Big data tools for LHCb Performance and Regression framework

Maciej Szymański
University of Chinese Academy of Sciences

CERN, 29 May 2018

Software in High Energy Physics

- Software is the **essential component** of modern experiments in HEP

Software in High Energy Physics

- Software is the **essential component** of modern experiments in HEP
- Upgraded on a **short timescales** compared with e.g. detectors

Software in High Energy Physics

- Software is the **essential component** of modern experiments in HEP
- Upgraded on a **short timescales** compared with e.g. detectors
- **Flexible**, but susceptible to issues and bugs

Software in High Energy Physics

- Software is the **essential component** of modern experiments in HEP
- Upgraded on a **short timescales** compared with e.g. detectors
- **Flexible**, but susceptible to issues and bugs
- Need of **systematic regression and performance tests**

The issue

- Tests of software performance often **ran manually by experts**
  - **specific knowledge** required to setup, run and understand the test
  - **resource consuming** to run the code on personal computer
  - **manual comparison** with other versions
  - results usually **not available publicly**

The issue

- Tests of software performance often **ran manually by experts**
    - **specific knowledge** required to setup, run and understand the test
    - **resource consuming** to run the code on personal computer
    - **manual comparison** with other versions
    - results usually **not available publicly**

- **Nightlies** infrastructure **not suitable** for performance tests
    - only boolean result and list of errors or warnings
    - running for all configurations usually not needed
    - risk of interference with builds in case of long tests

LHCb Performance and Regression framework

- **LHCbPR** is a framework for systematic monitoring of the LHCb software

LHCb Performance and Regression framework

- **LHCbPR** is a framework for systematic monitoring of the LHCb software
- Provides performance baseline in **controlled conditions**

LHCb Performance and Regression framework

- **LHCbPR** is a framework for systematic monitoring of the LHCb software
- Provides performance baseline in **controlled conditions**
- Enables to **inspect any changes** due to e.g.:
  - detector description
  - MC generators
  - physics of Geant4
  - new external libraries
  - merge requests

LHCb Performance and Regression framework

- **LHCbPR** is a framework for systematic monitoring of the LHCb software
- Provides performance baseline in **controlled conditions**
- Enables to **inspect any changes** due to e.g.:
  - detector description
  - MC generators
  - physics of Geant4
  - new external libraries
  - merge requests
- **Compare results** across different compilers and architectures

LHCb Performance and Regression framework

- **LHCbPR** is a framework for systematic monitoring of the LHCb software

- Provides performance baseline in **controlled conditions**

- Enables to **inspect any changes** due to e.g.:
  - ○ detector description
  - ○ MC generators
  - ○ physics of Geant4
  - ○ new external libraries
  - ○ merge requests

- **Compare results** across different compilers and architectures

- Not only to monitor resource consumption, but also to **measure the physics performance**

LHCb Performance and Regression framework

- **LHCbPR** is a framework for systematic monitoring of the LHCb software
- Provides performance baseline in **controlled conditions**
- Enables to **inspect any changes** due to e.g.:
  - detector description
  - MC generators
  - physics of Geant4
  - new external libraries
  - merge requests
- **Compare results** across different compilers and architectures
- Not only to monitor resource consumption, but also to **measure the physics performance**
- Benchmarking code (e.g. for reconstruction) especially crucial in the **LHC upgrade era**!

Design of LHCbPR

- Based on **microservice architecture**

Design of LHCbPR

- Based on **microservice architecture**
- **Generic form** of modules enables to apply the service not only in LHCb, but also in other experiments or companies

Design of LHCbPR

- Based on **microservice architecture**
- **Generic form** of modules enables to apply the service not only in LHCb, but also in other experiments or companies
- Usage of **software containers** to facilitate deployment

Design of LHCbPR

- Based on **microservice architecture**
- **Generic form** of modules enables to apply the service not only in LHCb, but also in other experiments or companies
- Usage of **software containers** to facilitate deployment
- **Flexibility** in running the software

Design of LHCbPR

- Based on **microservice architecture**
- **Generic form** of modules enables to apply the service not only in LHCb, but also in other experiments or companies
- Usage of **software containers** to facilitate deployment
- **Flexibility** in running the software
- **Friendly reporting** of the results
  - comparing histograms
  - trend analysis

Infrastructure

- Periodic tests started by the **Jenkins** job
  - nightly builds of the LHCb software are the input flavour
  - tests triggered when corresponding nightly builds ready (using RabbitMQ)

Infrastructure

- Periodic tests started by the **Jenkins** job
  - nightly builds of the LHCb software are the input flavour
  - tests triggered when corresponding nightly builds ready (using RabbitMQ)
- Configuration in **XML** files

Infrastructure

- Periodic tests started by the **Jenkins** job
  - nightly builds of the LHCb software are the input flavour
  - tests triggered when corresponding nightly builds ready (using RabbitMQ)
- Configuration in **XML** files
- Tests are running on dedicated, profiling machines
  - SLC6, CC7

Infrastructure

- Periodic tests started by the **Jenkins** job
  - nightly builds of the LHCb software are the input flavour
  - tests triggered when corresponding nightly builds ready (using RabbitMQ)
- Configuration in **XML** files
- Tests are running on dedicated, profiling machines
  - SLC6, CC7
- Results of the tests **parsed by the specific handlers**
  - to save metrics (int, float, string, file and JSON)

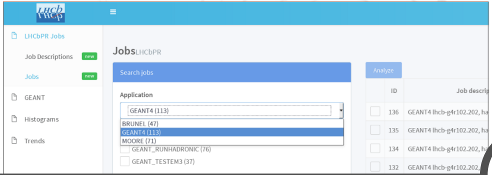
Infrastructure

- Periodic tests started by the **Jenkins** job
  - nightly builds of the LHCb software are the input flavour
  - tests triggered when corresponding nightly builds ready (using RabbitMQ)
- Configuration in **XML** files
- Tests are running on dedicated, profiling machines
  - SLC6, CC7
- Results of the tests **parsed by the specific handlers**
  - to save metrics (int, float, string, file and JSON)
- Zip file sent to the database through **Dirac Storage Element**
  - backend implemented in Django

