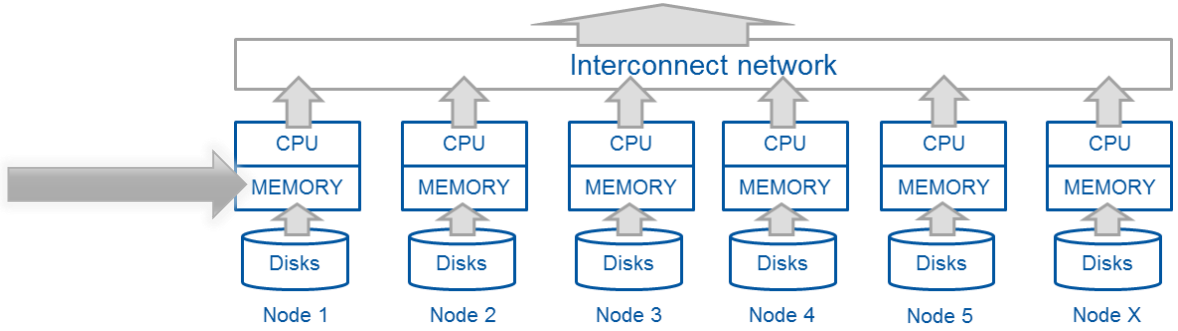
Hadoop and Spark services at CERN

CERN IT-DB Hadoop, Kafka and Spark Service
June 17th, 2019

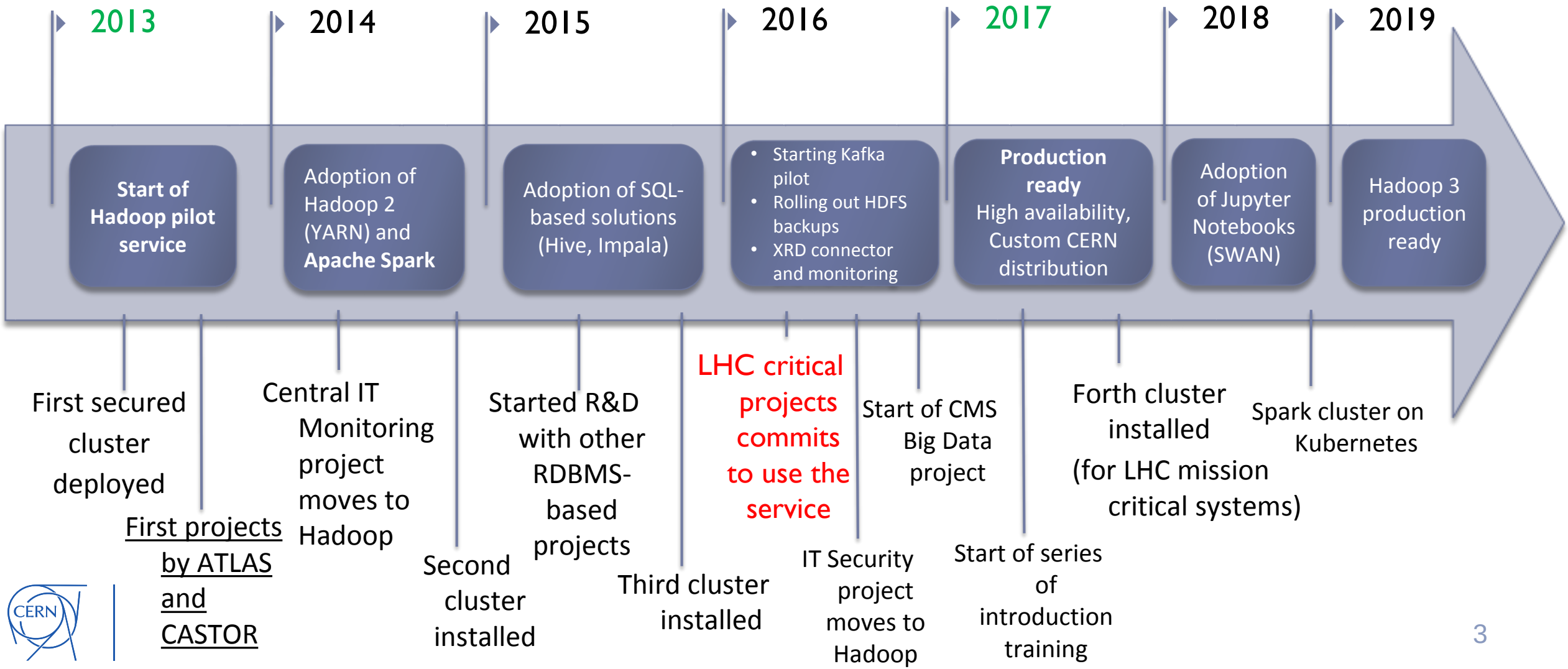
Hadoop and Spark for big data analytics

- 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



CERN Hadoop Service - Timeline



Hadoop Service at CERN IT

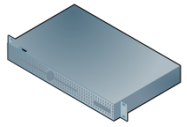
- Setup and run the infrastructure
- Support user community
  - Provide consultancy
  - Train on the technologies

- Facilitate use
  - Package libraries and configuration
  - Docker clients
  - Notebook service
  - https://hadoop-user-guide.web.cern.ch

Hadoop service in numbers

6 clusters

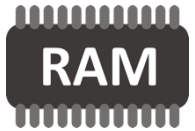
- 4 production (bare-metal)
- 2 QA clusters (VMs)



