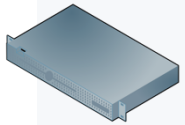
Hadoop and Spark Service at CERN IT

- Setup and run the infrastructure
- Support user community
  - Provide consultancy
  - Train on the technologies

- Facilitate use
  - Notebook service
  - Docker clients
  - Package libraries and configuration

Hadoop and Spark service in numbers

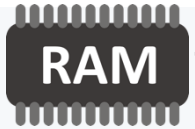
- ✧6 clusters
  - ✧4 production (bare-metal)
  - ✧2 QA clusters (VMs)



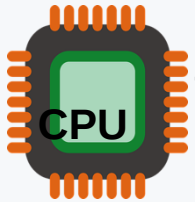
- ✧65 physical servers

- ✧40+ virtual machines

- ✧18+ PBs of Storage

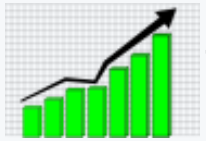


- ✧20+ TB of Memory



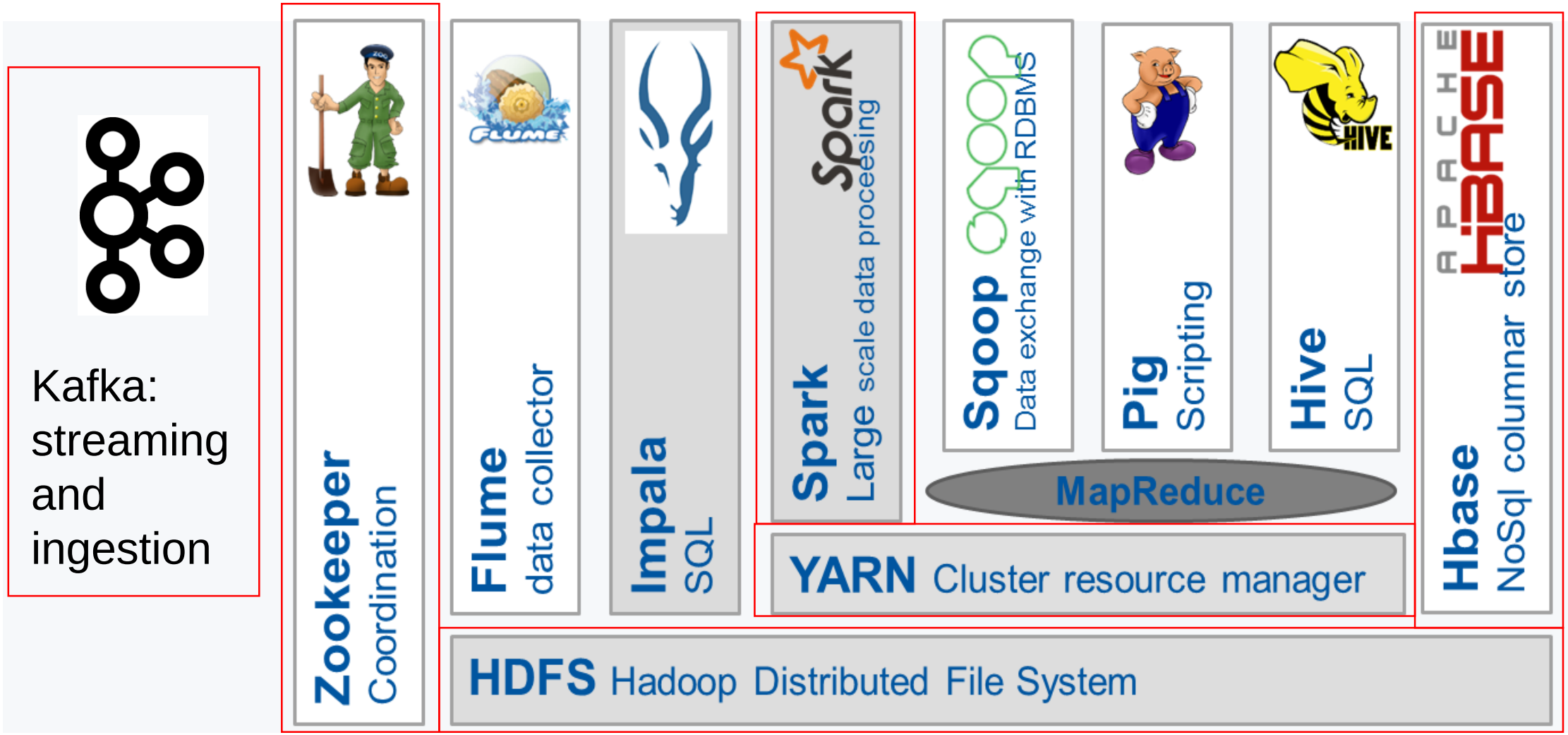
- ✧1500+ physical cores

- ✧HDDs and SSDs



- ✧Data growth: 4 TB per day

Overview of Hadoop components used



Hadoop and Spark production deployment

- Software distribution
  - Vanilla Apache (since 2017)
  - Cloudera

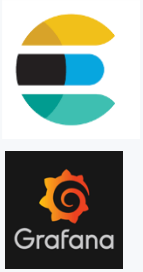
- Rolling change deployment
  - no service downtime
  - transparent in most of the cases

- Installation and configuration
  - CentOS 7.6
  - custom Puppet module

- Host monitoring and alerting
  - via CERN IT Monitoring infrastructure

- Security
  - authentication Kerberos
  - fine-grained authorization integrated ldap/e-groups

- Service level monitoring
  - metrics integrated with: Elastic + Grafana
  - custom scripts for availability and alerting



- High availability
  - automatic master failover
  - for HDFS, YARN and HBASE

- HDFS backups
  - Daily incremental snapshots
  - Sent to tapes (CASTOR)

Moving to Apache Hadoop distribution

- Better control of the core software stack
  - Independent from a vendor/distributor
  - In-house compilation
  - Enabling non-default features (compression algorithms, R for Spark)
  - Adding critical patches (that are not ported in upstream)
- We do rpm packaging for core components
  - HDFS and YARN, Spark, HBase
- Streamlined development
  - Available on Maven Central Repository

SWAN – Jupyter Notebooks On Demand

- Service for web based analysis (SWAN)
  - Developed at CERN, based on Jupyter notebooks
  - Tightly integrated with CERN resources; software, storage and mass compute
- An interactive platform that combines code, equations, text and visualizations
  - Ideal for exploration, reproducibility, collaboration
- Fully integrated with Spark and Hadoop at CERN
  - Python on Spark (PySpark) at scale
  - Modern, powerful and scalable platform for data analysis
  - Web-based: no need to install any software