Infrastructure

- Periodic tests started by the **Jenkins** job
  - nightly builds of the LHCb software are the input flavour
  - tests triggered when corresponding nightly builds ready (using RabbitMQ)
- Configuration in **XML** files
- Tests are running on dedicated, profiling machines
  - SLC6, CC7
- Results of the tests **parsed by the specific handlers**
  - to save metrics (int, float, string, file and JSON)
- Zip file sent to the database through **Dirac Storage Element**
  - backend implemented in Django

- **Web front-end** ▸ lblhcbpr.cern.ch
  - implemented in AngularJS
  - generic ROOT files viewer
  - trend analysis
  - custom modules



JobsLHCbPR

Application

GEANT4 (113)

BRUNEL (47)
GEANT4 (113)
MOORE (71)

□ GEANT_RUNHADRONIC (76)
□ GEANT_TESTEM3 (37)

	ID	Job descrip
□	136	GEANT4 lhcb-g4r102.202, ha
□	135	GEANT4 lhcb-g4r102.202, ha
□	134	GEANT4 lhcb-g4r102.202, ha
□	132	GEANT4 lhcb-g4r102.202,

Some statistics as of today

- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)

Some statistics as of today

- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)
- 44 option files

Some statistics as of today

- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)
- 44 option files
- 120 tests (running on different configutions of compilers and architectures)

Some statistics as of today

- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)
- 44 option files
- 120 tests (running on different configutions of compilers and architectures)
- $\sim$40 tests daily

Some statistics as of today

- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)
- 44 option files
- 120 tests (running on different configutions of compilers and architectures)
- $\sim$40 tests daily
- Single tests run from several minutes up to 10 hours

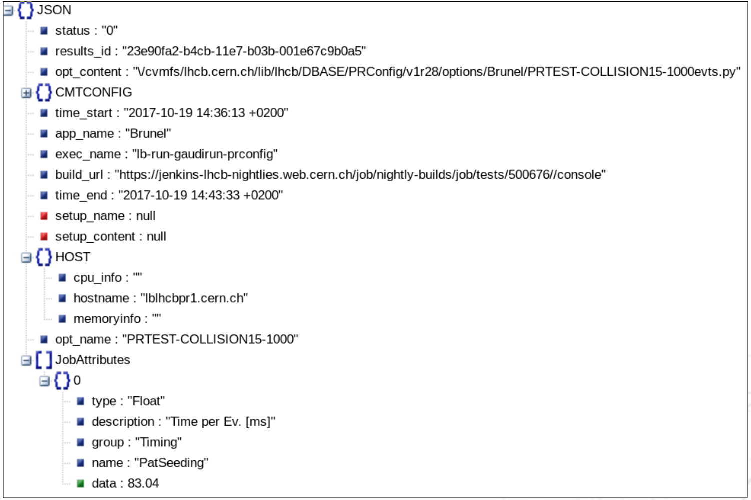
Some statistics as of today

- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)
- 44 option files
- 120 tests (running on different configutions of compilers and architectures)
- $\sim$40 tests daily
- Single tests run from several minutes up to 10 hours
- $\sim$7 GB collected up to now ($\sim$1.5 years)
  - zip files with JSON and ROOT files

Some statistics as of today

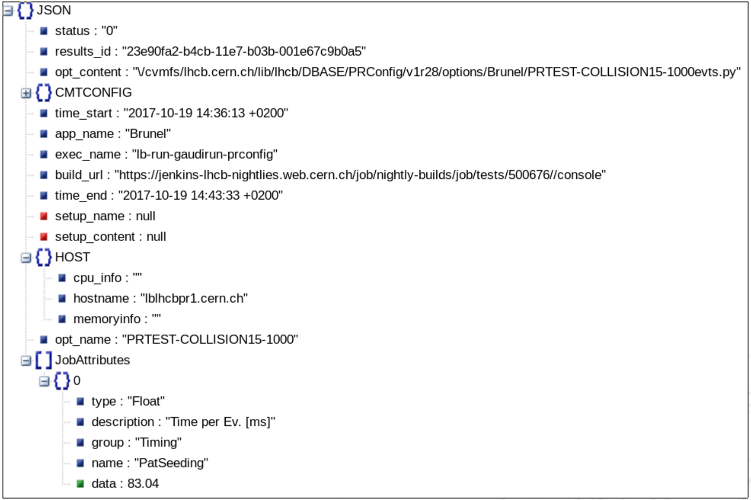
- 5 LHCb applications (e.g. for reconstruction, simulation, triggering)
- 44 option files
- 120 tests (running on different configutions of compilers and architectures)
- $\sim$40 tests daily
- Single tests run from several minutes up to 10 hours
- $\sim$7 GB collected up to now ($\sim$1.5 years)
  - zip files with JSON and ROOT files
- Tens of MB daily

Example of JSON schema



```
⊟ { } JSON
    ■ status : "0"
    ■ results_id : "23e90fa2-b4cb-11e7-b03b-001e67c9b0a5"
    ■ opt_content : "\/cvmfs/lhcb.cern.ch/lib/lhcb/DBASE/PRConfig/v1r28/options/Brunel/PRTEST-COLLISION15-1000evts.py"
  ⊞ { } CMTCONFIG
    ■ time_start : "2017-10-19 14:36:13 +0200"
    ■ app_name : "Brunel"
    ■ exec_name : "lb-run-gaudirun-prconfig"
    ■ build_url : "https://jenkins-lhcb-nightlies.web.cern.ch/job/nightly-builds/job/tests/500676//console"
    ■ time_end : "2017-10-19 14:43:33 +0200"
    ■ setup_name : null
    ■ setup_content : null
  ⊟ { } HOST
        ■ cpu_info : ""
        ■ hostname : "lblhcbpr1.cern.ch"
        ■ memoryinfo : ""
    ■ opt_name : "PRTEST-COLLISION15-1000"
  ⊟ [ ] JobAttributes
      ⊟ { } 0
            ■ type : "Float"
            ■ description : "Time per Ev. [ms]"
            ■ group : "Timing"
            ■ name : "PatSeeding"
            ■ data : 83.04
```

