Improvements to LHCbPR2

Maciej Szymański ¹, A.Mazurov ², B.Couturier ³

15.05.2017

¹ University of Chinese Academy of Sciences
² University of Birmingham
³ CERN
Necessity of code validation

- **Systematic measurements of software performance critical** for the profiling and validation.
Necessity of code validation

- **Systematic measurements of software performance critical** for the profiling and validation
- Need for performance baseline in **controlled conditions**
Necessity of code validation

• **Systematic measurements of software performance critical** for the profiling and validation
• Need for performance baseline in **controlled conditions**
• **Inspect any changes** due to e.g. detector description, MC generators, physics of Geant4, new external libraries
Necessity of code validation

- **Systematic measurements of software performance critical** for the profiling and validation
- Need for performance baseline in **controlled conditions**
- **Inspect any changes** due to e.g. detector description, MC generators, physics of Geant4, new external libraries
- **Compare results** across different compilers and architectures
Necessity of code validation

- **Systematic measurements of software performance critical** for the profiling and validation
- Need for performance baseline in **controlled conditions**
- **Inspect any changes** due to e.g. detector description, MC generators, physics of Geant4, new external libraries
- **Compare results** across different compilers and architectures
- Testing **special configurations** and features
Necessity of code validation

- **Systematic measurements of software performance critical** for the profiling and validation
- Need for performance baseline in **controlled conditions**
- **Inspect any changes** due to e.g. detector description, MC generators, physics of Geant4, new external libraries
- **Compare results** across different compilers and architectures
- Testing **special configurations** and features
- Efficient **check for bugs**
Necessity of code validation

• **Systematic measurements of software performance critical** for the profiling and validation
• Need for performance baseline in **controlled conditions**
• **Inspect any changes** due to e.g. detector description, MC generators, physics of Geant4, new external libraries
• **Compare results** across different compilers and architectures
• Testing **special configurations** and features
• Efficient **check for bugs**
• Not only to monitor resource consumption, but also to **measure the physics performance**
Necessity of code validation

- **Systematic measurements of software performance critical** for the profiling and validation
- Need for performance baseline in **controlled conditions**
- **Inspect any changes** due to e.g. detector description, MC generators, physics of Geant4, new external libraries
- **Compare results** across different compilers and architectures
- Testing **special configurations** and features
- Efficient **check for bugs**
- Not only to monitor resource consumption, but also to **measure the physics performance**
- **Daily maintenance** and user support crucial
The issue

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

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

- 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**
Requirements for the framework

• Definition of tests in **software repository**
  ○ e.g. PRConfig
Requirements for the framework

- Definition of tests in **software repository**
  - e.g. PRConfig
- **Flexibility** in running the software
  - choice of test frequency
  - usage of specific platform
Requirements for the framework

- Definition of tests in **software repository**
  - e.g. PRConfig

- **Flexibility** in running the software
  - choice of test frequency
  - usage of specific platform

- **Friendly reporting** of the results (e.g. comparing histograms, trend analysis, memory and CPU consumption)
Based on **microservice architecture**
**LHCb Performance and Regression**

- Based on **microservice architecture**
- **Loosely coupled modules** communicating through simple API’s
LHCb Performance and Regression

- Based on **microservice architecture**
- **Loosely coupled modules** communicating through simple API’s
- **Easy** to understand, maintain and scale
Based on microservice architecture

Loosely coupled modules communicating through simple API’s

Easy to understand, maintain and scale

Usage of software containers and management tools facilitates deployment process
LHCb Performance and Regression

- Based on **microservice architecture**

- **Loosely coupled modules** communicating through simple API’s
- **Easy** to understand, maintain and scale
- Usage of **software containers** and management tools facilitates deployment process
- **Generic form** of modules enables to apply the proposed architecture not only in LHCb, but can be adopted elsewhere
Sequence diagram

Build and Test Services

get parameters of the tests

run tests

save results

LHCbPR

analyse

User Clients

A.Mazurov, CHEP 2016
Infrastructure (1)

- Periodic tests started by the **Jenkins** job
  - nightly builds as input flavour

Maciej Szymański (UCAS)
LHCb computing workshop, 15.05.2017
Infrastructure (1)

- Periodic tests started by the **Jenkins** job
  - nightly builds as input flavour