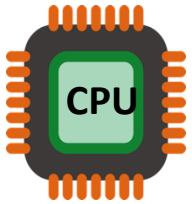
65 physical servers

40+ virtual machines

18+ PBs of Storage

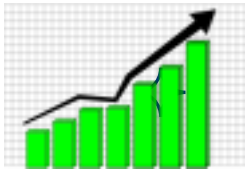


20+ TB of Memory



1500+ physical cores

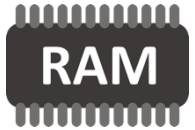
HDDs and SSDs



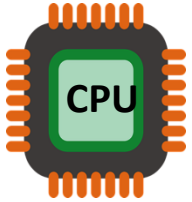
Data growth: 4 TB per day

Analytix (General Purpose) Cluster

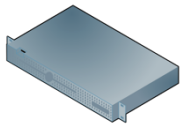
Bare-Metal



13+TB of Memory



800+ physical cores



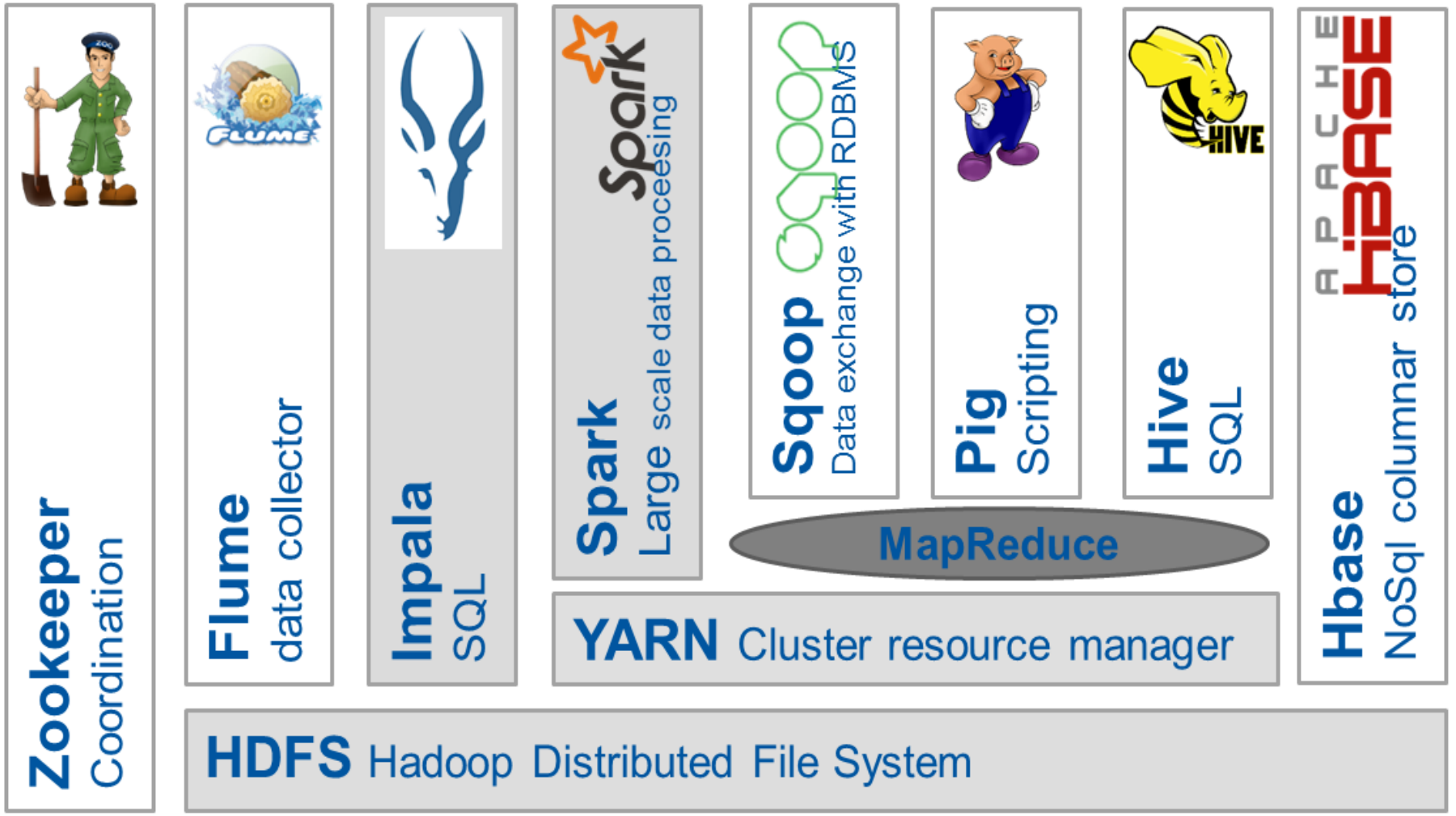
50+ Servers

HDDs and SSDs

10+PB of Storage

Overview of Available Components used

Kafka:
streaming
and ingestion



Apache Spark, Swiss Army Knife of Big Data

One tool, many uses

- Data extraction and manipulation
- Large ecosystem of data sources
- Engine for distributed computing
- Runs SQL
- Streaming
- Machine Learning
- GraphX
- ..

