Evolution of the LHCb Continuous Integration system

M. Clemencic, R. Currie, R. Mataev, on behalf of the LHCb collaboration.
Introduction

Within LHCb the Real Time Analysis (RTA) project develops and maintains the real-time processing of LHCb’s data for Run 3 and beyond.

One of the specific goals of this project is to improve and maintain the quality assurance of code being used by the experiment.

An useful tool to contribute to this effort is the development of a Continuous Integration system for the LHCb experiment.

Regularly and reliably testing the code being used improves the experience for everyone.
# LHCb Software Stack

<table>
<thead>
<tr>
<th>Application</th>
<th>Library</th>
<th>Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss</td>
<td>Boole</td>
<td>Brunel</td>
</tr>
<tr>
<td>Bender</td>
<td>Erasmus</td>
<td>Panoptix</td>
</tr>
<tr>
<td>DaVinci</td>
<td>Phys</td>
<td>Moore</td>
</tr>
<tr>
<td>LbCom</td>
<td>LHCb</td>
<td>Gaudi</td>
</tr>
</tbody>
</table>

**Project Dependency**

**Evolution of the LHCb Continuous Integration system**

Robert Currie, Edinburgh University
LHCb Nightly Build System

The LHCb computing infrastructure is a system comprised of \( O(300) \) cores running jobs managed by Jenkins (http://jenkins.io).

It is used to check the current build and test status of LHCb software against external dependencies on a variety of different platforms.

Previous CHEP presentations on the LHCb Build System:

2009: The nightly build and test system for LCG AA and LHCb software
2013: A New Nightly Build System for LHCb
Overview of the LHCb Nightly Build System

Existing LHCb nightly build system automatically builds the whole software stack each night.

Previous solutions to integrate with the development workflow merged all pending changes into the LHCb build stack each night.

Building and testing this makes it possible to determine how multiple changes across the stack impact each other.

However, this also makes it difficult to determine which MR introduced which change(s).
Overview of the LHCb Nightly Build System
Continuous Integration for LHCb

To enhance the CI System for LHCb we want to be able to do the following:

• Automatically build the whole LHCb stack integrating current isolated changes from a MR

• Automatically run tests on the result of this build

• Understand the impact of a given MR on the wider experiment

• Evaluate the quality of these changes
Isolated Testing of an MR

Isolated testing of an MR allows us to cleanly evaluate the impact of a given MR on external projects.

If the work being done is split across multiple projects (multiple MRs) we must also be able to reliably test these together.

Important tests for LHCb are built into the application layer.

Due to this the only way to properly evaluate the impact changes have on these is to build and test the whole software stack.
Isolated Testing of an MR (cont)

To understand the impact of code changes we build and test the LHCb stack at the base and tip of the MR.

- **ref** Build from base of MR
- **mr** Build from tip of the MR
CI Summaries

Screenshot from triggering a webhook with gitlab and final user feedback:

Gerhard Raven @graven · 1 day ago
/ci-test --merge

Lhcb Software @lhbsoft · 1 day ago
Started reference and integration test builds. Once done, check the comparison of build and test results.

- [2019-10-26 17:46] Validation started with lhcb-master-mr#27

Edited by Lhcb Software 1 day ago
Workflow Integration (cont)

A more advanced use of this command might be the following:

```
/ci-test LHCb!123 Rec!234 Brunel!456 Gaudi@someshal
platform=x86_64 ...
```

Perform a full build of the LHCb stack using the following:

- MR 123 from the LHCb project
- MR 456 from the Brunel project
- using commit-ID “someshal” from Gaudi
- for the x86_64 platform
Nightly Build Dashboard

This nightly build system boasts a web feedback system built atop Apache CouchDB:
https://lhcb-nightlies.cern.ch/nightly/summary/
CI Feedback

A comparison between the ref and mr builds:

<table>
<thead>
<tr>
<th>Project</th>
<th>Version</th>
<th>x86_64-centos7-gcc8-opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaudi</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Online</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>LHCb</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Ibcom</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Rec</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Brunel</td>
<td>master</td>
<td>8</td>
</tr>
<tr>
<td>Phys</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Hit</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Moore</td>
<td>master</td>
<td>17</td>
</tr>
<tr>
<td>MooreOnline</td>
<td>master</td>
<td>1</td>
</tr>
<tr>
<td>Stripping</td>
<td>master</td>
<td>246</td>
</tr>
<tr>
<td>Analysis</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>DaVinci</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Boole</td>
<td>master</td>
<td>OK</td>
</tr>
<tr>
<td>Alignment</td>
<td>master</td>
<td>20</td>
</tr>
<tr>
<td>AlignmentOnline</td>
<td>master</td>
<td>1</td>
</tr>
<tr>
<td>LHCbIntegrationTests</td>
<td>master</td>
<td>3</td>
</tr>
<tr>
<td>Bender</td>
<td>master</td>
<td>OK</td>
</tr>
</tbody>
</table>

Note: The numbers in parentheses indicate the number of failures or warnings.
Changes to developers workflow

Old workflow:

- MR developed and tagged as ready to merge
- MR included in nightly build
- Experts work out which MR impacted tests
- Developer informed of changes needed

(>= 1 day start to finish)

New workflow:

- Developer triggers manual build/test
- Developer fixes external issues

(≈ few hr)
Future Work

Additional enhancements to the nightly build infrastructure will also improve building and testing of individual MRs:

• Caching of build artefacts

• Faster feedback to developers

• Improved feedback for manual builds
Summary

• LHCb nightly build system has proven very flexible

• Isolated testing of MRs against the LHCb build stack has been demonstrated to work

• Work is ongoing to improve the experience

• Using this will improve the software management experience for everyone on LHCb
Thanks for Listening!