- Configuration in **XML** files
  - https://gitlab.cern.ch/lhcb-core/LHCbNightlyConf/blob/master/test_schedule2.xml

```xml
<periodictest>
  <schedule type="week" time="10:00">Mon,Tue,Wed,Thu,Fri</schedule>
  <slot>lhcb-future</slot>
  <project>Brunel</project>
  <platform>x86_64-slc6-gcc62-opt</platform>
  <test runner="lhbpr" group="MiniBrunel" env="lb-run-gaurdin|TimeLineHandler"/>
  <os_label>perf</os_label>
  <count>5</count>
</periodictest>
```
Infrastructure (1)

• Periodic tests started by the **Jenkins job**
  - nightly builds as input flavour

• Configuration in **XML files**

```xml
<periodictest>
  <schedule type="week" time="10:00">Mon, Tue, Wed, Thu, Fri</schedule>
  <slot>lhcb-future</slot>
  <project>Brunel</project>
  <platform>x86_64-slc6-gcc62-opt</platform>
  <test runner="lhcbrpr" group="MiniBrunel" env="1b-run-gaudirun|TimeLineHandler"/>
  <os_label>perf</os_label>
  <count>5</count>
</periodictest>
```

schedule weekly or monthly
Infrastructure (1)

• Periodic tests started by the **Jenkins** job
  - nightly builds as input flavour

• Configuration in **XML** files

```xml
<periodictest>
  <schedule type="week" time="10:00">Mon, Tue, Wed, Thu, Fri</schedule>
  <slot>lhcb-future</slot>
  <project>Brunel</project>
  <platform>x86_64-slc6-gcc62-opt</platform>
  <test runner="lhcbrp" group="MiniBrunel" env="lb-run-gaudirun|TimelineHandler"/>
  <os_label>perf</os_label>
  <count>5</count>
</periodictest>
```

- schedule weekly or monthly
- description of options (stored in PRConfig)
Infrastructure (1)

- Periodic tests started by the **Jenkins** job
  - nightly builds as input flavour

- Configuration in **XML** files

```xml
<periodictest>
  <schedule type="week" time="10:00">Mon, Tue, Wed, Thu, Fri</schedule>
  <slot>lhcb-future</slot>
  <project>Brunel</project>
  <platform>x86_64-slc6-gcc62-opt</platform>
  <test runner="lhcbrpr" group="MiniBrunel" env="1b-run-gaudirun|TimeLineHandler"/>
  <os_label>perf</os_label>
  <count>5</count>
</periodictest>
```

- schedule weekly or monthly
- description of options (stored in PRConfig)
- executable and handler (LHCbPR API)
Infrastructure (1)

- Periodic tests started by the **Jenkins** job
  - nightly builds as input flavour

- Configuration in **XML** files
  - https://gitlab.cern.ch/lhcb-core/LHCbNightlyConf/blob/master/test_schedule2.xml

```xml
<periodictest>
  <schedule type="week" time="10:00">Mon, Tue, Wed, Thu, Fri</schedule>
  <slot>lhcb-future</slot>
  <project>Brunel</project>
  <platform>x86_64-slc6-gcc62-opt</platform>
  <test runner="lhcbrpr" group="MiniBrunel" env="lbc-run-gaudirun|TimelLineHandler"/>
  <os_label>perf</os_label>
  <count>5</count>
</periodictest>
```

- schedule weekly or monthly
- description of options (stored in PRConfig)
- executable and handler (LHCbPR API)
- machine label
Infrastructure (2)

- Tests are running on dedicated machines
  - volhcb05 with SLC6
  - lblhcbpr3 with CERN CentOS7

Results of the tests parsed by the specific handlers
- LHCbPR2HD
  - https://gitlab.cern.ch/lhcb-core/LHCbPR2HD
  - to save int, float, string, file and json

Zip file sent to the database through Dirac Storage Element
- LHCbPR2BE
  - https://gitlab.cern.ch/lhcb-core/LHCbPR2BE

Web front–end
- http://lblhcbpr2.cern.ch/
  - LHCbPR2FE
  - https://gitlab.cern.ch/lhcb-core/LHCbPR2FE
Infrastructure (2)

- Tests are running on dedicated machines
  - volhcb05 with SLC6
  - lblhcbpr3 with CERN CentOS7