Image Credit: Vector Pocket Knife from Clipart.me

Hadoop and Spark production deployment

- Software distribution
  - Cloudera (since 2013)
  - Vanilla Apache (since 2017)

- Installation and configuration
  - CentOS 7.4
  - custom Puppet module

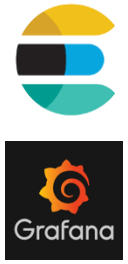
- Security
  - authentication Kerberos
  - fine-grained authorization integrated with ldap/e-groups (since 2018)

- High availability (since 2017)
  - automatic master failover
  - for HDFS, YARN and HBASE

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

- Host monitoring and alerting
  - via CERN IT Monitoring infrastructure

- Service level monitoring (since 2017)
  - metrics integrated with: Elastic + Grafana
  - custom scripts for availability and alerting



- HDFS backups (since 2016)
  - Daily incremental snapshots
  - Sent to tapes (CASTOR)

CASTOR
CERN Advanced STORage manager

CERN

Moving to Apache Hadoop distribution (since 2017)

- 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, initially for physics analysis by EP-SFT
- An interactive platform that combines code, equations, text and visualizations
  - Ideal for exploration, reproducibility, collaboration

- Fully integrated with Spark and Hadoop at CERN (2018)
  - Python on Spark (PySpark) at scale
  - Modern, powerful and scalable platform for data analysis
  - Web-based: no need to install any software