Git repo: https://github.com/swan-cern

FILE EDIT VIEW INSERT CELL KERNEL HELP Trusted Python 2

Code

Do the heavylifting in spark and collect aggregated view to panda DF

In [11]:
```
df_loadAvg_pandas = spark.sql("SELECT submitter_host, \
                                      avg(body.LoadAvg) as avg, \
                                      hour(from_unixtime(timestamp / 1000, 'yyyy-MM-dd HH:mm:ss')) as hr \
                                FROM loadAvg \
                                WHERE submitter_hostgroup = 'hadoop/itdb/datanode' \
                                AND dayofmonth(from_unixtime(timestamp / 1000, 'yyyy-MM-dd HH:mm:ss')) = 15 \
                                GROUP BY hour(from_unixtime(timestamp / 1000, 'yyyy-MM-dd HH:mm:ss')), submitter_host")\
                    .toPandas()
```

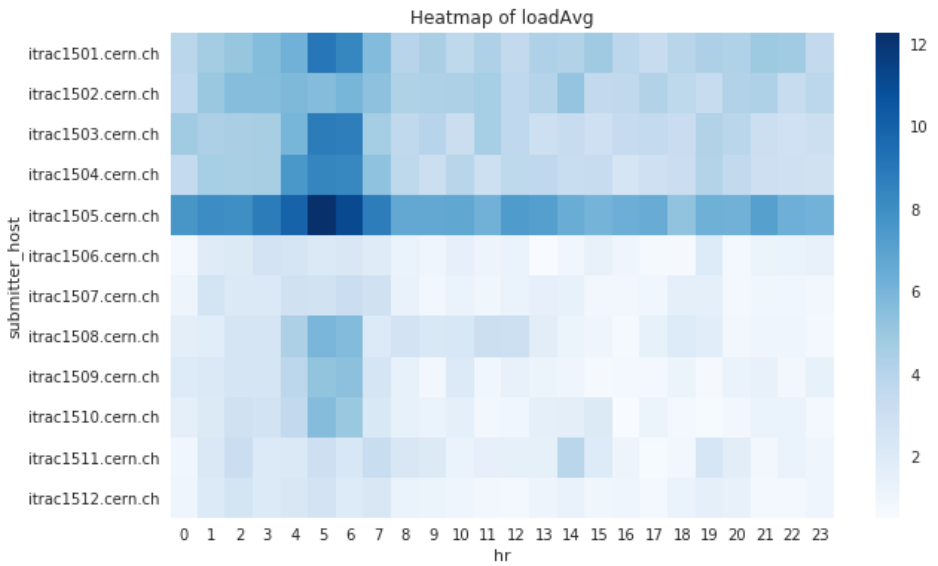
Apache Spark: 90 EXECUTORS 180 CORES Jobs: 1 COMPLETED

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
3	toPandas	COMPLETED	2/2	388 / 388	4 minutes ago	36s

Visualize with seaborn

In [19]:
```
# heatmap of service availability
plt.figure(figsize=(10, 6))
ax = sns.heatmap(df_loadAvg_pandas.pivot(index='submitter_host', columns='hr', values='avg'), cmap="Blues")
ax.set_title("Heatmap of loadAvg")
```

Out[19]: Text(0.5,1,u'Heatmap of loadAvg')