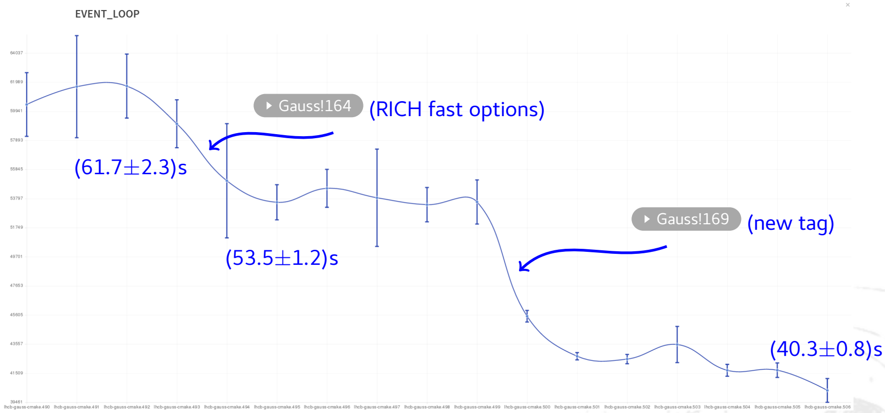
Example of JSON schema



```
{} JSON
   status : "0"
   results_id : "23e90fa2-b4cb-11e7-b03b-001e67c9b0a5"
   opt_content : "\/cvmfs/lhcb.cern.ch/lib/lhcb/DBASE/PRConfig/v1r28/options/Brunel/PRTEST-COLLISION15-1000evts.py"
   {} CMTCONFIG
   time_start : "2017-10-19 14:36:13 +0200"
   app_name : "Brunel"
   exec_name : "lb-run-gaudirun-prconfig"
   build_url : "https://jenkins-lhcb-nightlies.web.cern.ch/job/nightly-builds/job/tests/500676//console"
   time_end : "2017-10-19 14:43:33 +0200"
   setup_name : null
   setup_content : null
   {} HOST
      cpu_info : ""
      hostname : "lblhcbpr1.cern.ch"
      memoryinfo : ""
   opt_name : "PRTEST-COLLISION15-1000"
   [ ] JobAttributes
      {} 0
         type : "Float"
         description : "Time per Ev. [ms]"
         group : "Timing"
         name : "PatSeeding"
         data : 83.04
```

- Main use case: trend analysis for job attributes of interest
- In addition, ROOT files with TH1 and TH2

Timing of simulation application vs. SW version



▸ Gauss!164 (RICH fast options)

(61.7±2.3)s

(53.5±1.2)s

▸ Gauss!169 (new tag)

(40.3±0.8)s

Big data tools for LHCbPR

- How can we profit from the **state-of-art technologies** like Hadoop?

Big data tools for LHCbPR

- How can we profit from the **state-of-art technologies** like Hadoop?
- Increasing **data volume** in LHCbPR

Big data tools for LHCbPR

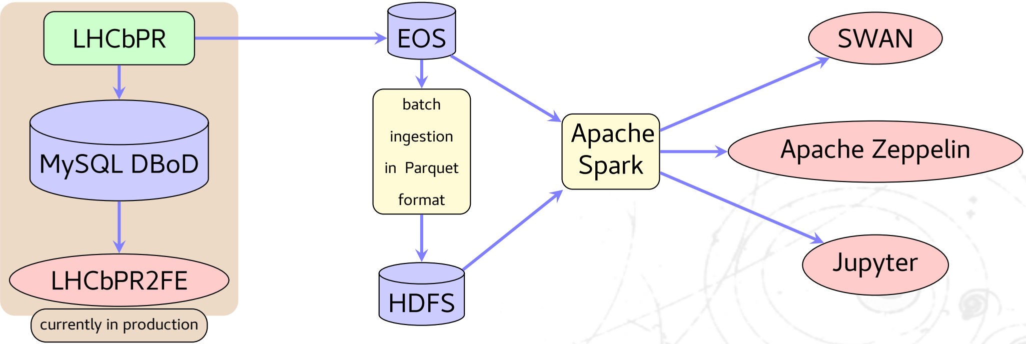
- How can we profit from the **state-of-art technologies** like Hadoop?
- Increasing **data volume** in LHCbPR
- **Interactive** exploration, **fast turn-around** could be very useful feature for the LHCbPR

Big data tools for LHCbPR

- How can we profit from the **state-of-art technologies** like Hadoop?
- Increasing **data volume** in LHCbPR
- **Interactive** exploration, **fast turn-around** could be very useful feature for the LHCbPR
- Investigation of **data processing** tools and **data formats**

Big data tools for LHCbPR

- How can we profit from the **state-of-art technologies** like Hadoop?
- Increasing **data volume** in LHCbPR
- **Interactive** exploration, **fast turn-around** could be very useful feature for the LHCbPR
- Investigation of **data processing** tools and **data formats**



Technologies to consider

- Apache **Spark** used as an engine for data processing
    - fault-tolerant, scalable, fast
    - de facto standard

Technologies to consider

- Apache **Spark** used as an engine for data processing
  - fault-tolerant, scalable, fast
  - de facto standard

- Apache **Parquet** data format seems to be good solution for data format
  - fast analytics and random access
  - easiness of use and integration into framework
  - partitioning crucial for the speed

Technologies to consider

- Apache **Spark** used as an engine for data processing
  - fault-tolerant, scalable, fast
  - de facto standard

- Apache **Parquet** data format seems to be good solution for data format
  - fast analytics and random access
  - easiness of use and integration into framework
  - partitioning crucial for the speed

- Apache **Zeppelin** and **SWAN** used as a notebook
  - collaboration, reproducibility
  - PySpark interpreter included
  - widgets for data inputs (e.g. drop-down list, text field)

Setup of the prototype

- **First look** on Hadoop ecosystem using local machine

Setup of the prototype

- **First look** on Hadoop ecosystem using local machine
- Moving to CERN infrastructure (`analytix` node) after **promising results**