- Results of the tests parsed by the specific handlers
  - LHCbPR2HD [https://gitlab.cern.ch/lhcb-core/LHCbPR2HD](https://gitlab.cern.ch/lhcb-core/LHCbPR2HD)
  - to save int, float, string, file and json
Infrastructure (2)

• Tests are running on dedicated machines
  ○ volhcb05 with SLC6
  ○ lblhcbpr3 with CERN CentOS7

• Results of the tests parsed by the specific handlers
  ○ LHCbPR2HD https://gitlab.cern.ch/lhcb-core/LHCbPR2HD
  ○ to save int, float, string, file and json

• Zip file sent to the database through Dirac Storage Element
  /lhcb/prdata/zips
  ○ LHCbPR2BE https://gitlab.cern.ch/lhcb-core/LHCbPR2BE
Infrastructure (2)

- Tests are running on dedicated machines
  - volhcb05 with SLC6
  - lblhcbpr3 with CERN CentOS7

- Results of the tests parsed by the specific handlers
  - LHCbPR2HD https://gitlab.cern.ch/lhcb-core/LHCbPR2HD
  - to save int, float, string, file and json

- Zip file sent to the database through Dirac Storage Element
  - /lhcb/prdata/zips
    - LHCbPR2BE https://gitlab.cern.ch/lhcb-core/LHCbPR2BE

- Web front-end http://lblhcbpr2.cern.ch/
  - LHCbPR2FE https://gitlab.cern.ch/lhcb-core/LHCbPR2FE
Latest improvements

• Migration of the job polling the periodic tests to the new Jenkins instance
• Storing logs of test jobs on EOS (/eos/lhcb/storage/lhcbpr/logs/)
Latest improvements

- **Migration** of the job polling the periodic tests to the new Jenkins instance
- Storing logs of test jobs on EOS (/eos/lhcb/storage/lhcbpr/logs/)
- **New dashboard** to monitor how the periodic test jobs finalise (in collaboration with Stefan Chitic)

https://lbnightlies.cern.ch/periodic/
Latest improvements

- Upgrades with **messaging system** (RabbitMQ)
Latest improvements

- Upgrades with **messaging system** (RabbitMQ)
- Users may run tests on demand
- The message is sent to the queue checked by Jenkins job every 10 minutes

`'Start new periodic test' button available after login`
Latest improvements

• Running tests without explicit time scheduling
  
  ◦ nightlies scripts send the message once the builds are done
  ◦ Jenkins job checks periodically the queue whether there are new builds for the defined tests for the given day
  ◦ exact time is ignored
  ◦ no need for waiting
  ◦ build may not be ready on scheduled time
    ◦ if yes, the test is scheduled to run

Maciej Szymański (UCAS)
LHCb computing workshop, 15.05.2017
Latest improvements

• Running tests without explicit time scheduling
  ◦ nightlies scripts send the message once the builds are done

 Maciej Szymański (UCAS)  
 LHCb computing workshop, 15.05.2017
Latest improvements

• Running tests without explicit time scheduling
  ◦ nightlies scripts send the message once the builds are done
  ◦ Jenkins job checks periodically the queue whether there are new builds for the defined tests for the given day
    • exact time is ignored
    • no need for waiting
    • build may not be ready on scheduled time
Latest improvements

• Running tests without explicit time scheduling
  ○ nightlies scripts send the message once the builds are done
  ○ Jenkins job checks periodically the queue whether there are new builds for the defined tests for the given day
    • exact time is ignored
    • no need for waiting
    • build may not be ready on scheduled time
  ○ if yes, the test is scheduled to run
Latest improvements

• Running tests without explicit time scheduling
  ◦ nightlies scripts send the message once the builds are done
  ◦ Jenkins job checks periodically the queue whether there are new builds for the defined tests for the given day
    • exact time is ignored
    • no need for waiting
    • build may not be ready on scheduled time
  ◦ if yes, the test is scheduled to run
  ◦ implemented, testing, not yet in production
Load of volhcb05

- 6 executors
- busy executors
- time span of 24h

Load statistics: volhcb05

Load statistics keep track of four key metrics of resource utilization:

1. **Number of online executors**
   - For a job, if the computer is online then this is the number of executors that the computer has, if the computer is offline then this is zero.
   - For a label, this is the sum of all executors across all online computers in the label.
   - For the entire Jenkins, this is the sum of all executors across all online computers in the Jenkins installation.