Text

Code

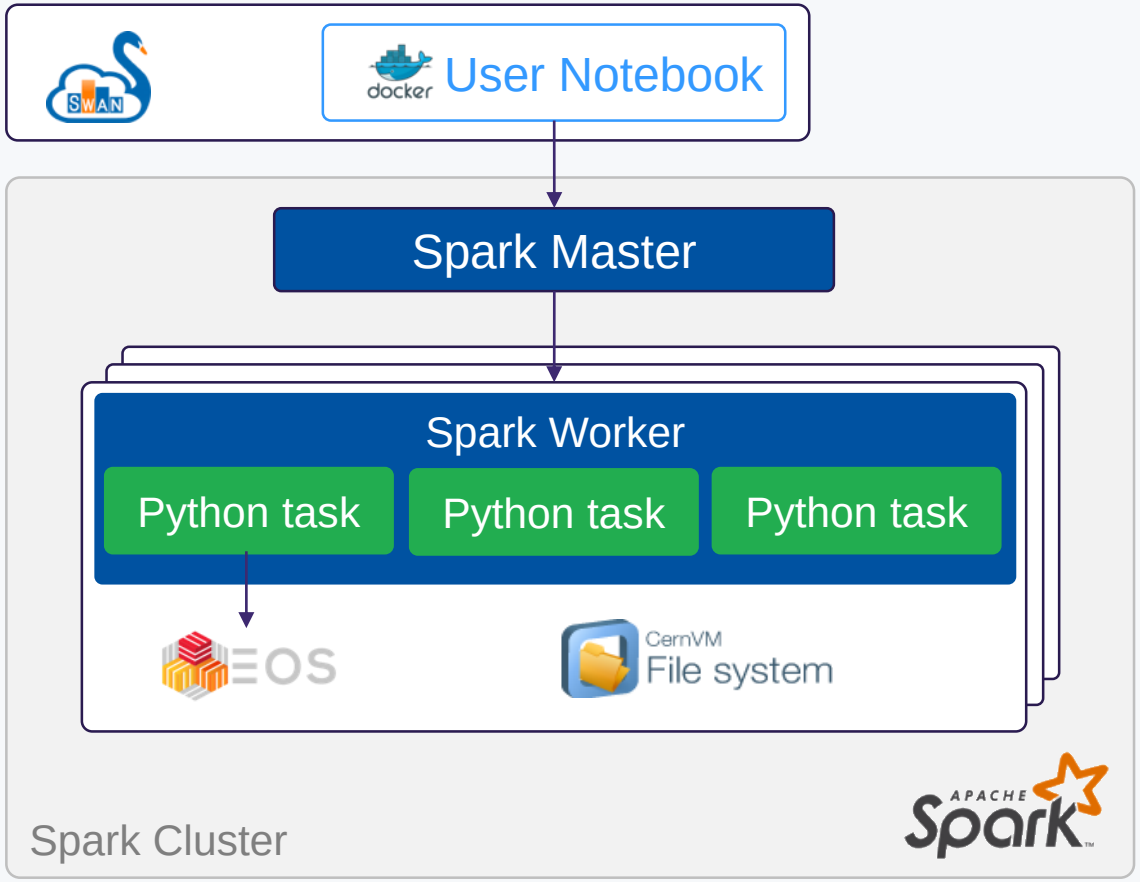
Monitoring

Visualizations

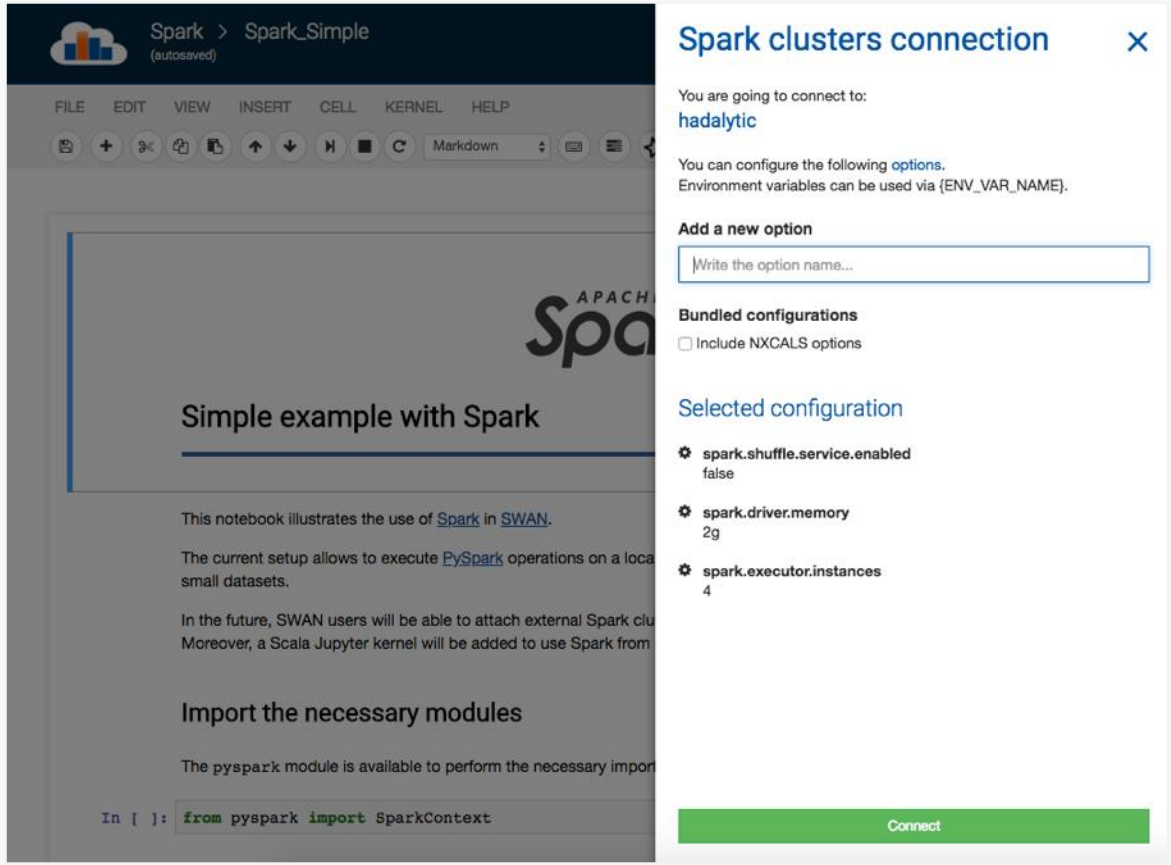
APACHECON EUROPE

SWAN – Integration with Spark

- Connection to CERN Spark Clusters
  - Spark: general purpose distributed computing framework
- Same environment across platforms (local/remote)
  - User data - EOS
  - Software - CVMFS
- Graphical Jupyter extensions developed
  - Spark Connector
  - Spark Monitor
- Not only used for Physics Analysis at CERN
- Spark Clusters
  - NXCals – Dedicated cluster for accelerator logging
  - Analytix – General purpose YARN cluster
  - Cloud Containers – General purpose Kubernetes cluster



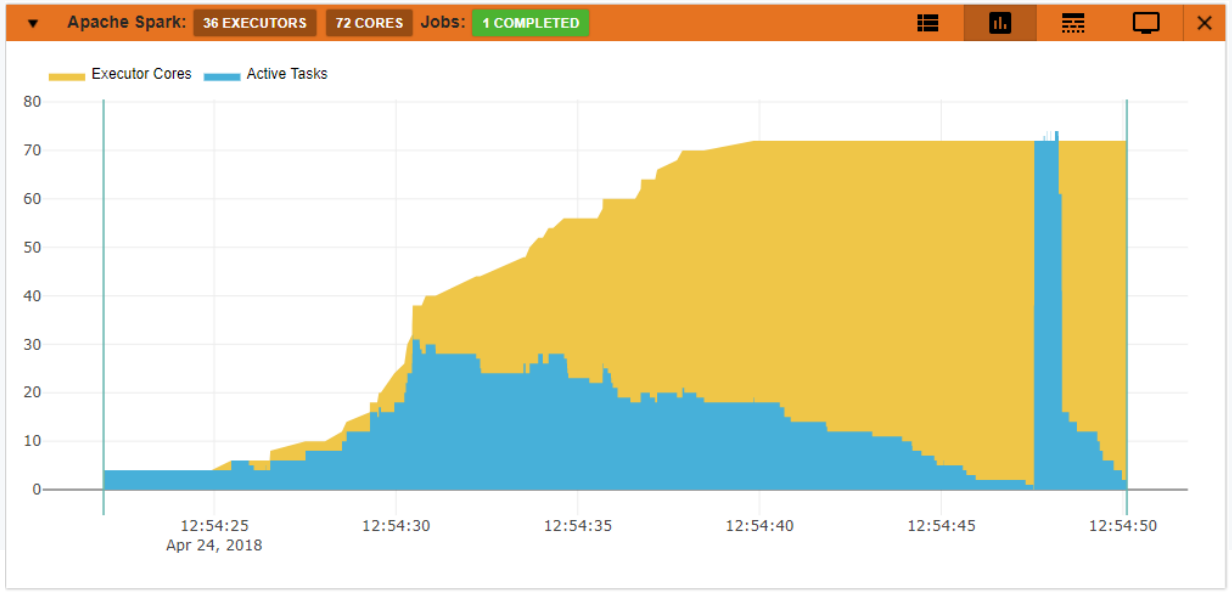
SWAN – Spark Connector



- Spark Connector – handling the spark configuration complexity
  - User is presented with Spark Session (Spark) and Spark Context (sc)
  - Ability to bundle configurations specific to user communities
  - Ability to specify additional configuration