Setup of the prototype

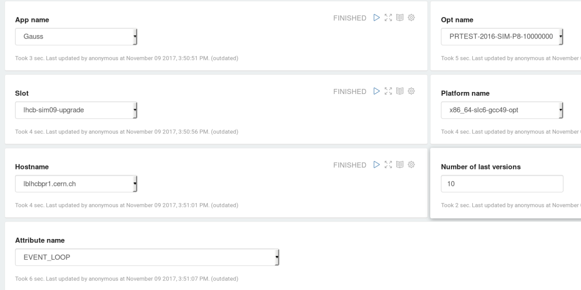
- **First look** on Hadoop ecosystem using local machine

- Moving to CERN infrastructure (`analytix` node) after **promising results**

- **Porting analysis modules** to Zeppelin/SWAN notebooks using both Scala and PySpark

Setup of the prototype

- **First look** on Hadoop ecosystem using local machine
- Moving to CERN infrastructure (`analytix` node) after **promising results**
- **Porting analysis modules** to Zeppelin/SWAN notebooks using both Scala and PySpark
- Using widgets to mimic the interface of current LHCbPR web frontend



App name

Gauss

Took 3 sec. Last updated by anonymous at November 09 2017, 3:50:51 PM. (outdated)

Opt name

PRTEST-2016-SIM-P8-10000000

Took 5 sec. Last updated by anonymous at November

Slot

lhcb-sim09-upgrade

Took 4 sec. Last updated by anonymous at November 09 2017, 3:50:56 PM. (outdated)

Platform name

x86_64-slc6-gcc49-opt

Took 4 sec. Last updated by anonymous at November

Hostname

lblhcbpr1.cern.ch

Took 4 sec. Last updated by anonymous at November 09 2017, 3:51:01 PM. (outdated)

Number of last versions

10

Took 2 sec. Last updated by anonymous at November

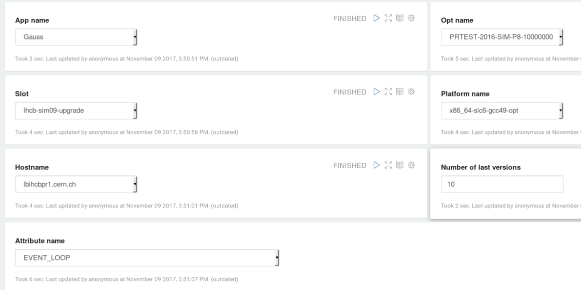
Attribute name

EVENT_LOOP

Took 6 sec. Last updated by anonymous at November 09 2017, 3:51:07 PM. (outdated)

Setup of the prototype

- **First look** on Hadoop ecosystem using local machine
- Moving to CERN infrastructure (`analytix` node) after **promising results**
- **Porting analysis modules** to Zeppelin/SWAN notebooks using both Scala and PySpark
- Using widgets to mimic the interface of current LHCbPR web frontend
- `matplotlib` for **displaying** the data



App name

Gauss

Took 3 sec. Last updated by anonymous at November 09 2017, 3:50:51 PM. (outdated)

Opt name

PRTEST-2016-SIM-P8-10000000

Took 5 sec. Last updated by anonymous at November

Slot

lhcb-sim09-upgrade

Took 4 sec. Last updated by anonymous at November 09 2017, 3:50:56 PM. (outdated)

Platform name

x86_64-slc6-gcc49-opt

Took 4 sec. Last updated by anonymous at November

Hostname

lblhcbpr1.cern.ch

Took 4 sec. Last updated by anonymous at November 09 2017, 3:51:01 PM. (outdated)

Number of last versions

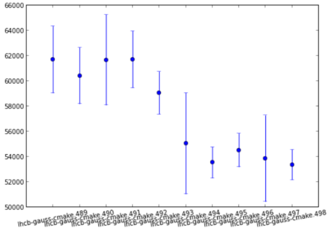
10

Took 2 sec. Last updated by anonymous at November

Attribute name

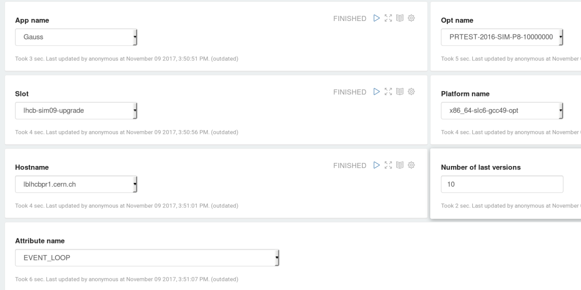
EVENT_LOOP

Took 6 sec. Last updated by anonymous at November 09 2017, 3:51:07 PM. (outdated)



Setup of the prototype

- **First look** on Hadoop ecosystem using local machine
- Moving to CERN infrastructure (`analytix` node) after **promising results**
- **Porting analysis modules** to Zeppelin/SWAN notebooks using both Scala and PySpark
- Using widgets to mimic the interface of current LHCbPR web frontend
- `matplotlib` for **displaying** the data
- Preference for using SWAN
    - once Spark and HDFS integration was provided



App name
Gauss
Took 3 sec. Last updated by anonymous at November 09 2017, 3:50:51 PM. (outdated)

Opt name
PRTEST-2016-SIM-P8-10000000
Took 5 sec. Last updated by anonymous at November

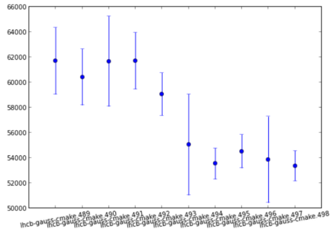
Slot
lhcb-sim09-upgrade
Took 4 sec. Last updated by anonymous at November 09 2017, 3:50:56 PM. (outdated)

Platform name
x86_64-slc6-gcc49-opt
Took 4 sec. Last updated by anonymous at November

Hostname
lblhcbpr1.cern.ch
Took 4 sec. Last updated by anonymous at November 09 2017, 3:51:01 PM. (outdated)

Number of last versions
10
Took 2 sec. Last updated by anonymous at November

Attribute name
EVENT_LOOP
Took 6 sec. Last updated by anonymous at November 09 2017, 3:51:07 PM. (outdated)