2. **Number of busy executors**
   - This line tracks the number of executors (among the executors counted above) that are carrying out builds. The ratio of this to the number of online executors gives you the resource utilization. If all your executors are busy for a prolonged period of time, consider adding more computers to your Jenkins cluster.

3. **Number of available executors**
   - This line tracks the number of executors (among the online executors counted above) that are available to carry out builds. The ratio of this to the total number of executors gives you the resource availability. If none of your executors are available for a prolonged period of time, consider adding more computers to your Jenkins cluster.

4. **Queue length**
   - This is the number of jobs that are in the build queue, waiting for an available executor (of this computer, of this label, or of this Jenkins, respectively). This does not include jobs that are in the quiet period, nor does it include jobs that are in the queue because worker builds are still in progress. If this line ever goes above 0, that means your Jenkins will run more builds by adding more computers.

**Note:** The number of busy executors and the number of available executors are not necessarily equal to the number of online executors as executors can be suspended from accepting builds and thus be neither busy nor available.

The graph is an exponential moving average of periodically collected data values. 3 timespans are updated every 10 seconds, 1 minute and 1 hour respectively.

Maciej Szymański (UCAS) LHCb computing workshop, 15.05.2017
Load of lblhcbpr3

3 executors

busy executors

time span of 24h

Load statistics keep track of four key metrics of resource utilization:

- **Number of online executors**
  - For a computer, if the computer is online then this is the number of executors that the computer has; if the computer is offline then this is zero.
  - For a label, this is the sum of all executors across all online computers in this label.
  - For the entire Jenkins: this is the sum of all executors across all online computers in the Jenkins installation.

- **Number of busy executors**
  - This line tracks the number of executors (among the executors counted above) that are carrying out builds. The ratio of this to the number of online executors gives you the resource utilization. If all your executors are busy for a prolonged period of time, consider adding more compute to your Jenkins cluster.

- **Number of available executors**
  - This line tracks the number of executors (among the online executors counted above) that are available to carry out builds. The ratio of this to the total number of executors gives you the resource availability. If none of your executors are available for a prolonged period of time, consider adding more compute to your Jenkins cluster.

- **Queue length**
  - This is the number of jobs that are in the build queue waiting for an available executor (of this computer, of this label, or in this Jenkins respectively). This doesn’t include jobs that are in the quiet period, nor does it include jobs that are in the queue because worker builds are still in progress. If this line ever goes above 0, that means your Jenkins will run more builds by adding more compute.

**Note:** The number of busy executors and the number of available executors need not necessarily be equal to the number of online executors as executors can be suspended from accepting builds and be neither busy nor available.

The graph is an exponential moving average of periodically collected data values. 3 time spans are updated every 10 seconds, 1 minute and 1 hour respectively.
LHCbPR - current tests

- Brunel, timing information
- MiniBrunel
  - work of Stefan Roiser and Chris Hasse (see Upgrade session on Thursday morning)
  - callgrind, timing, physics performance
LHCbPR - current tests

- Brunel, timing information
- MiniBrunel
  - work of Stefan Roiser and Chris Hasse (see Upgrade session on Thursday morning)
  - callgrind, timing, physics performance
- Moore
  - testing rates of High Level Trigger
LHCbPR - current tests

- Brunel, timing information
- MiniBrunel
  - work of Stefan Roiser and Chris Hasse (see Upgrade session on Thursday morning)
  - callgrind, timing, physics performance
- Moore
  - testing rates of High Level Trigger
- Gauss and Geant4
  - see talk by Tim Williams on 67th A&S week
  - timing the simulation software per detector and particle
  - validation of different build systems
  - early physics validation
  - standalone tests isolating other changes (calorimetry, hadronic cross sections, multiple scattering)
Brunel timing


- time spent by reconstruction algorithm per event as a function of the software version
HLT rate tests

https://lhcbpr-docs.web.cern.ch/lhcbpr-docs/

R.Currie, R.Matev, M.Vesterinen, S.Stahl, C.Fitzpatrick

**LHCb PR**

Results of HLT Tests

### Last 20 tests

<table>
<thead>
<tr>
<th>Date</th>
<th>Test</th>
<th>Slot</th>
<th>build</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-12-11 20:16:53</td>
<td>2016NB_25ns_LOFlt0x160B_Physics_pp_Draft2016_DefHlt1</td>
<td></td>
<td>lhcb-head.1319</td>
</tr>
<tr>
<td>2015-12-10 20:21:01</td>
<td>2016NB_25ns_LOFlt0x160B_Physics_pp_Draft2016_DefHlt1</td>
<td></td>
<td>lhcb-2018-patches 233</td>
</tr>
</tbody>
</table>