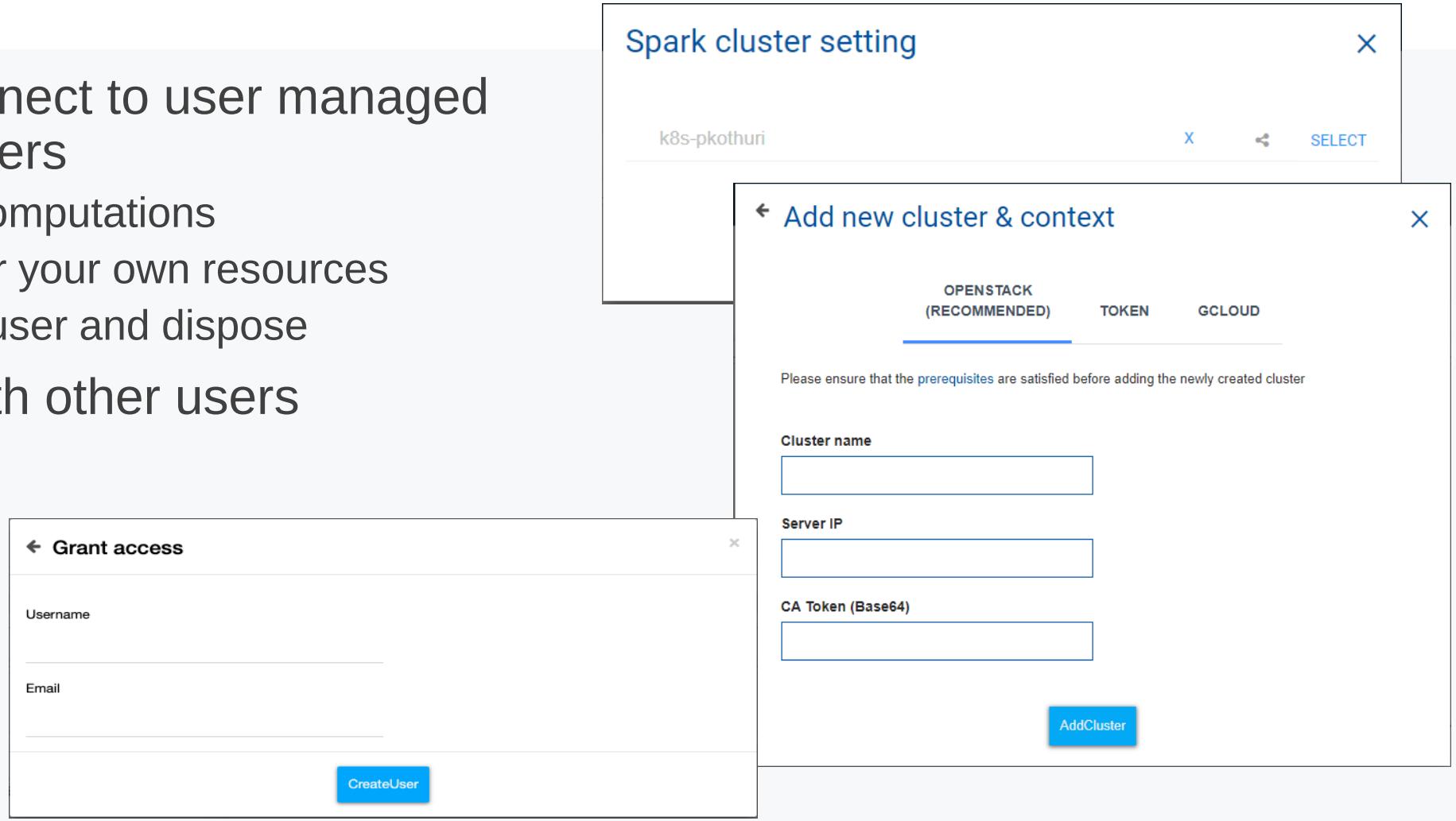
SWAN – Spark Monitor

- Spark Monitor – jupyter notebook extension
  - For live monitoring of spark jobs spawned from the notebook
  - Access to Spark WEB UI from the notebook



SWAN – Ephemeral Spark K8s clusters

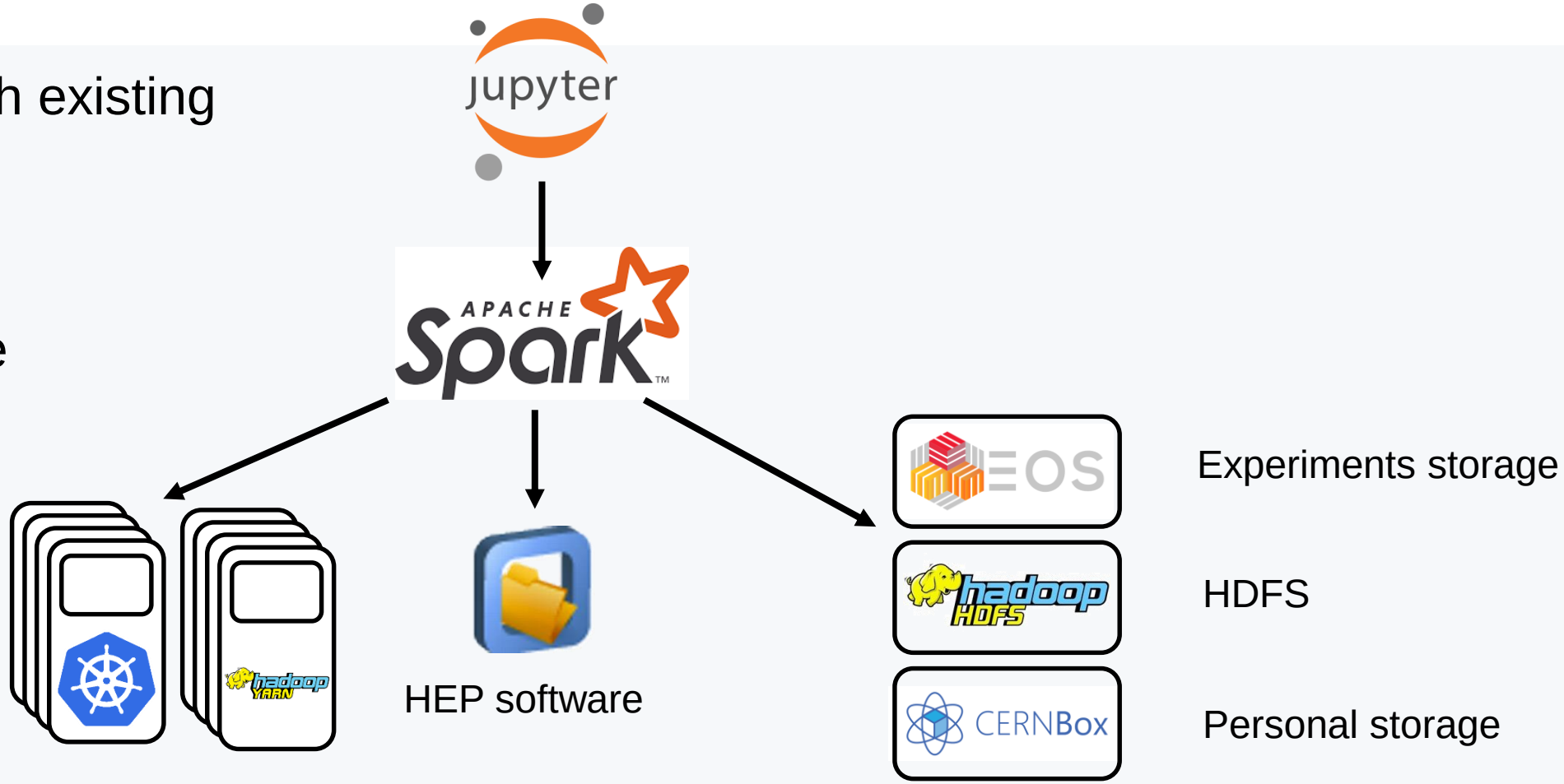
- Possibility to connect to user managed Kubernetes clusters
  - Offload Spark computations
  - Control and user your own resources
  - Quickly create, user and dispose
- Share access with other users