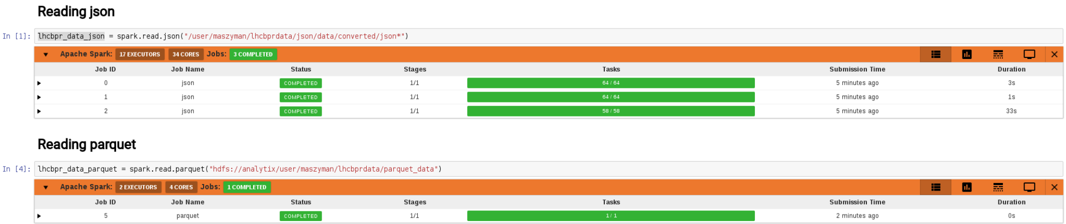
Data ingestion

- First, batch copy of all the historical LHCbPR data into HDFS

Data ingestion

- First, batch copy of all the historical LHCbPR data into HDFS
- Then, automated daily copies (cron job):
    - framework produces test results in JSON format
    - files saved on EOS
    - cron job merges JSON files once a day copies them to HDFS
    - copy to HDFS
        - `hdfs dfs -put` command
    - another cron job converts data into Parquet format
        - compression ratio wrt JSON: ~ 16

Read and data scan performance



Reading json

```
In [1]: lhcbpr_data_json = spark.read.json("/user/maszyman/lhcbprdata/json/data/converted/json*")
```

▼ Apache Spark: 17 EXECUTORS 34 CORES Jobs: 3 COMPLETED

	Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
▸	0	json	COMPLETED	1/1	64 / 64	5 minutes ago	3s
▸	1	json	COMPLETED	1/1	64 / 64	5 minutes ago	1s
▸	2	json	COMPLETED	1/1	58 / 58	5 minutes ago	33s

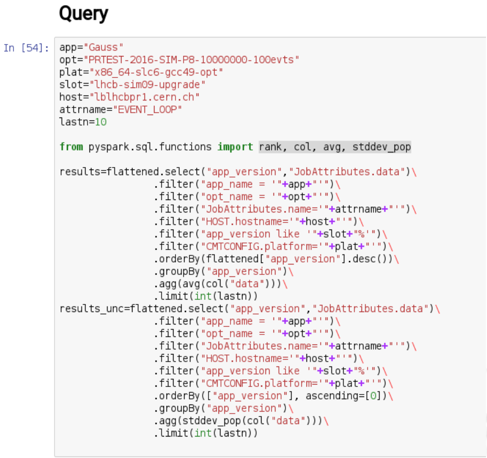
Reading parquet

```
In [4]: lhcbpr_data_parquet = spark.read.parquet("hdfs://analytix/user/maszyman/lhcbprdata/parquet_data")
```

▼ Apache Spark: 2 EXECUTORS 4 CORES Jobs: 1 COMPLETED

	Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
▸	5	parquet	COMPLETED	1/1	1 / 1	2 minutes ago	0s

Read and data scan performance



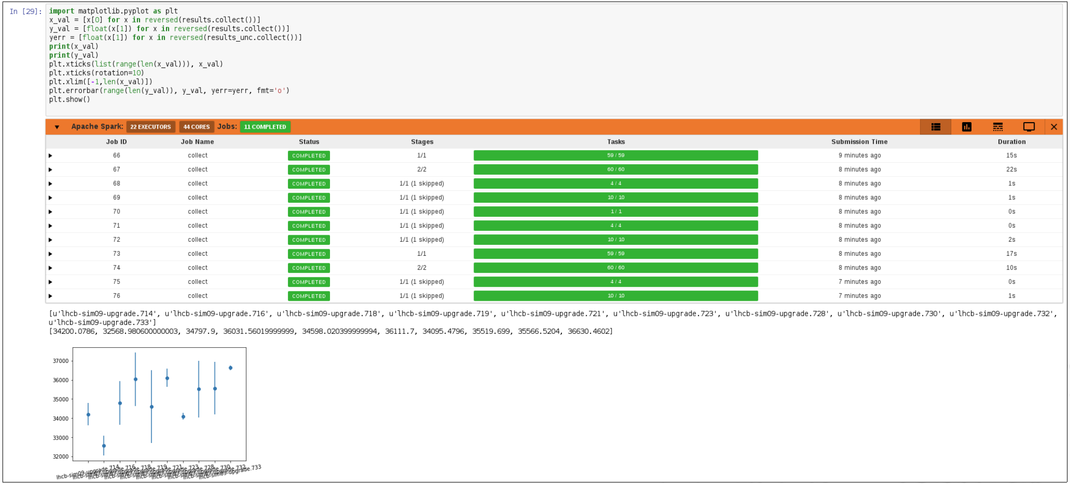
Query

```
In [54]: app="Gauss"
         opt="PRTEST-2016-SIM-P8-10000000-100evts"
         plat="x86_64-slc6-gcc49-opt"
         slot="lhcb-sim09-upgrade"
         host="lblhcbpr1.cern.ch"
         attrname="EVENT_LOOP"
         lastn=10

         from pyspark.sql.functions import rank, col, avg, stddev_pop

         results=flattened.select("app_version","JobAttributes.data")\
                         .filter("app_name = '"+app+"'")\
                         .filter("opt_name = '"+opt+"'")\
                         .filter("JobAttributes.name='"+attrname+"'")\
                         .filter("HOST.hostname='"+host+"'")\
                         .filter("app_version like '"+slot+"%'")\
                         .filter("CMTCONFIG.platform='"+plat+"'")\
                         .orderBy(flattened["app_version"].desc())\
                         .groupBy("app_version")\
                         .agg(avg(col("data")))\
                         .limit(int(lastn))
         results_unc=flattened.select("app_version","JobAttributes.data")\
                         .filter("app_name = '"+app+"'")\
                         .filter("opt_name = '"+opt+"'")\
                         .filter("JobAttributes.name='"+attrname+"'")\
                         .filter("HOST.hostname='"+host+"'")\
                         .filter("app_version like '"+slot+"%'")\
                         .filter("CMTCONFIG.platform='"+plat+"'")\
                         .orderBy(["app_version"], ascending=[0])\
                         .groupBy("app_version")\
                         .agg(stddev_pop(col("data")))\
                         .limit(int(lastn))
```