#### HLT1ByRegex

<table>
<thead>
<tr>
<th>Regex</th>
<th>Inclusive</th>
<th>Exclusive</th>
</tr>
</thead>
<tbody>
<tr>
<td>HLT1</td>
<td>87.9 ± 1.6 5063.3 90.0</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>30.8 ± 2.3 30.3 30.3</td>
<td>1.0 ± 0.1 1.0 ± 0.1 1.0 ± 0.1 1.0 ± 0.1 1.0 ± 0.1 1.0 ± 0.1 1.0 ± 0.1</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>41.4 ± 2.2 41.4 41.4</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>42.0 ± 2.1 42.0 42.0</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>42.6 ± 2.0 42.6 42.6</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>43.2 ± 1.9 43.2 43.2</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>43.8 ± 1.8 43.8 43.8</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>44.4 ± 1.7 44.4 44.4</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>45.0 ± 1.6 45.0 45.0</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>45.6 ± 1.5 45.6 45.6</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>46.2 ± 1.4 46.2 46.2</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>46.8 ± 1.3 46.8 46.8</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>47.4 ± 1.2 47.4 47.4</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>HLT1&amp;TrackMiss acceleration</td>
<td>48.0 ± 1.1 48.0 48.0</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
</tbody>
</table>

#### HLT2Rates_Totals

<table>
<thead>
<tr>
<th>Line</th>
<th>Incl.</th>
<th>Excl.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>91.7 ± 1.5 6518.0</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>Total_Full</td>
<td>61.7 ± 1.4 4545.0</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>Total_TurboCalib</td>
<td>41.0 ± 1.3 576.7 576.7</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>Total_TurboCalib</td>
<td>32.7 ± 1.1 41.2 41.2</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>Total_TurboCalib</td>
<td>24.3 ± 0.9 30.1 30.1</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
<tr>
<td>Total_TurboCalib</td>
<td>16.0 ± 0.7 26.8 26.8</td>
<td>0.0 ± 0.0 0.0 ± 0.0 0.0 ± 0.0</td>
</tr>
</tbody>
</table>

**Settings**

<table>
<thead>
<tr>
<th>Property Setting</th>
<th>settings Physics_pp_Draft2016</th>
<th>inputdata 2015NB_25ns_LOFlt0x1606</th>
<th>Events 30000</th>
</tr>
</thead>
</table>

**InputStats**

**EventSizes**

<table>
<thead>
<tr>
<th>RB</th>
<th>descr.</th>
<th>Full evts (kB)</th>
<th>Turbo evts (kB)</th>
<th>PR part (kB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RB88 Turbo</td>
<td>0.0 ± 0.0</td>
<td>0.0 ± 0.0</td>
<td>0.0 ± 0.0</td>
<td></td>
</tr>
<tr>
<td>RB46 HLT1</td>
<td>32.4 ± 0.1</td>
<td>32.4 ± 0.1</td>
<td>32.4 ± 0.1</td>
<td></td>
</tr>
<tr>
<td>RB90 TurboCalib</td>
<td>0.0 ± 0.0</td>
<td>0.0 ± 0.0</td>
<td>0.0 ± 0.0</td>
<td></td>
</tr>
<tr>
<td>RB87 Full</td>
<td>0.0 ± 0.0</td>
<td>0.0 ± 0.0</td>
<td>0.0 ± 0.0</td>
<td></td>
</tr>
</tbody>
</table>

- R.Currie, R.Matev, M.Vesterinen, S.Stahl, C.Fitzpatrick
Gauss timing


Total time in each detector volume

- time spent by simulation software per detector
### Gauss timing

Timing of particle groups per detector for Job3351-lhcb-sim09.353-x86_64-slc6-gcc49-opt