Analytics Platform Outlook

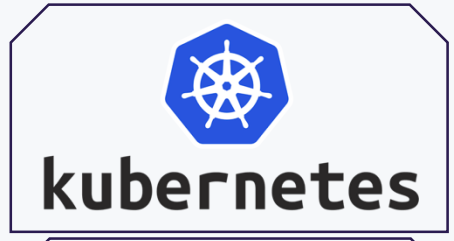
Integrating with existing infrastructure:

- Software
- Data
- Compute



Spark as a service on CERN private cloud

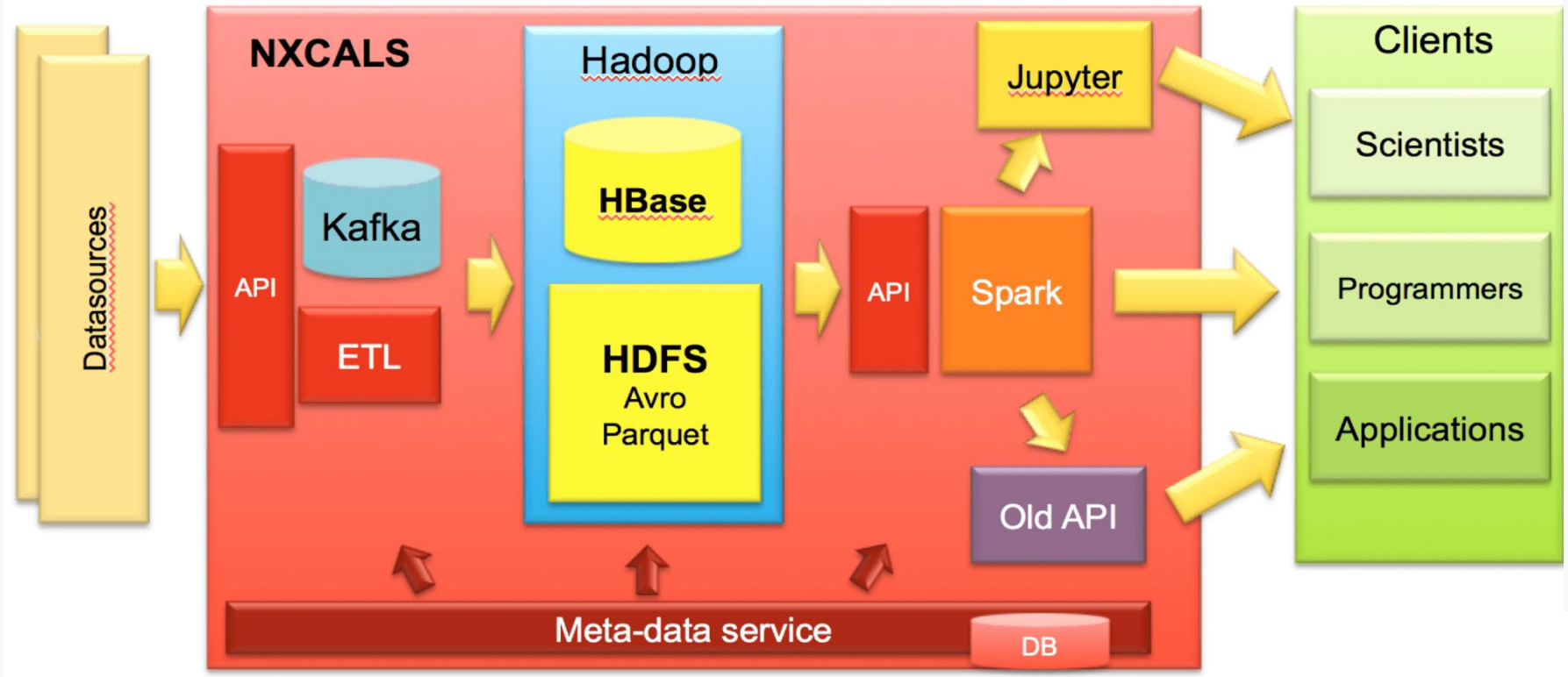
- Appears to be a good solution when data locality is not needed
  - CPU and memory intensive rather than IO intensive workloads
  - Reading from external storages (CERN Disk Storage - EOS, foreign HDFS)
  - Elasticity of compute resources
- Spark clusters - on containers
  - Kubernetes over Openstack
  - Leveraging the Kubernetes support in Spark 2.3
- Use cases
  - Ad-hoc users with high demand computing resource demanding workloads
  - Streaming jobs (e.g. accessing Apache Kafka)