Read and data scan performance



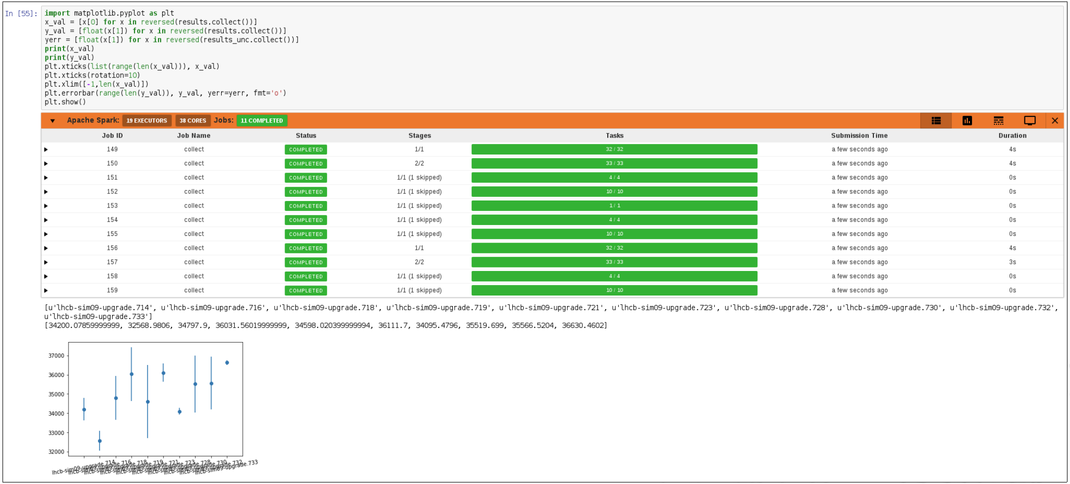
In [29]:

```
import matplotlib.pyplot as plt
x_val = [x[0] for x in reversed(results.collect())]
y_val = [float(x[1]) for x in reversed(results.collect())]
yerr = [float(x[1]) for x in reversed(results_unc.collect())]
print(x_val)
print(y_val)
plt.xticks(list(range(len(x_val))), x_val)
plt.xticks(rotation=10)
plt.xlim([-1,len(x_val)])
plt.errorbar(range(len(y_val)), y_val, yerr=yerr, fmt='o')
plt.show()
```

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
66	collect	COMPLETED	1/1	59 / 59	9 minutes ago	15s
67	collect	COMPLETED	2/2	60 / 60	8 minutes ago	22s
68	collect	COMPLETED	1/1 (1 skipped)	4 / 4	8 minutes ago	1s
69	collect	COMPLETED	1/1 (1 skipped)	10 / 10	8 minutes ago	1s
70	collect	COMPLETED	1/1 (1 skipped)	1 / 1	8 minutes ago	0s
71	collect	COMPLETED	1/1 (1 skipped)	4 / 4	8 minutes ago	0s
72	collect	COMPLETED	1/1 (1 skipped)	10 / 10	8 minutes ago	2s
73	collect	COMPLETED	1/1	59 / 59	8 minutes ago	17s
74	collect	COMPLETED	2/2	60 / 60	8 minutes ago	10s
75	collect	COMPLETED	1/1 (1 skipped)	4 / 4	7 minutes ago	0s
76	collect	COMPLETED	1/1 (1 skipped)	10 / 10	7 minutes ago	1s

```
[u'lhcb-sim09-upgrade.714', u'lhcb-sim09-upgrade.716', u'lhcb-sim09-upgrade.718', u'lhcb-sim09-upgrade.719', u'lhcb-sim09-upgrade.721', u'lhcb-sim09-upgrade.723', u'lhcb-sim09-upgrade.728', u'lhcb-sim09-upgrade.730', u'lhcb-sim09-upgrade.732',
u'lhcb-sim09-upgrade.733']
[34200.0786, 32568.980600000003, 34797.9, 36031.56019999999, 34598.020399999994, 36111.7, 34095.4796, 35519.699, 35566.5204, 36630.4602]
```

Read and data scan performance



In [55]:

```
import matplotlib.pyplot as plt
x_val = [x[0] for x in reversed(results.collect())]
y_val = [float(x[1]) for x in reversed(results.collect())]
yerr = [float(x[1]) for x in reversed(results_unc.collect())]
print(x_val)
print(y_val)
plt.xticks(list(range(len(x_val))), x_val)
plt.xticks(rotation=10)
plt.xlim([-1,len(x_val)])
plt.errorbar(range(len(y_val)), y_val, yerr=yerr, fmt='o')
plt.show()
```

Apache Spark: 19 EXECUTORS 38 CORES Jobs: 11 COMPLETED

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
149	collect	COMPLETED	1/1	32 / 32	a few seconds ago	4s
150	collect	COMPLETED	2/2	33 / 33	a few seconds ago	4s
151	collect	COMPLETED	1/1 (1 skipped)	4 / 4	a few seconds ago	0s
152	collect	COMPLETED	1/1 (1 skipped)	10 / 10	a few seconds ago	0s
153	collect	COMPLETED	1/1 (1 skipped)	1 / 1	a few seconds ago	0s
154	collect	COMPLETED	1/1 (1 skipped)	4 / 4	a few seconds ago	0s
155	collect	COMPLETED	1/1 (1 skipped)	10 / 10	a few seconds ago	0s
156	collect	COMPLETED	1/1	32 / 32	a few seconds ago	4s
157	collect	COMPLETED	2/2	33 / 33	a few seconds ago	3s
158	collect	COMPLETED	1/1 (1 skipped)	4 / 4	a few seconds ago	0s
159	collect	COMPLETED	1/1 (1 skipped)	10 / 10	a few seconds ago	0s

```
[u'lhcb-sim09-upgrade.714', u'lhcb-sim09-upgrade.716', u'lhcb-sim09-upgrade.718', u'lhcb-sim09-upgrade.719', u'lhcb-sim09-upgrade.721', u'lhcb-sim09-upgrade.723', u'lhcb-sim09-upgrade.728', u'lhcb-sim09-upgrade.730', u'lhcb-sim09-upgrade.732',
u'lhcb-sim09-upgrade.733']
[34200.07859999999, 32568.9806, 34797.9, 36031.56019999999, 34598.020399999994, 36111.7, 34095.4796, 35519.699, 35566.5204, 36630.4602]
```

Read and data scan performance

	load data	benchmark query
JSON	31s	69s
Parquet	<1s	15s

Table : Query: filter values of the metric by application name, option, host, platform

What about ROOT files?