<table>
<thead>
<tr>
<th></th>
<th>prs</th>
<th>rich2</th>
<th>muon</th>
<th>velo</th>
<th>rich1</th>
<th>spd</th>
<th>it</th>
<th>converter</th>
<th>magnet</th>
<th>pipe</th>
<th>ot</th>
<th>tt</th>
<th>hcal</th>
<th>ecal</th>
</tr>
</thead>
<tbody>
<tr>
<td>opticalphoton</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1261.410</td>
<td>0.0200</td>
<td>1.8600</td>
<td>441.4000</td>
<td>0.1000</td>
<td>0.0100</td>
<td>0.0600</td>
<td>0.0600</td>
<td>0.0100</td>
<td>0.0200</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mu</td>
<td>0.0200</td>
<td>1.0600</td>
<td>0.4300</td>
<td>0.1000</td>
<td>0.1700</td>
<td>0.0900</td>
<td>0.0900</td>
<td>0.0200</td>
<td>0.0600</td>
<td>0.0100</td>
<td>0.2600</td>
<td>0.1100</td>
<td>0.4100</td>
<td>0.6300</td>
</tr>
<tr>
<td>K</td>
<td>0.1100</td>
<td>0.7500</td>
<td>0.5100</td>
<td>3.8800</td>
<td>0.3000</td>
<td>0.1700</td>
<td>0.4700</td>
<td>0.3600</td>
<td>0.1000</td>
<td>0.4100</td>
<td>0.3900</td>
<td>0.3300</td>
<td>1.5200</td>
<td>4.4400</td>
</tr>
<tr>
<td>e</td>
<td>18.5900</td>
<td>32.2100</td>
<td>35.7000</td>
<td>11.9500</td>
<td>13.7600</td>
<td>4.0400</td>
<td>5.0700</td>
<td>22.8700</td>
<td>18.9500</td>
<td>40.3300</td>
<td>5.9500</td>
<td>5.1200</td>
<td>112.7300</td>
<td>406.8700</td>
</tr>
</tbody>
</table>

- time spent by simulation software per particle and per detector
Geant4 - CMT vs. Cmake


T.Williams

- comparison of physics results for different build systems
Ongoing developments

- Integration tests (Rob Currie)
  - Moore → Tesla → DaVinci

- New analysis modules
  - E.g. replacement of the web-based HLT rate tests monitoring

- Deploying new tests
  - Gauss
  - MiniBrunel

- Continuous feedback from users
Ongoing developments

• Integration tests (Rob Currie)
  ◦ Moore → Tesla → DaVinci

• New analysis modules
  ◦ E.g. replacement of the web-based HLT rate tests monitoring
Ongoing developments

- Integration tests (Rob Currie)
  - Moore → Tesla → DaVinci

- New analysis modules
  - E.g. replacement of the web-based HLT rate tests monitoring

- Deploying new tests
  - Gauss
  - MiniBrunel
Ongoing developments

- Integration tests (Rob Currie)
  - Moore → Tesla → DaVinci

- New analysis modules
  - E.g. replacement of the web-based HLT rate tests monitoring

- Deploying new tests
  - Gauss
  - MiniBrunel

- Continuous feedback from users
LHCbPR - Resources

- Web application:
  https://lblhcbpr2.cern.ch
  https://lblhcbpr2.cern.ch/api/
  https://gitlab.cern.ch/lhcb-core/LHCbPR2FE

- API service:
  https://gitlab.cern.ch/lhcb-core/LHCbPR2BE

- ROOT HTTP service:
  https://gitlab.cern.ch/lhcb-core/LHCbPR2ROOT

- Tests’ output handlers:
  https://gitlab.cern.ch/lhcb-core/LHCbPR2HD

- Project builder:
  https://gitlab.cern.ch/amazurov/LHCbPR2

- Jenkins configuration
  https://gitlab.cern.ch/lhcb-core/LbNightlyTools

- Configuration of the periodic tests
  https://gitlab.cern.ch/lhcb-core/LHCbNightlyConf/
Summary

- Monitoring of the software is **essential tool** in large scientific project
• Monitoring of the software is **essential tool** in large scientific project
• **LHCbPR** has already shown to be the **useful framework**
Summary

- Monitoring of the software is **essential tool** in large scientific project
- LHCbPR has already shown to be the **useful framework**
- **Extensible project** thanks to modular architecture (e.g. new analysis modules)
Summary

- Monitoring of the software is **essential tool** in large scientific project
- **LHCbPR** has already shown to be the **useful framework**
- **Extensible project** thanks to modular architecture (e.g. new analysis modules)
- Feedback is welcome, we are available to discuss new features, deploy new tests etc.