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()
```

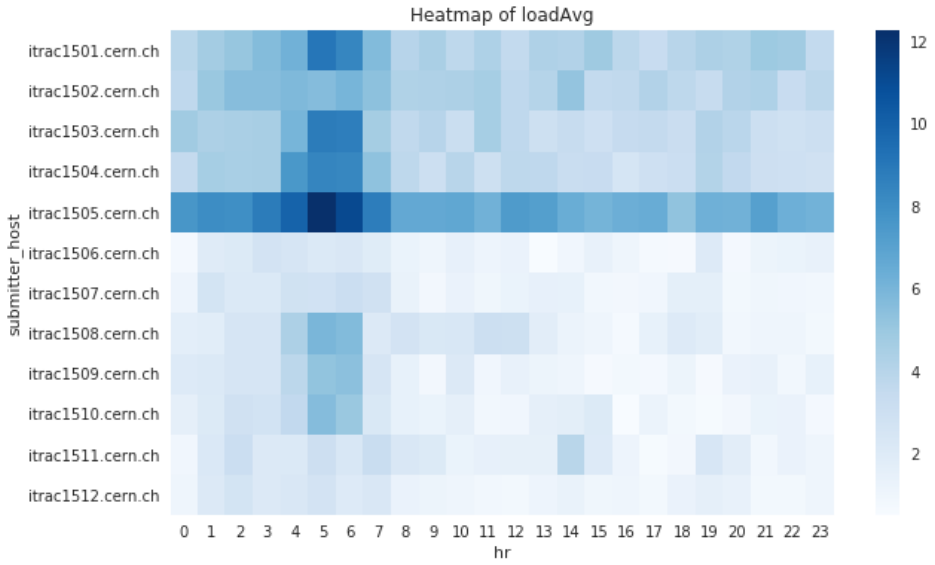
Apache Spark: 90 EXECUTORS 180 CORES Jobs: 1 COMPLETED

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

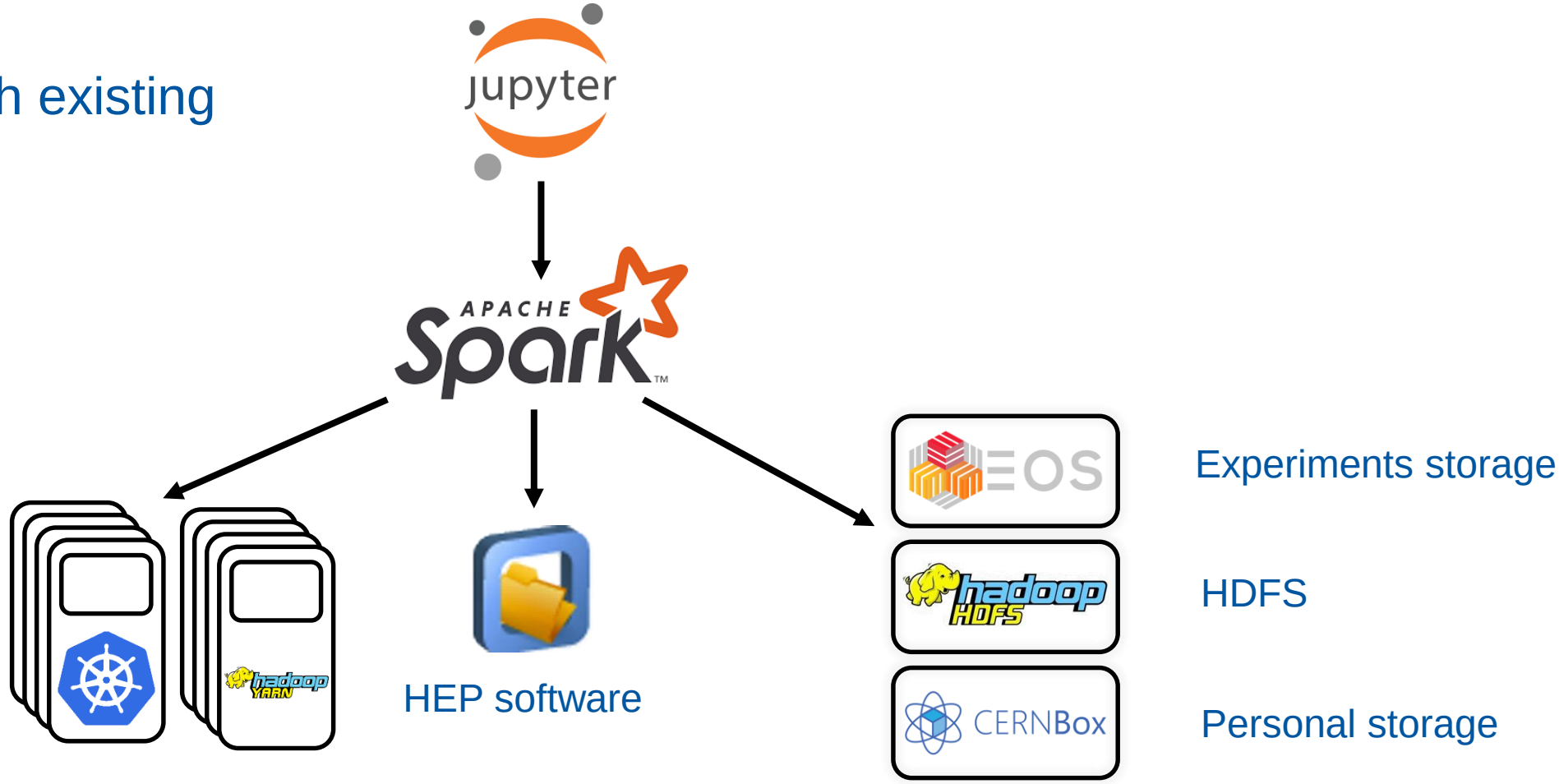
Monitoring

Visualizations

Analytics Platform Outlook

Integrating with existing infrastructure:

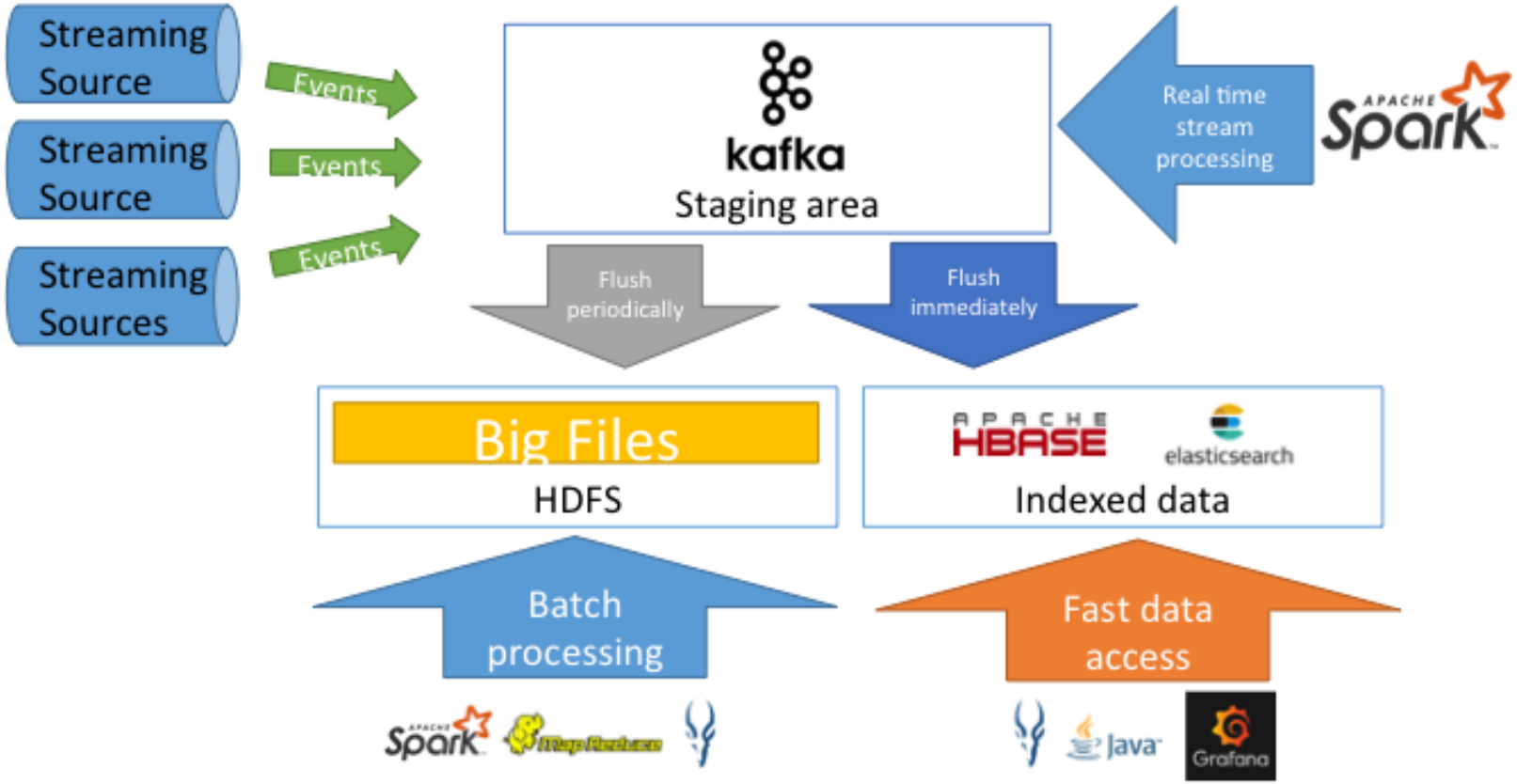
- Software
- Data



Extended Big Data ecosystem

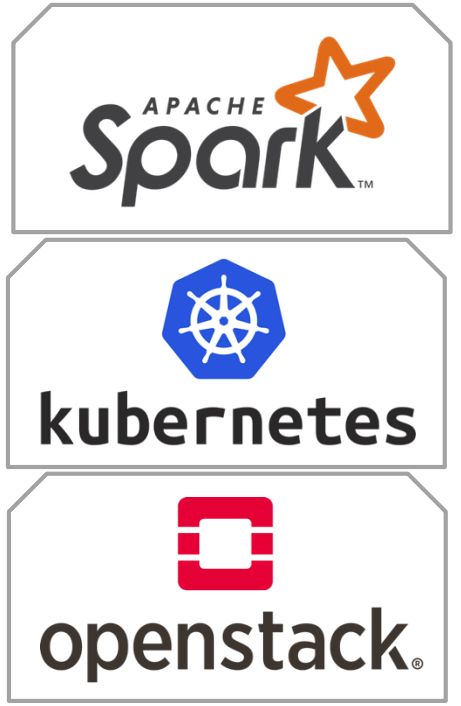
Apache Kafka - data streaming at scale

- Apache Kafka streaming platform becomes a standard component for modern scalable architectures
- Started providing Kafka as a pilot service in 2017



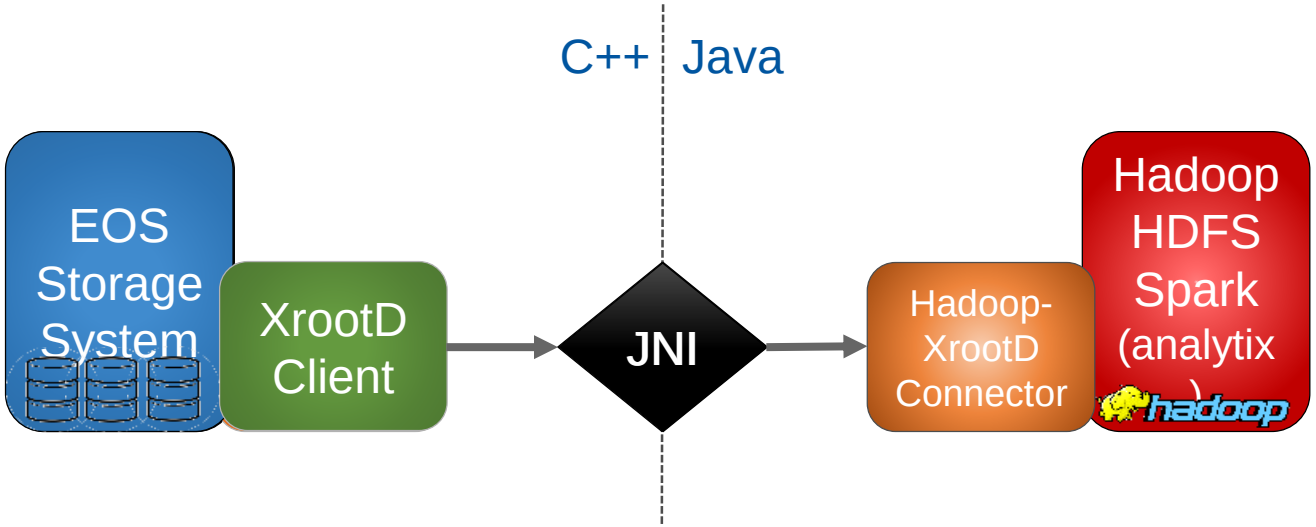
Spark as a service on a private cloud

- Under R&D since 2018
- 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 (AFS, EOS, foreign HDFS)
  - Compute resources can be flexibly scale out
- 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)



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 from EOS and CASTOR directly