- Following effort from: `https://github.com/diana-hep/spark-root`

What about ROOT files?

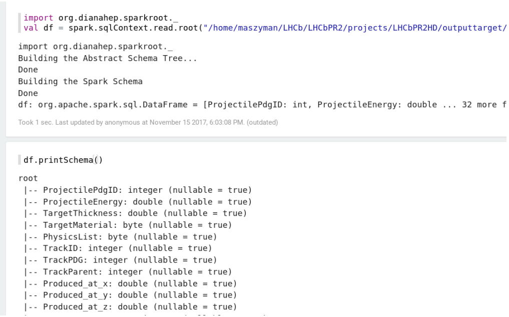
- Following effort from: `https://github.com/diana-hep/spark-root`
- Support for TTree and primitive data types
  - would be useful to read TH1, TH2, etc.

What about ROOT files?

- Following effort from: `https://github.com/diana-hep/spark-root`
- Support for TTree and primitive data types
  - would be useful to read TH1, TH2, etc.
- Simple use case with reading one of the files from G4 test

What about ROOT files?

- Following effort from: `https://github.com/diana-hep/spark-root`
- Support for TTree and primitive data types
  - would be useful to read TH1, TH2, etc.
- Simple use case with reading one of the files from G4 test



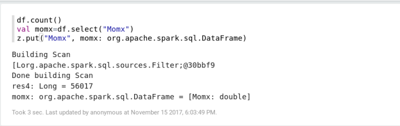
```
import org.dianahep.sparkroot._
val df = spark.sqlContext.read.root("/home/maszyman/LHCb/LHCbPR2/projects/LHCbPR2HD/outputtarget/
```

```
import org.dianahep.sparkroot._
Building the Abstract Schema Tree...
Done
Building the Spark Schema
Done
df: org.apache.spark.sql.DataFrame = [ProjectilePdgID: int, ProjectileEnergy: double ... 32 more f
```

Took 1 sec. Last updated by anonymous at November 15 2017, 6:03:08 PM. (outdated)

```
df.printSchema()
```

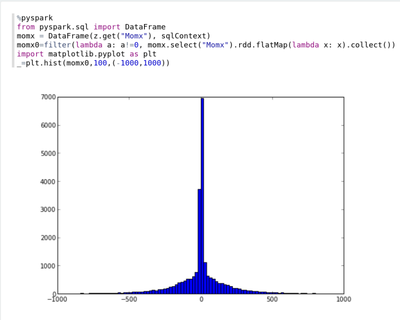
```
root
 |-- ProjectilePdgID: integer (nullable = true)
 |-- ProjectileEnergy: double (nullable = true)
 |-- TargetThickness: double (nullable = true)
 |-- TargetMaterial: byte (nullable = true)
 |-- PhysicsList: byte (nullable = true)
 |-- TrackID: integer (nullable = true)
 |-- TrackPDG: integer (nullable = true)
 |-- TrackParent: integer (nullable = true)
 |-- Produced_at_x: double (nullable = true)
 |-- Produced_at_y: double (nullable = true)
 |-- Produced_at_z: double (nullable = true)
```



```
df.count()
val momx=df.select("Momx")
z.put("Momx", momx: org.apache.spark.sql.DataFrame)
```

```
Building Scan
[Lorg.apache.spark.sql.sources.Filter;@30bbf9
Done building Scan
res4: Long = 56017
momx: org.apache.spark.sql.DataFrame = [Momx: double]
```

Took 3 sec. Last updated by anonymous at November 15 2017, 6:03:49 PM.



```
%pyspark
from pyspark.sql import DataFrame
momx = DataFrame(z.get("Momx"), sqlContext)
momx0=filter(lambda a: a!=0, momx.select("Momx").rdd.flatMap(lambda x: x).collect())
import matplotlib.pyplot as plt
_=plt.hist(momx0,100,(-1000,1000))
```

Remarks on using Hadoop in LHCbPR

- Hadoop approach gives much more **flexibility**
  - develop **custom scripts** in place and see the results immediately
  - **easier and faster** access to data
  - useful for data **exploration** (e.g. plotting specific metrics filtered according to current needs)
  - now, need to implement custom analysis module in JS, create MR, and deploy new image

Remarks on using Hadoop in LHCbPR

- Hadoop approach gives much more **flexibility**
  - develop **custom scripts** in place and see the results immediately
  - **easier and faster** access to data
  - useful for data **exploration** (e.g. plotting specific metrics filtered according to current needs)
  - now, need to implement custom analysis module in JS, create MR, and deploy new image

- **Relational DB not really mandatory** for the LHCbPR case

Remarks on using Hadoop in LHCbPR

- Hadoop approach gives much more **flexibility**
    - develop **custom scripts** in place and see the results immediately
    - **easier and faster** access to data
    - useful for data **exploration** (e.g. plotting specific metrics filtered according to current needs)
    - now, need to implement custom analysis module in JS, create MR, and deploy new image

- **Relational DB not really mandatory** for the LHCbPR case

- The plan is to **combine** the current Django + AngularJS approach with HDFS + SWAN
    - reading data using WebHDFS protocol

Remarks on using Hadoop in LHCbPR

- Hadoop approach gives much more **flexibility**
  - develop **custom scripts** in place and see the results immediately
  - **easier and faster** access to data
  - useful for data **exploration** (e.g. plotting specific metrics filtered according to current needs)
  - now, need to implement custom analysis module in JS, create MR, and deploy new image

- **Relational DB not really mandatory** for the LHCbPR case

- The plan is to **combine** the current Django + AngularJS approach with HDFS + SWAN
  - reading data using WebHDFS protocol

