Testing and Deployment Strategies

Their Impact on Accelerator Performance

Jean-Christophe Garnier
TE-MPE-MS

Acknowledgement to numerous colleagues for their help and inputs

Introduction

- Software from Accelerator Control Systems: CO, OP, RF, MPE, BI, EPC, etc.
- High level software, from FESA to User Interfaces, Servers, Analysis Tools, etc.
  - Java, C++ and Python
- How much do we spend on tests, validations and rollbacks?
- How confident are we in our software stack and environment?
- More than performance impact, what is the risk for the Machine itself?
- Only preliminary answers and many more questions

Outline

1. Review of test and deployment impact on LHC performance
2. Review of testing strategies
3. Review of deployment strategies

Outline

1. Review of test and deployment impact on LHC performance
2. Review of testing strategies
3. Review of deployment strategies

Numerous sources of information

- With help from
  - elogbook
  - logs.cern.ch
  - AFT
  - Exploitation meetings
  - Smooth Upgrades Working Group
  - Post Mortem
- Many details in numerous places - could they be correlated easily? automatically?

Smooth Upgrades Working Group

- Coordinates upgrades
- For each, verifies
  - Risks for operations
  - How validation tests can be performed
  - How it can be rolled back
  - Service dependencies and their impact
- Combines knowledge from numerous experts, with lots of work, but couldn't prevent some issues
  - Delay in powering due to deployment of new RBAC rules (TS1 2015)
  - 5 days of system failure, didn't block operation but made it slower (TS1 2015)
- Could it benefit from some software support?
  - Many dependencies between control system components
  - Could a service expose them?

Qualitative Evaluation from Logbook 2016

- Hints on inefficiency
  - “couple of hours (...) to the new software deployment”
  - “Problem in sector 2-3 after (???) redeployment”
  - “FESA server rolled back to version released in Aug 2015. -> Seems to work better”
  - ...
- Main causes
  - Lack of tests and validations before deployment, or validation when beams were wanted
  - Impact on dependent systems was underestimated
    - because relations are not easy to know

Outline

1. Review of test and deployment impact on LHC performance
2. Review of testing strategies
3. Review of deployment strategies

Testing Practices

- Testing is performed at different level
- Sample, from lower to higher
  - Unit Testing: a small unit of code in isolation
  - Integration Testing: multiple components together
  - User Acceptance Testing: the specifications are fulfilled
  - Staging: validate in environment as close as possible to production
  - Validation in production: requires less time if other strategies are applied
- Initial investment cost, quickly recovered through automation compared to manual testing in production
- Ease maintenance and refactoring of legacy code

Unit and Integration Testing Review

- Must be considered at design time
  - All already written software is not unit testable
  - All software can be written in a testable way
- Python, C++, Java all have multiple unit test frameworks
- Discrepancy in testability depending on core tools we use
  - FESA: Generally difficult to unit test, very complicated to isolate logic from the framework
  - japc-ext-mockito: Simplifies testing of Java clients

- Core tools should take testability of user applications into account
- How do we know if the tests are correct or effective?

Test Quality Review

- Code Coverage: % of source code executed by the tests
- 365/654 Java projects report less than 50% coverage in SonarQube
  - Is it comfortable?
  - What about C++ and Python?

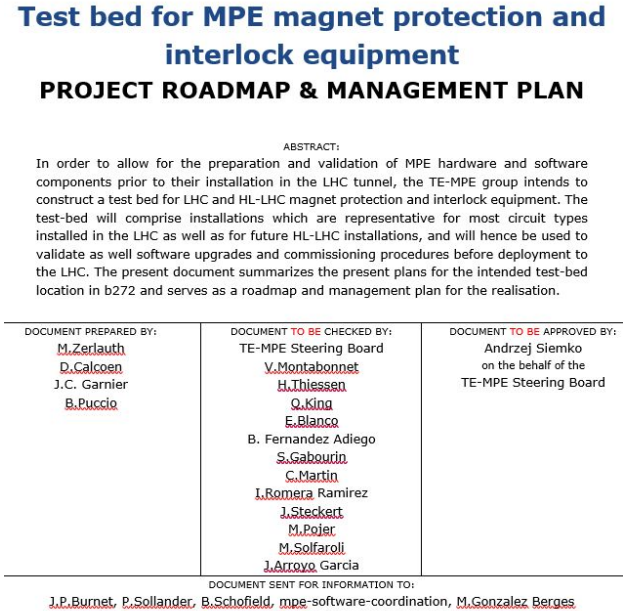
- Centralized place to publish metrics?
  - Infrastructure is present:
    - Bamboo: builds and publishes metrics
    - SonarQube: aggregates and reports
  - Why not being transparent?
    - User confidence
    - Encourage quality