  - No need to copy the data to HDFS before processing
  - Works with Grid certificates and Kerberos for authentication



R&D: Data Analysis with PySpark for HEP

ROOT
Data Analysis Framework

EOS Storage Service

WLCG
Worldwide LHC Computing Grid

APACHE Spark™

hadoop

jupyter

SWAN

Conclusions

- Demand of “Big Data” platforms and tools is growing at CERN
  - Many projects started and running
  - Projects around Monitoring, Security, Accelerators logging/controls, physics data, streaming…
- Hadoop, Spark, Kafka services at CERN IT
  - Service is evolving: High availability, security, backups, external data sources, notebooks, cloud…
- Experience and community
  - Technologies evolve rapidly and knowledge sharing very important
  - We are happy to share/exchange our experience, tools, software with others
  - HEP sites welcome to join discussions at Hadoop User Forum: https://indico.cern.ch/category/5894/

CERN

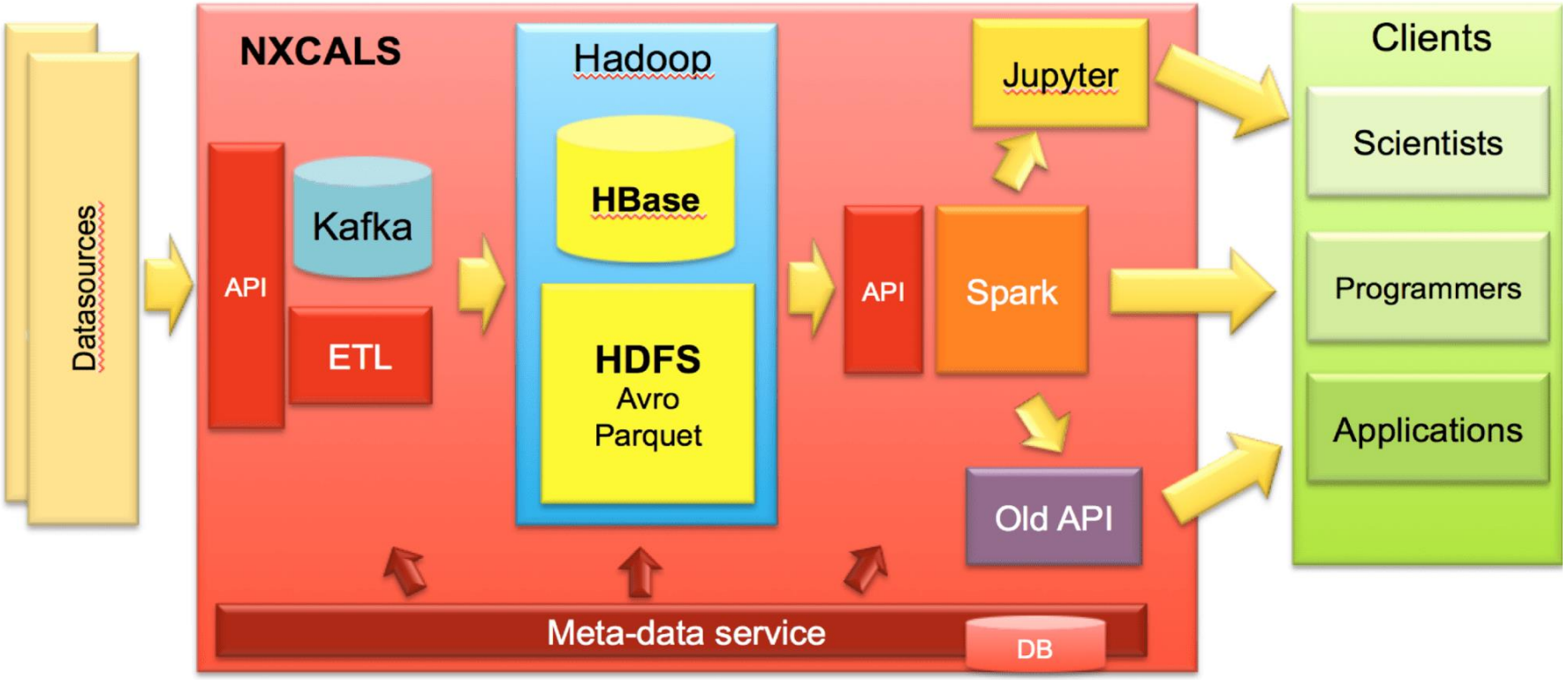
Future work

- Spark on Kubernetes (continuation). For those not profiting from data locality.