- **Porting** existing analysis modules **impossible one-to-one**
  - interactive plots (clickable data points)
  - less functionality in terms of user interface
  - sufficient for trend analysis and basic comparison of various metrics

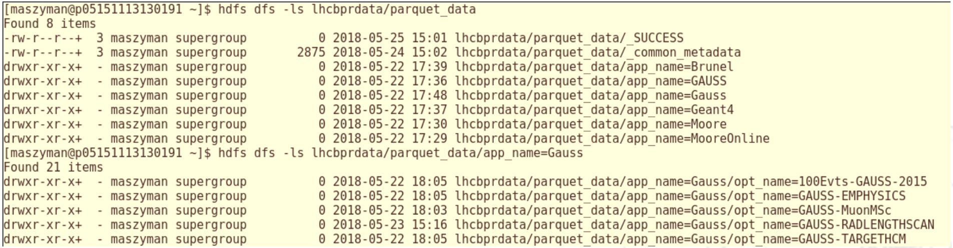
Remarks on Hadoop service

- **Scala** interpreter seems to be **faster than PySpark**
  - but the users want to use **python**!
  - more encouraging to implement custom reports for newcomers

Remarks on Hadoop service

- **Scala** interpreter seems to be **faster than PySpark**
  - but the users want to use **python**!
  - more encouraging to implement custom reports for newcomers

- Significant **speedup when data partitioned** properly
  - depending on the specific queries
  - in our use case partitioning by application and option names

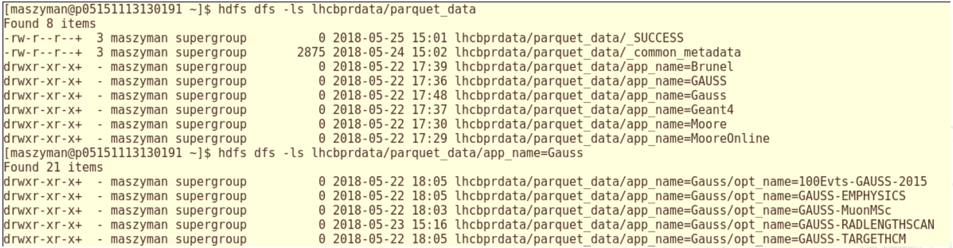


```
[maszyman@p05151113130191 ~]$ hdfs dfs -ls lhcbprdata/parquet_data
Found 8 items
-rw-r--r--+  3 maszyman supergroup          0 2018-05-25 15:01 lhcbprdata/parquet_data/_SUCCESS
-rw-r--r--+  3 maszyman supergroup       2875 2018-05-24 15:02 lhcbprdata/parquet_data/_common_metadata
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:39 lhcbprdata/parquet_data/app_name=Brunel
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:36 lhcbprdata/parquet_data/app_name=GAUSS
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:48 lhcbprdata/parquet_data/app_name=Gauss
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:37 lhcbprdata/parquet_data/app_name=Geant4
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:30 lhcbprdata/parquet_data/app_name=Moore
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:29 lhcbprdata/parquet_data/app_name=MooreOnline
[maszyman@p05151113130191 ~]$ hdfs dfs -ls lhcbprdata/parquet_data/app_name=Gauss
Found 21 items
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:05 lhcbprdata/parquet_data/app_name=Gauss/opt_name=100Evts-GAUSS-2015
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:05 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-EMPHYSICS
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:03 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-MuonMSc
drwxr-xr-x+  - maszyman supergroup          0 2018-05-23 15:16 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-RADLENGTHSCAN
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:05 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-TARGETHCM
```

Remarks on Hadoop service

- **Scala** interpreter seems to be **faster than PySpark**
  - but the users want to use **python**!
  - more encouraging to implement custom reports for newcomers

- Significant **speedup when data partitioned** properly
  - depending on the specific queries
  - in our use case partitioning by application and option names



```
[maszyman@p05151113130191 ~]$ hdfs dfs -ls lhcbprdata/parquet_data
Found 8 items
-rw-r--r--+  3 maszyman supergroup          0 2018-05-25 15:01 lhcbprdata/parquet_data/_SUCCESS
-rw-r--r--+  3 maszyman supergroup       2875 2018-05-24 15:02 lhcbprdata/parquet_data/_common_metadata
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:39 lhcbprdata/parquet_data/app_name=Brunel
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:36 lhcbprdata/parquet_data/app_name=GAUSS
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:48 lhcbprdata/parquet_data/app_name=Gauss
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:37 lhcbprdata/parquet_data/app_name=Geant4
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:30 lhcbprdata/parquet_data/app_name=Moore
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 17:29 lhcbprdata/parquet_data/app_name=MooreOnline
[maszyman@p05151113130191 ~]$ hdfs dfs -ls lhcbprdata/parquet_data/app_name=Gauss
Found 21 items
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:05 lhcbprdata/parquet_data/app_name=Gauss/opt_name=100Evts-GAUSS-2015
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:05 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-EMPHYSICS
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:03 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-MuonMSc
drwxr-xr-x+  - maszyman supergroup          0 2018-05-23 15:16 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-RADLENGTHSCAN
drwxr-xr-x+  - maszyman supergroup          0 2018-05-22 18:05 lhcbprdata/parquet_data/app_name=Gauss/opt_name=GAUSS-TARGETHCM
```

- **More than one Spark connection** in SWAN would be nice

Summary

- Monitoring of the software is **essential tool** in large scientific project such as LHCb

Summary

- Monitoring of the software is **essential tool** in large scientific project such as LHCb
- **LHCbPR** has already shown to be **versatile framework** useful for the whole Collaboration

Summary

- Monitoring of the software is **essential tool** in large scientific project such as LHCb
- **LHCbPR** has already shown to be **versatile framework** useful for the whole Collaboration
- **Big data** tools appear to be **promising** for the integration with LHCbPR

Summary

- Monitoring of the software is **essential tool** in large scientific project such as LHCb
- **LHCbPR** has already shown to be **versatile framework** useful for the whole Collaboration
- **Big data** tools appear to be **promising** for the integration with LHCbPR
- **Many thanks** to CERN IT for providing the service!