AstroCal Configuration Management Plan

AstroCal
October 6, 2022

Jake LaJeunesse
Mitchell Vivian

1. INTRODUCTION

The AstroCal project uses a variety of standards and tools to maintain configuration control. Git, pull requests through GitHub, and peer peer reviewing are all part of this process. Changes and change requests are documented and stored in AstroCal's Jira or Confluence.

1.1 Purpose

This document will provide a detailed explanation of how members of the AstroCal project manage configuration. Software engineers, product owners, and other team members may refer to the software Configuration Plan to gain an understanding of the process of configuration and change management in the AstroCal project.

Configuration of Git through Github will be covered.

Baselines, mainlines, branches, merging, and versioning will all be covered in this document. Git is used for source control, and GitHub is used as a Git repository. Our most up-to-date baseline is on the "main" branch. Our mainline is preserved in the commit history of the main branch, previous versions of the software are maintained there.

The process of requesting and integrating changes to the software are covered as well. Changes are requested through communication between the client, product owner, and developers.

1.2 Scope

This document defines Configuration Management (CM) activities for all Software and Data produced during the development of the AstroCal (AC) software. This document applies to all module products, end-user products, and data developed and maintained for the AC Program.

1.3 Definitions and Acronyms

1.3.1 Key acronyms

AC -	AstroCal
CM -	Configuration Management
PR -	Pull Request
SCM -	Software Configuration Manager
SE -	Software Engineer

2. SCM ACTIVITIES

2.1 Configuration Identification

AstroCal is currently in development, product baseline to be approved upon delivery.

Product baselines are to be maintained by the AC SCM. The Baseline will consist of the following:

- Source files
- Libraries
- Executable files
- Data files
- User's Manuals

2.1.1 Specification Identification

The astrocal repository currently contains the following items:

- Python source code contained in AstroCal/
- Required Python packages contained in requirements.txt
- Unit tests in tests/
- An SQLite DB containing location data
- User manuals
- VSCode launch and debug configuration
- Product license

2.1.2 Change Control Form Identification

Each significant change to the baseline of the product must be accompanied with a Change Control Form. The change control form must contain the following information:

- Change Requester
- Outline of Requested Change, including description and reason/benefits of change
- The name of the person analyzing the change once submitted
- The date of analysis
- The affected features
- The affected components
- An assessment of the change
    - Priority
    - Estimation of:
        - effort impact
        - budget impact
        - schedule impact
        - business impact

        - how the change is to be implemented
        - date of assessment
- The decision of the development team on the status of the change request
- The date this decision was made
- The version the change was requested to be made to.
- Any comments about the change from the team or the requestor.

A template of the required information is on the development Confluence site and can be provided upon request.

2.1.3 Library

This project uses Git hosted on Github for version repository management purposes. Restrictions are imposed on baseline branches to enforce control management processes.

2.2 Configuration Control

2.2.1 Procedures for changing baseline

Changes to source files are to be performed on branches created from the ‘dev’ branch. These branches are to follow the naming convention outlined in the AstroCal confluence.

At the end of each iteration cycle the baseline is updated. Unit tests are run before proceeding. A review is conducted by the product owners, configuration manager, and optionally a small team of developers to ensure the iteration meets the expectations set at the beginning of the sprint. Afterwards, the dev branch is merged with the main branch.

Major changes in the iteration are stored in the release notes. Additionally, source file changes are tracked through the commit log in Git.

Previous baseline versions can be retrieved through the mainline stored on the main branch.

2.2.2 Procedures for processing change requests and approvals

Change requests are processed by the product owners. Approved changes will be posted to AstroCal’s Jira backlog as user stories or tasks. Once introduced to the backlog, these items will be estimated and prioritized before introduction to a future sprint.

Rarely, urgent changes may be approved to be introduced to the current sprint as ‘interloper’ tasks.

2.2.3 Level of control

Management of changes to the software baseline are done by the Client in consultation with the development team. Changes are discussed and approved based on the teams' ability to complete the requested change and how the change affects the overall project scope.

Evaluation of the required changes are done by the development team in consultation with the product owner and a decision is made as to the change being approved or declined.

If a change has been approved, a task will be created in the product backlog to implement that change and will be moved into the sprint backlog when able / necessary.

2.2.4 Storage, handling and release of project media

All project code and media is stored on our distributed version control system, Git. VS Code is used to manage version control and develop software for the project.

2.2.5 Information and Control

2.2.6 Release process

At the end of each sprint, a functional prototype is shown to the client and with approval, the development branch is merged with main to complete that sprint's iteration of the product. The commit history of the main branch contains the mainline history.

2.3 Configuration Audits and Reviews

Pull requests via Git are how the project manages the project files and configuration. Each project branch is reviewed by several members of the development team prior to being merged into the development baseline. Once an iteration is completed, the development baseline is merged into the main branch for release.

3. Training

Hosted on our development Confluence site, there are some tutorials on how to make pull requests, a Change Request template and supplementary materials created by our development team to complete common tasks such as setting up your development environment and how to properly complete a peer review.