Selected “Big Data” Projects at CERN

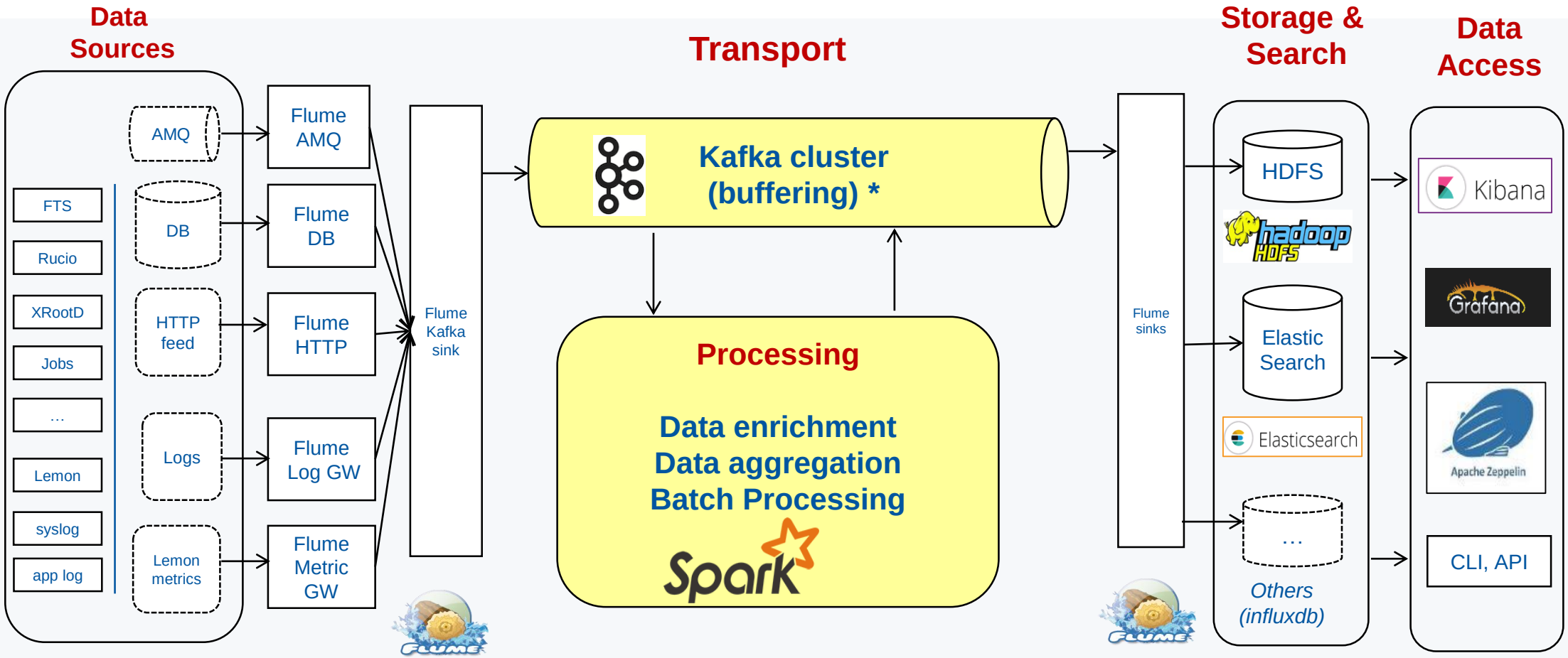
Next Gen. CERN Accelerator Logging

- A control system with: Streaming, Online System, API for Data Extraction
- Critical system for running LHC - 700 TB today, growing 200 TB/year
- Challenge: service level for critical production



New CERN IT Monitoring Infrastructure

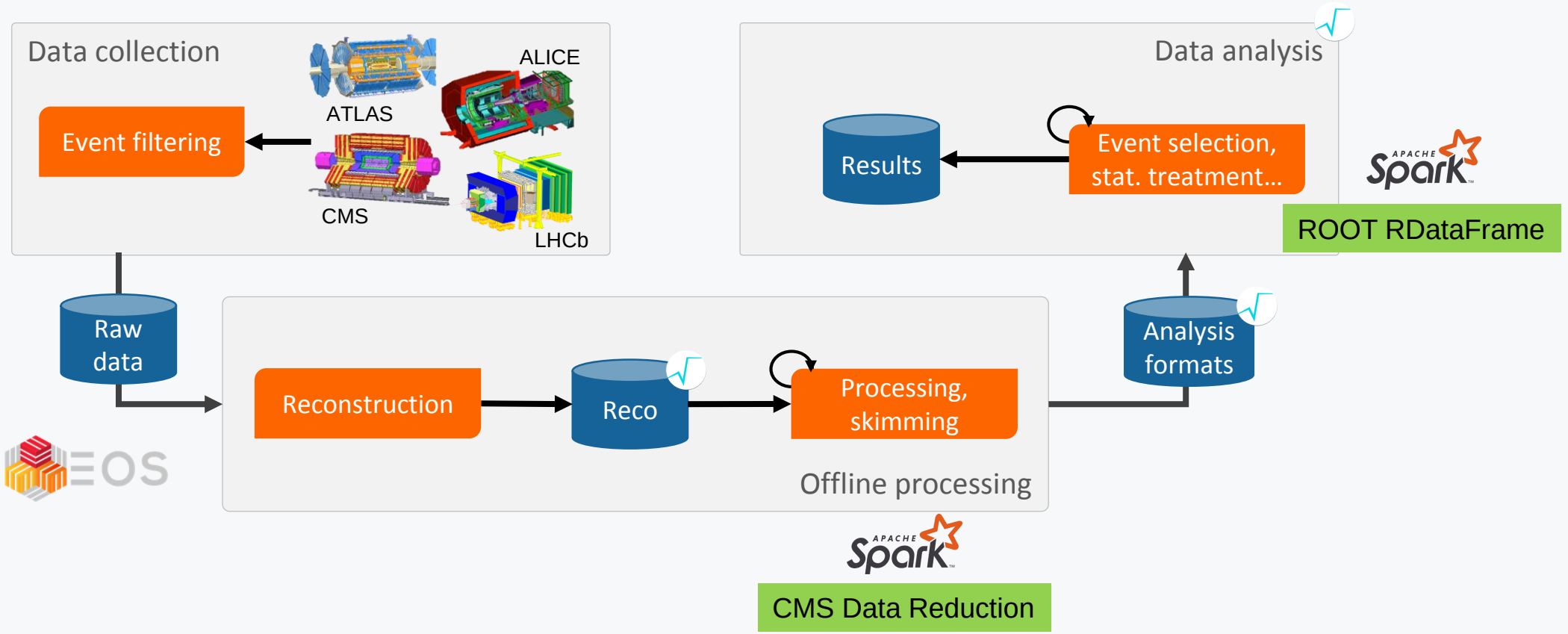
Critical for CC operations and WLCG



- Data now 200 GB/day, 200M events/day
- At scale 500 GB/day
- Proved to be effective in several occasions

