LArSoft GitHub / pull request testing in preparation for migration

Erica Snider
Lynn Garren
Patrick Gartung
Kyle Knoepfel
Fermilab

Nov 5, 2019

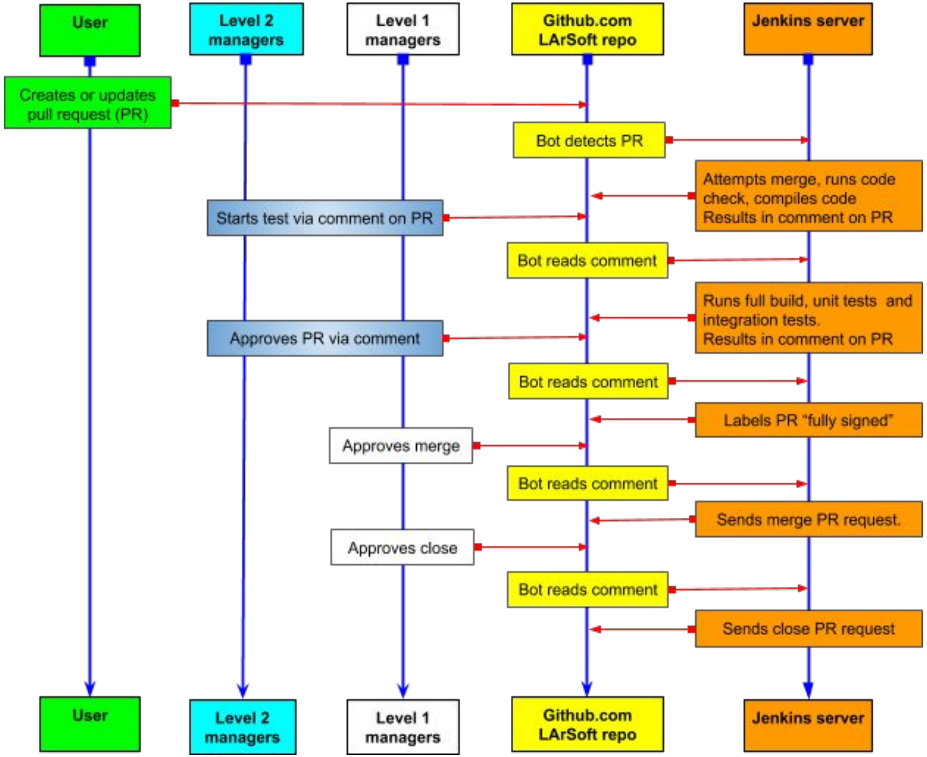
GitHub migration

- Moving all LArSoft repositories to GitHub **before the end of the year**
- Starting at that time, code integration will proceed via pull-requests
  - Each request triggers a set of automated tests, review by a set of managers
  - Mergers to develop after approvals
- Technical changes for system completed, documentation drafted
- Starting testing phase for pull-request workflow now

This is a kick-off of the pull-request testing

The pull-request system overview

- Users push commits to personal GitHub account, then open PR
- “Level 2” managers from relevant experiment reviews, launches tests
- Level 2 manager approves merge if test results as expected (including for other experiments)
- Level 1 manager merges, unless a breaking change
  - Breaking changes integrated during release creation process, as is done now



User

Level 2 managers

Level 1 managers

Github.com LArSoft repo

Jenkins server

Creates or updates pull request (PR)

Bot detects PR

Attempts merge, runs code check, compiles code Results in comment on PR

Starts test via comment on PR

Bot reads comment

Runs full build, unit tests and integration tests. Results in comment on PR

Approves PR via comment

Bot reads comment

Labels PR "fully signed"

Approves merge

Bot reads comment

Sends merge PR request.

Approves close

Bot reads comment

Sends close PR request

User

Level 2 managers

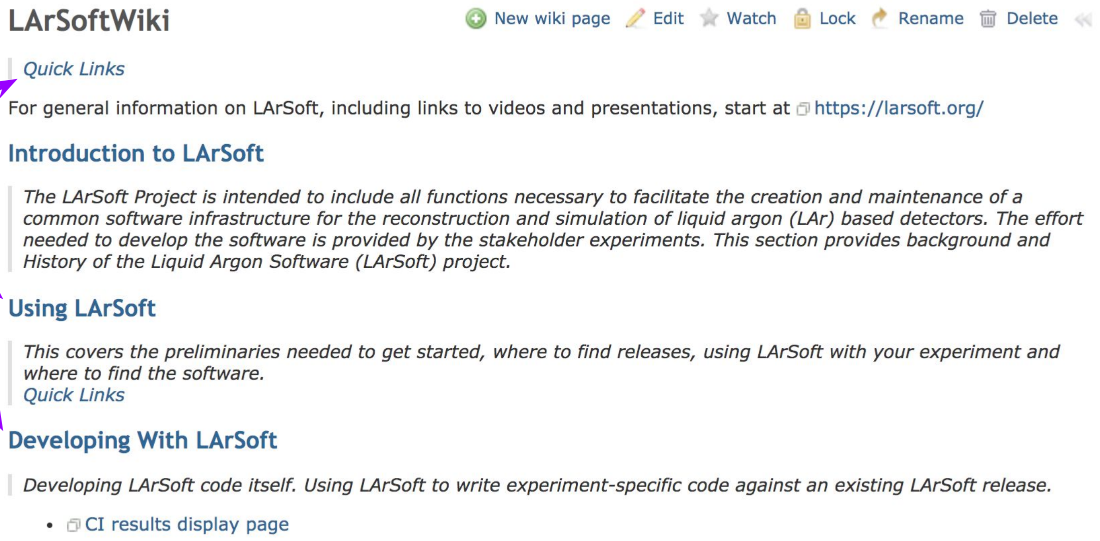
Level 1 managers

Github.com LArSoft repo

Jenkins server

Pull-request system documentation

Can find all documentation linked from “Quick Links”, “Using LArSoft” and “Developing with LArSoft” on LArSoft wiki