  - Ephemeral cluster
  - Bring your own cluster
- SQL on top of Big Data
  - Scale-out database
- Service web portal for users (in-progress)
  - Integrate multiple components of the service
  - Single place for monitoring and requesting the service resources
    - HDFS quota, CPUs, memory, Kafka topics etc.
- Explore further the big data technology landscape
  - Presto, Phoenix, Apache Kudu, Apache Beam, Drill etc.

Selected “Big Data” Projects at CERN

Next Gen. CERN Accelerator Logging

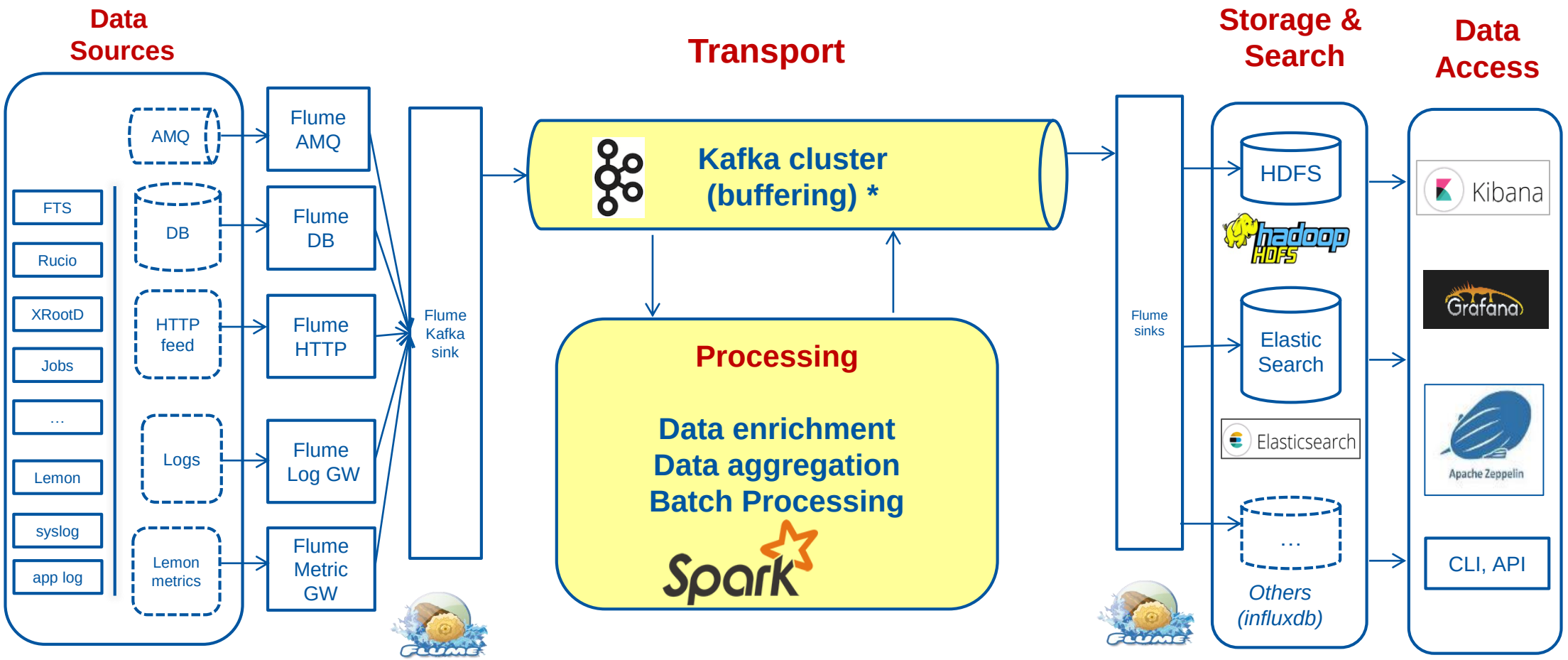
- A control system with: Streaming, Online System, API for Data Extraction
- Critical system for running LHC - 700 TB today, growing 200 TB/year
- Challenge: service level for critical production



Credit: Jakub Wozniak, BE-CO-DS

New CERN IT Monitoring infrastructure

Critical for CC operations and WLCG

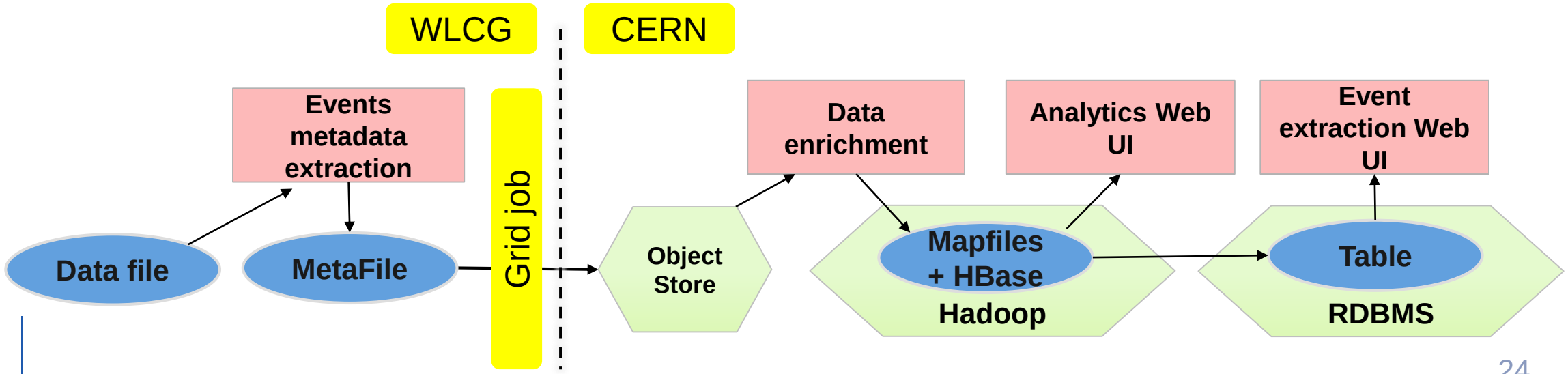


- Data now 200 GB/day, 200M events/day
- At scale 500 GB/day
- Proved to be effective in several occasions

