Improved software production for the LHC tunnel cryogenics control system

Czesław Fluder¹, Tomasz Wolak², Adam Drozd², Michal Dudek¹, Francesco Frassinelli³, Marco Pezzetti³, Antonio Tovar-Gonzalez¹, Marcin Zapolski²

1. CERN, CH-1211 Geneva 23, Switzerland
2. AGH UST, 30 Mickiewicza Av, 30-059 Krakow, Poland
3. Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milan, Italy

Abstract

- The software development for the control system of the cryogenics in the LHC is partially automated. However, every single modification requires a sequence of consecutive and interdependent tasks to be executed manually by software developers. A large number of control system consolidations and evolution of used IT technologies lead to reviewing the software production methodology. As a result, an open-source continuous integration server has been employed integrating all development tasks, tools and technologies in a predefined time and event triggers. This paper describes the main improvements that have been made to fully automate the process of software production and achieved results.

Cryogenics in the LHC tunnel

- The architecture of Continuous Integration system for control system software development.

The control system

- The solution allowed to produce and to successfully deploy control software in all sectors of the cryogenics systems of the LHC tunnel and also to implement all requested last-minute changes. Until now two sectors are being cooled while others are operational, in recommissioning phase.

- Continuous Integration practice, including automated builds and tests, is used successfully in software engineering since many years. Discussed developments, experiences and results of applying this approach to improve process of producing control system software for cryogenics in the LHC tunnel proved that the methodology can be equally useful in the field of industrial automation. It allows significantly optimize development process and (with help of hardware simulation devices) raise quality of the software produced for large-scale control systems.

Conclusions

- The LHC Layout DB
- PLC Hardware
- Source code for PLC
- Static components (UCPC and CRG)
- Build PLC project
- PLC Project
- Control system deployment in testing environment
- Release and deployment in production environment

Software development process

- Data consistency check
- Database import
- SCADA Project
- Build PLC project
- PLC Project
- Deployment
- Testing
- Release and deployment in production environment

Development process with Continuous Integration service

- Generate control system configuration
- Control system configuration generation
- Continuous Integration service nodes
- Build PLC projects
- Version control (SVN)
- SCADA database preparation and deployment
- Execute job
- Get report
- Execute job
- Get results or artifacts
- Execute job
- Get results or artifacts
- Build servers SLC6 (Linux)
- Continuous Integration server CRG (lenovo)
- SCADA database importation and specification
- Source code generation
- SCADA Project
- Data validation rules
- Data validation
- Control system object specification verified
- Verify code templates
- Control system object specification
- Generate object specification
- Build servers ICL2 (Windows)
- Generate hardware conf.
- Check data consistency
- Generate source code
- Continuous Integration framework
- Cygwin, Python 2.6
- 8 virtual machines with 1 job slot (SIMATIC limitations)

Development of software

- Build PLC project
- Version control
- Check system response
- Test system
- Continuous Integration service CRG (lenovo)
- Issue tracking code reviews
- Build servers ICL2 (Windows)
- Continuous Integration server CRG (lenovo)
- Build servers SLC6 (Linux)
- Continuous Integration framework
- Cygwin, Python 2.6
- 8 virtual machines with 1 job slot (SIMATIC limitations)

Presented at the 25th International Cryogenic Engineering Conference & International Cryogenic Materials Conference 2014

Technology Department, CERN, 1211 Gineve 23, Switzerland