Contribution ID: **151** Type: **Talk (15min + 5min)**

Advantages and Challenges of a Gitlab CI/CD Pipeline Architecture for the Build and Release System of a Multi-Project Satellite Simulation Software

Tuesday 25 February 2025 14:30 (20 minutes)

The aim of this presentation is to demonstrate the benefits and constraints of using a Continuous Integration and Continuous Development (CI/CD) for the testing, documentation, build and release of numerous Gitlab projects (science modules) as well as the desktop GUI application for the "Modelling software for quantum sensors in space" (MoQSpace) project at DLR (German Aerospace Centre) Institute of Satellite Geodesy and Inertial Sensing. The objective of MoQSpace is to provide a satellite simulation toolchain for the advanced orbital propagation and test mass dynamics of novel sensors with a software library of science modules developed in various programming languages with many contributions from different authors over the past two decades.

With such diverse and large legacy research code, it was crucial to plan standardized workflows to develop, test, document and publish modules with a module versioning and dependency management system. Consequently, the modules were grouped in packages such that the package constituents are inter-compatible for a given development environment. The defined structure and development processes accelerate the development of new additions to the library. The users can view which modules can be selected and downloaded together in a desktop GUI application called VENQS that is developed in Python. The CI/CD in Gitlab is a powerful tool for the aforementioned facets of the project but involves several challenges.

Firstly, each module is a separate Gitlab project with parallel releases in multiple software versions and OS versions. It was essential to automate the process with build scripts at a single source of truth for traceability and agile development. Hence, a separate project in the parent Gitlab Group was created with modular CI/CD scripts that are imported to all the modules. But the re-usability of the code for parallel jobs for varying metadata variables is limited to the 'parallel'keyword that cannot be mapped one-on-one to the corresponding jobs in the next stage in the CI/CD pipeline with the existing methods.

Secondly, two release architectures were needed. In the down-to-top scenario, modules are built and released individually. When a new package of modules is published in the VENQS App, it fetches the published releases from the Package Registry of each module. In the top-to-down approach, for an existing package, all modules need to be re-built in a new version, e.g. for a new OS system. This requires one parent CI/CD script that triggers all the module pipelines to re-run to add a release for an existing package in the Package registry. The existing tools in Gitlab can achieve this but require a well-planned workflow to optimize the traceability and parallel pipelines across multiple Gitlab projects based on the project requirements.
Concluding, an objective in this talk is to shed light on the multi-faceted benefits we achieved in automating the build and release of research software in Gitlab CI/CD. Moreover, the lessons learned with the challenges in the design and implementation will be discussed with the hope to generate a fruitful discussion.

I want to participate in the youngRSE prize

no

Primary author: CHAND, Suditi (DLR - Institute for Satellite Geodesy and Inertial Sensing)

Co-authors: PFANNENSTIEL, Mike (DLR - Institute for Satellite Geodesy and Inertial Sensing); Dr BREMER, Stefanie (DLR - Institute for Satellite Geodesy and Inertial Sensing)

Presenter: CHAND, Suditi (DLR - Institute for Satellite Geodesy and Inertial Sensing)

Session Classification: Facets of large Software Infrastructures

Track Classification: Research Software: Continuous Integration