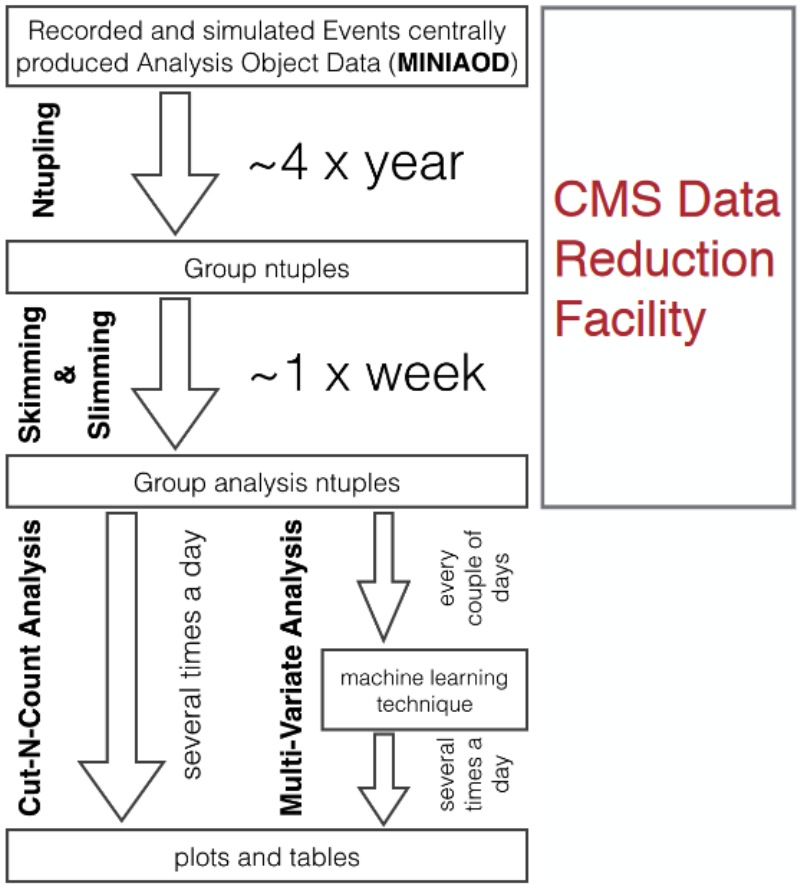
Spark for Physics Data Analysis

LHC Data Pipeline at CERN



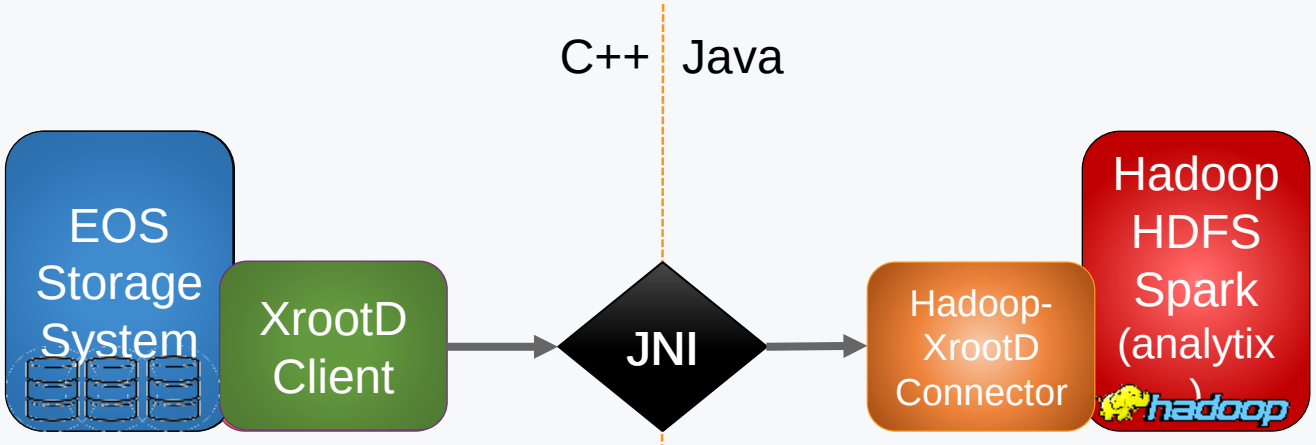
CMS Data Reduction Facility

- R&D, CMS Bigdata project, CERN openlab, Intel:
  - Reduce time to physics for PB-sized datasets
  - Exploring a possible new way to do HEP analysis Improve computing resource utilization
  - Enable physicists to use tools and methods from “Big Data” and open source communities
- CMS Data Reduction Facility:
  - Goal: produce reduced data n-tuples for analysis in a more agile way than current methods
  - Full reduction of 1PB physics data was completed; still in discussion on future direction



XRootD connector for Hadoop and Spark

- A library that binds Hadoop-based file system API with XRootD native client
  - Developed by CERN IT
- Allows most of components from Hadoop stack (Spark, MapReduce, Hive etc) to read/write directly from CERN storages – EOS and CASTOR
- Also developed Spark data source to load ROOT files into DataFrames



Git repo. - https://github.com/cerndb/hadoop-xrootd

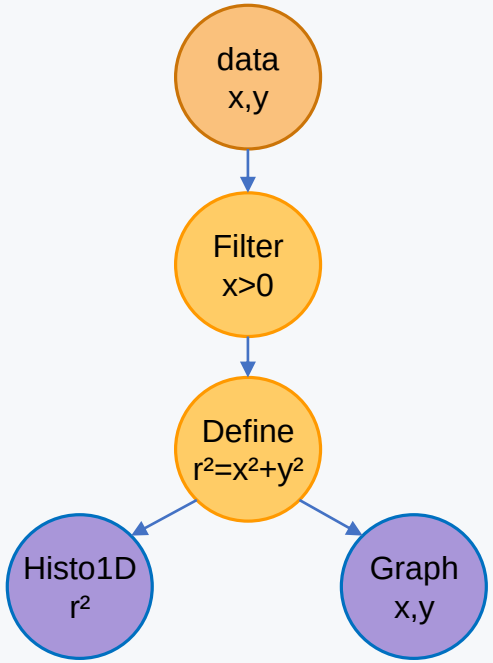
Git repo. - https://github.com/diana-hep/spark-root

