BRINGING QUALITY IN THE CONTROLS SOFTWARE DELIVERY PROCESS


Introduction

Alba is a 3rd generation synchrotron located near Barcelona, Spain. It comprises accelerators and 7 beamlines and was successfully commissioned in 2012. Nowadays all its beamlines host user experiments regularly. At the same time, more beamlines are under construction and it is planned to expand the facility even more in the near future.

The Alba Controls Section develops and operates a diverse variety of controls software which is shared within international communities of users and developers. This includes: generic frameworks like Sardana and Taurus, numerous Tango device servers and applications where, among others, we can find PyAlarm and Panic, and specific experiment procedures and hardware controllers. A study has commenced on how to improve the delivery process of our software from the hands of developers to laboratories, by making this process more reliable, predictable and risk-controlled.

Code control

- Code repositories were migrated from SVN to GIT
- Easier branching and merging
- Easier tools and workflows
- Distributed architecture
- Better performance
- Cleaner history of commits with less effort

**SEMANTIC VERSIONING**

Given a version number MAJOR.MINOR.PATCH, increment the:

1. **MAJOR** version when you make incompatible API changes
2. **MINOR** version when you add functionality in a backwards-compatible manner
3. **PATCH** version when you make backwards-compatible bug fixes

Packaging

- Python packaging libraries e.g. distutils builds rpm, deb, msi packages
- msi allows unattended installations necessary for SCM
- Package repositories (repo) e.g. yum for rpm, needs to be setup
- Proof-of-concept Taurus CD pipeline: builds rpm and msi, uploads them to the repo and SCM pulls from the repo

Testing

- SEPs established the common testing strategy for Sardana and Taurus.
- Tests should be written before developers start work on the features that they test.
- The automated test suite should be run by the Continuous Integration service on every commit.

**Jenkins**

- Mid or legacy projects should start automating the most common and important use cases.
- The rest of the scenarios should initially be tested manually.

Continuous Delivery

- Agile & Continuous Delivery (CD) aims to transform a concept into a working software as fast as possible.
- CD is based on fully automated, reliable, repeatable and constantly improving software delivery pipelines.
- Start with pipelines with just 3 stages: commit, acceptance and user acceptance (UAT).
- Jenkins works as the pipeline orchestrator.
- **Commit**: unit tests, code analysis, build packages
- **Acceptance** deploy with SCM to production-like environment, execute automated acceptance tests
- **Use Docker** to prepare & run lightweight, reliable and isolated acceptance test environments
- **UAT** (on demand): deploy with SCM to production-like environment, execute manual acceptance tests

Configuration Management

- Bliss system (by ESRF) was compared with Software Configuration Management (SCM) tools in terms of automated packaging and deployments
- Bliss pros: intuitive for non-packaging experts, offline access to hosts' configurations
- Bliss cons: restricted to bliss users only, poor performance
- Bliss cons: restricted to bliss users only, poor packaging libraries e.g. yum
- Sourceforge
- Better performance
- Single repository (repo) e.g. yum for operating system agnostic configurations
- Supported Windows, easy integration Continuous delivery tools, high scalability
- New SCM cons: require new packaging from projects, and setup of repos

Continuous documentation

- Sardana and Taurus migrated their documentation to Read The Docs (RTD)
- Documentation is build on every commit
- Maintenance of the servers and the necessary software is outsourced to RTD
- Several versions of the docs e.g. stable or latest are available
- Docs are available in different formats: html, pdf, epub

Next steps

- Sardana and Taurus projects could already apply the CD strategy to their biannual releases.
- Ideally their pipelines should be accessible by the whole community of developers.
- Try online providers for the continuous integration/delivery tools.
- Try online code review platforms. It may bring new quality to the current review processes, making them more accessible to the developers and reducing the workload on the integration managers.

Acknowledgement

- Alba Controls Section and especially: R. Homs, D. Roldan, J. Andreu, F. Becheri and S. Blanch
- Sardana and Taurus community members
- Alba IT Systems Section and especially to S. Puos, M. Rodríguez, M. Díaz and A. Pérez
- Alba MIS Section and especially to D. Salvat