Staging/User Acceptance Test Review

- Close to real environment, to run tests at any time
- Simulations can replace unavailable hardware/software components
- Must be considered at design time
- Automated deployments and automated User Acceptance Tests to validate software product
- Examples:
  - QPS Swiss Tool: Assert all features on testbed hardware
  - Orbit Feedback: Automated test without beams before deployments
    - Helped understanding legacy software
    - Helped reducing testing time in operation, performed by operators and experts

Staging in Complete Testbeds

- Complete testbeds
  - For CMW, Timing, and FESA
  - For FGCs
  - Full magnet protection and interlock system - without magnet
- Challenge: staging performed in Technical Network
  - Could it be clearly separated from operational environment?
  - Risk of errors/mistakes - RBAC should protect us



Test bed for MPE magnet protection and interlock equipment

PROJECT ROADMAP & MANAGEMENT PLAN

ABSTRACT:

In order to allow for the preparation and validation of MPE hardware and software components prior to their installation in the LHC tunnel, the TE-MPE group intends to construct a test bed for LHC and HL-LHC magnet protection and interlock equipment. The test-bed will comprise installations which are representative for most circuit types installed in the LHC as well as for future HL-LHC installations, and will hence be used to validate as well software upgrades and commissioning procedures before deployment to the LHC. The present document summarizes the present plans for the intended test-bed location in b272 and serves as a roadmap and management plan for the realisation.

DOCUMENT PREPARED BY:	DOCUMENT TO BE CHECKED BY:	DOCUMENT TO BE APPROVED BY:
M.Zerlauth D.Calcoen J.C. Garnier B.Puccio	TE-MPE Steering Board V.Montabonnet H.Thiessen Q.King E.Blanco B. Fernandez Adiego S.Gabourin C.Martin I.Romera Ramirez J.Steckert M.Pojer M.Solfaroli J.Arroyo Garcia	Andrzej Siemko on the behalf of the TE-MPE Steering Board

DOCUMENT SENT FOR INFORMATION TO:

J.P.Burnet, P.Sollander, B.Schofield, mpe-software-coordination, M.Gonzalez Berges

Outline

1. Review of test and deployment impact on LHC performance
2. Review of testing strategies
3. Review of deployment strategies

Deployment Practices

- GUIs: quite good with PREVIOUS/PRO/NEXT weblinks
- Server deployment
  - Developer must log in operational server/FECs and elevate his/her privileges
  - Usually run 1 command
- Server rollback
  - Developer modifies some paths on the operational server to clean the previous deployment
  - With elevated privileges
- Imagine rolling back multi-tier architecture
- What about a fully featured deployment service?

Requirements for a Deployment Service

- Safe and easy deployment
  - With scheduling, the deployment is pending until enabled by:
    - Operator
    - State of the Machine
- Safe and easy rollback, from operators themselves
- Tracking
  - Software Version
  - Dependencies between Software Services
- Is there such a tool already?
  - Some components are already in place: Bamboo + Ansible

Recommendations

- Further analyse impact of testing and deployment on availability
  - Including the entire software stack, with LabVIEW and WinCC OA
- Promote testing and establish common best practices
  - Considering testing at design time
  - Reduce beam time devoted to tests
  - Extend Code Quality Management with SonarQube to all operational languages
- Invest in a fully featured Deployment service to increase safety and tracking of deployments
- Many testbed initiatives, would it make sense to integrate them together?
- Should all this be followed up by a task force from CO3?