ROOT RDataFrame

- RDataFrame
  - ROOT is tailored data analysis framework for HEP.
  - Implemented in C++, interfaced also to Python
  - Declarative API to build chain of DAG operations to process ROOT files

```
df = RDataFrame(dataset)

df2 = df.Filter('x > 0')
        .Define('r2', 'x*x + y*y')

h = df2.Histo1D('r2')
g = df2.Graph('x', 'y')
```

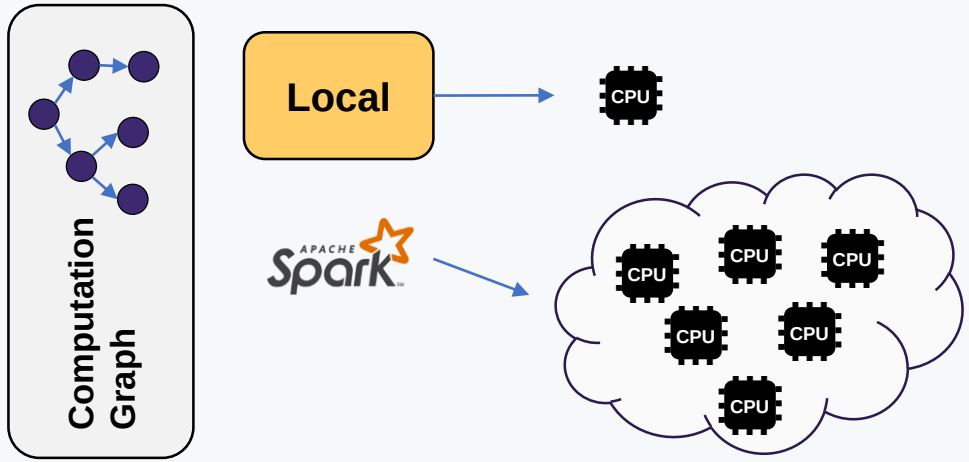
https://root.cern.ch



Distributed RDataFrame (experimental)

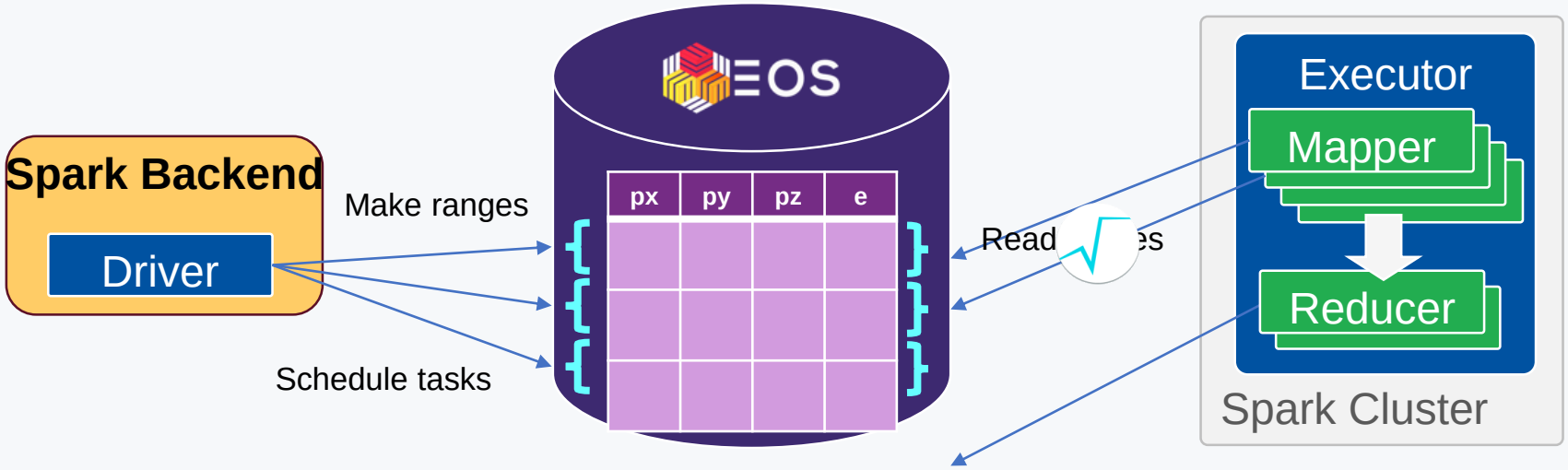
- Distributed RDataFrame
  - Parallelize RDataFrame computations with multiple backends
  - Allows to offload computations to spark clusters (Kubernetes or YARN) of possibly 1000s of vCPU

```
df = RDataFrame(dataset)

df2 = df.Filter('x > 0')
        .Define('r2', 'x*x + y*y')

h = df2.Histo1D('r2')
g = df2.Graph('x', 'y')
```



Distributed RDataFrame (experimental)

- Distributed RDataFrame
  - Map-reduce workflow where every mapper runs the RDataFrame computation graph on a range of collision events
  - Distributed computation is transparent to the user



Published at EuroPar 2019:
Declarative Big Data Analysis for High-Energy Physics: TOTEM Use Case

Machine Learning

Input: labeled data and DL models

Feature engineering at scale

Hyperparameter optimization (Random/Grid search)

Distributed model training

Output: particle selector model

Machine Learning Pipelines with Apache Spark and Intel BigDL:

https://databricks.com/session/deep-learning-on-apache-spark-at-cerns-large-hadron-collider-with-intel-technologies

Conclusions

- Demand of “Big Data” platforms and tools is growing at CERN
  - Many projects in-production
  - Projects around Monitoring, Security, Accelerators logging/controls, physics data, streaming…
- Hadoop and Spark services at CERN IT
  - Service is evolving: High availability, security, backups, external data sources, notebooks, short-lived disposable clusters
- Experience and community
  - Technologies evolve rapidly and knowledge sharing very important
  - We are happy to share/exchange our experience, tools, software with others
  - Our blog - http://db-blog.web.cern.ch/

APACHECON Europe, Oct 24th, 2019

OPEN SOURCE BIG DATA TOOLS ACCELERATING PHYSICS RESEARCH AT CERN

Prasanth Kothuri – prasanth.Kothuri@cern.ch