Credits: Alberto Aimar, IT-CM-MM

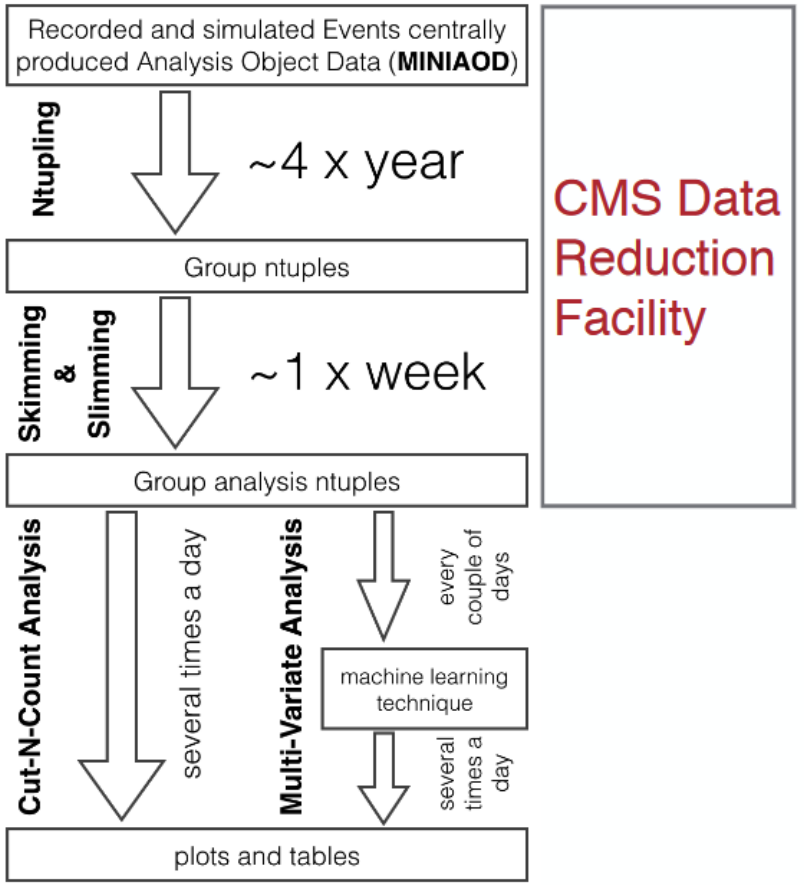
The ATLAS EventIndex

- Catalogue of all collisions in the ATLAS detector
  - Over 120 billion of records, 150TB of data
  - Current ingestion rates 5kHz, 60TB/year



CMS Data Reduction Facility

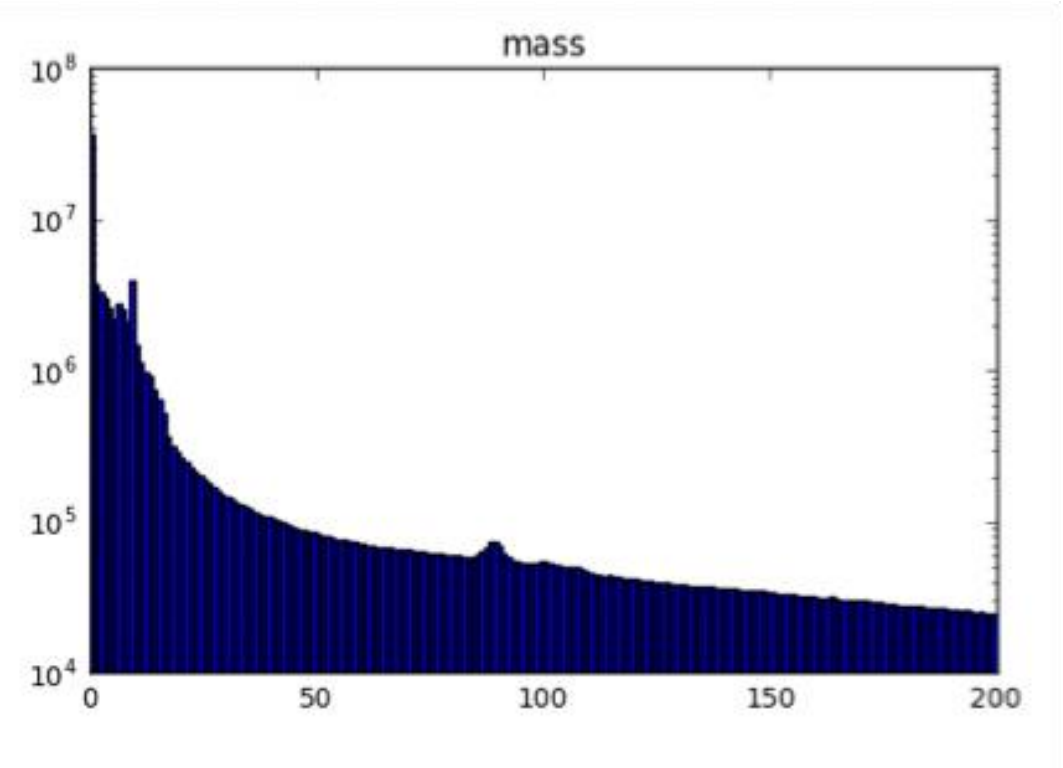
- R&D, CMS Bigdata project, CERN openlab, Intel:
  - Reduce time to physics for PB-sized datasets
  - Exploring a possible new way to do HEP analysis Improve computing resource utilization
  - Enable physicists to use tools and methods from “Big Data” and open source communities
- CMS Data Reduction Facility:
  - Goal: produce reduced data n-tuples for analysis in a more agile way than current methods
  - Currently testing: scale up with larger data sets, first prototype successful but only used 1TB



Data Processing: CMS Open Data Example

Let's calculate the invariant mass of a di-muon system?!

- Transform a collection of muons to an invariant mass for each Row (Event).
- Aggregate (histogram) over the entire dataset.