LArSoftWiki

Quick Links

For general information on LArSoft, including links to videos and presentations, start at https://larsoft.org/

Introduction to LArSoft

The LArSoft Project is intended to include all functions necessary to facilitate the creation and maintenance of a common software infrastructure for the reconstruction and simulation of liquid argon (LAr) based detectors. The effort needed to develop the software is provided by the stakeholder experiments. This section provides background and History of the Liquid Argon Software (LArSoft) project.

Using LArSoft

This covers the preliminaries needed to get started, where to find releases, using LArSoft with your experiment and where to find the software.
Quick Links

Developing With LArSoft

Developing LArSoft code itself. Using LArSoft to write experiment-specific code against an existing LArSoft release.

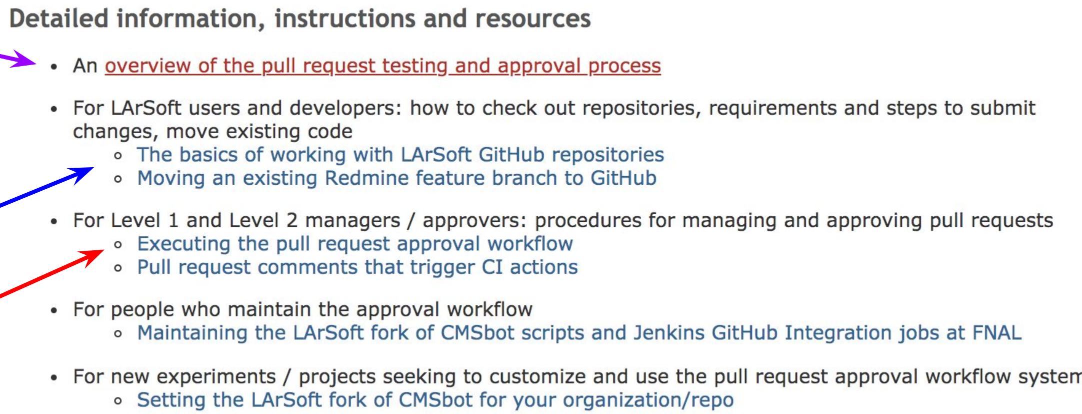
- CI results display page

Pull-request system documentation

Key documents

- Overview of testing and approval process
- For users
  - Basics of working with GitHub
  - Moving existing (Redmine) feature branches to GitHub
- For Level 2 managers
  - Instructions for executing the pull request approval workflow

[Working with GitHub overview page]



Detailed information, instructions and resources

- An overview of the pull request testing and approval process
- For LArSoft users and developers: how to check out repositories, requirements and steps to submit changes, move existing code
  - The basics of working with LArSoft GitHub repositories
  - Moving an existing Redmine feature branch to GitHub
- For Level 1 and Level 2 managers / approvers: procedures for managing and approving pull requests
  - Executing the pull request approval workflow
  - Pull request comments that trigger CI actions
- For people who maintain the approval workflow
  - Maintaining the LArSoft fork of CMSbot scripts and Jenkins GitHub Integration jobs at FNAL
- For new experiments / projects seeking to customize and use the pull request approval workflow system
  - Setting the LArsoft fork of CMSbot for your organization/repo

Important changes for users

- Will need forks of LArSoft repositories in personal GitHub account
  - Can use the “hub” utility to assist with creating forks
  - Pay attention to details of configuring the GitHub account
- If working on a feature branch that existed prior to migration
  - Will need to run a script to truncate history prior to pushing to LArSoft fork on GitHub, or making a pull-request
- Under no circumstances will users be able to merge directly to “develop” branch of central LArSoft repositories
  - Intervention by Level 2 managers to needed for this

What will not change for users

- Checking out code from GitHub
  - Uses mrb, as before
  - Just need to point it to GitHub

```
mrb gitCheckout --repo-type github …
```

Level 2 managers

- Responsible for executing testing and review workflow
  - Initially the experiment offline leads, release managers + any others designated by experiments
  - Review code relevant to their experiment

- Workflow tasks include
  - Verifying PR submitter is authorized
  - Review that code complies with standards, plans
    - Experiment + as agreed with LArSoft
    - Some standards checking can be automated
  - Check results of CI system tests via CI results webpages
    - Build, unit tests, integration tests
  - Approve for merge, or refer back to author for corrective actions

Level 1 managers

- Responsible executing merge operations for approved PRs
  - Currently the LArSoft code and release managers, SciSoft team leaders + other designees

- Workflow tasks
  - Final reviews of code to be merged
  - Schedule merge

Detailed instructions for Level 1 and Level 2 managers on “Executing the pull request approval workflow” page

Testing the system

- We need feedback on using the system from both user and manager perspectives

- What can I do?
  - Please exercise the system
    - Can use arbitrary commits
    - The existing GitHub repositories will be replaced during official migration
  - Take note of procedures, documentation
  - Pay attention to notifications
    - Currently concerned that email notification scheme will be hard to use
  - Send comments on anything / everything

- Will keep testing open for at least two weeks

The end