```
# read in the data
df = sqlContext.read\
                .format("org.dianahep.sparkroot.experimental")\
                .load("hdfs:/path/to/files/*.root")

# count the number of rows:
df.count()

# select only muons
muons =
df.select("patMuons_slimmedMuons__RECO_.patMuons_slimmedMuo
ns__RECO_obj.m_state").toDF("muons")

# map each event to an invariant mass
inv_masses = muons.rdd.map(toInvMass)

# Use histogrammar to perform aggregations
empty = histogrammar.Bin(200, 0, 200, lambda row: row.mass)
h_inv_masses = inv_masses.aggregate(empty,
                histogrammar.increment,
                histogrammar.combine)
```

Credits: V. Khristenko, J. Pivarski, diana-hep and CMS Big Data project

Machine Learning

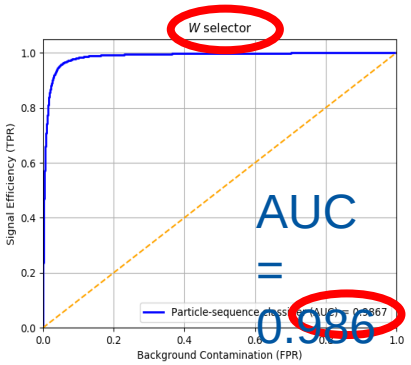
Data and models from Physics

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://indico.cern.ch/event/755842/contributions/3293911/attachments/1784423/2904604/posterMatteo.pdf