# CI: Testing and validation of production software

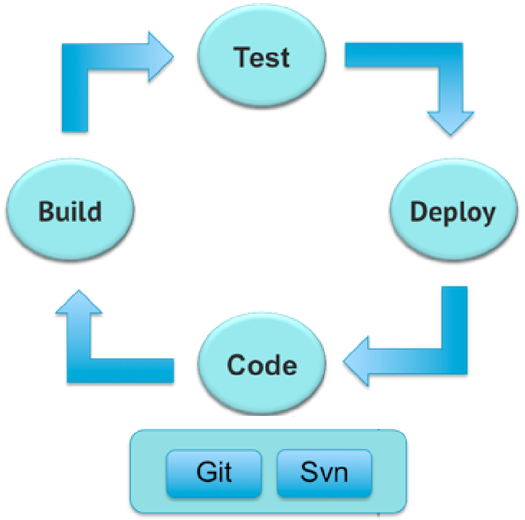
Vito Di Benedetto for CI Project Team

FIFE Workshop

21st-22nd June 2017

## Introduction: Continuous Integration (CI)

- Continuous integration is a software engineering practice in which changes in a software code are immediately tested and reported
- The goal is to provide rapid feedback helping identifying defects introduced by code changes as soon as possible.
- Issues detected early on in development are typically smaller, less complex and easier to resolve.
- Each "commit" is verified by an automated build procedure that tests the code and allows teams to detect problems early, hopefully before the code goes in production.



## Introduction: why Continuous Integration

- Bad habits in code development
  can break your code...
  …or someone else's code!

## Introduction: why Continuous Integration

- Sometime also good practice in code development can lead to some hidden bug...

## Introduction: why Continuous Integration

- The more code you write without testing, the more paths you have to check for errors.
  - Keep on a straight path with proper code testing.

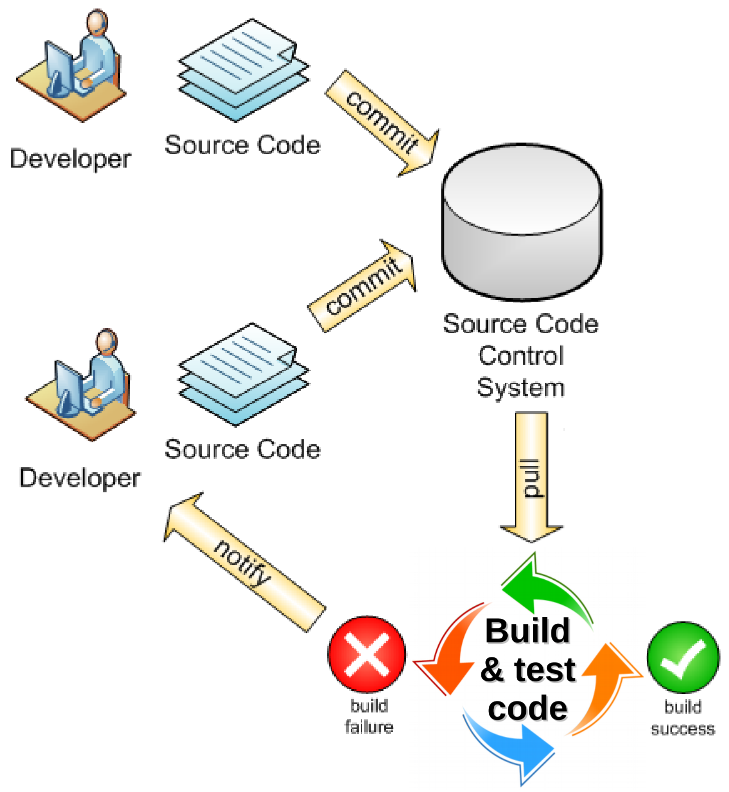
# The CI Project

- The aim of the CI Project is to improve the existing tools and extend the CI service to IF experiments;
- Continuous Integration practice is already used by:
  - LArSoft-based experiments:
    µBooNE, DUNE, LArIAT and ArgoNeuT
  - NOvA
  - MINERvA
  - GENIE
  - GlideinWMS (under dev)
- The CI Project can help to have healthy code at all times.

## The CI Project provides

- Jenkins project associated to the CI build
- repository with general scripts to handle CI builds
- repository for the experiment CI configuration files
- CI web application to monitor code status
- DB to collect statistics (build time, memory usage, ...), logs, plots, ...

# CI build schema



- Developers commit new code implementing bug fix, new feature, …
  - CI build job is triggered.

**CI workflow**

  - Pull the code from the repository.
  - Build the code.
  - Run unit tests.
  - Install the code.
  - Run CI tests.

  (depending on the experiment code these steps can be different)
  - Report the status of the CI build.
  - Notify developers in case of failure in the CI build caused by last commits.

## CI requirements from CI users

- A set of instructions to set up the **CI workflow**:

CI phases

  - setup the build environment
  - checkout the code
  - build the code
  - run unit tests
  - install the code
  - run integration tests

  (depending on the experiment code these steps can be different)
- Recommended storage:
  - all (most of ) package dependencies should live on CVMFS (it is used to run the code on OSG sites)
  - all data files required by the CI build job should live in dCache (reference files, input files, … )

# CI system configuration

- The CI system it is vastly configurable
- The CI workflow configuration allows to define quite arbitrary CI phases (see https://cdcvs.fnal.gov/redmine/projects/ci/wiki/Workflowcfg)
- CI tests configuration allows to run tests using the experiment executable with required options/args or using a script that helps to set up the experiment executable call (see https://cdcvs.fnal.gov/redmine/projects/ci/wiki/Ci_testscfg)
- CI validation configuration allow to set up grid jobs to process an experiment workflow defining details for each stage (see https://cdcvs.fnal.gov/redmine/projects/ci/wiki/CI_validation_test_using_the_grid)
- For more details there is the “Talk to expert: CI support” session on tomorrow

## CI test categories

- Regression test:
  - runs existing tests against modified code;
  - checks whether code changes break anything that worked prior to the change.
- Reproducibility test:
  - make sure that running the code using the same input, will “always” generate the same output.
- Back-compatibility test:
  - make sure that new code is able to access data files produced with a previous code release.
- Validation test:
  - make sure that new code produces meaningful results.
- …

# CI validation and grid support

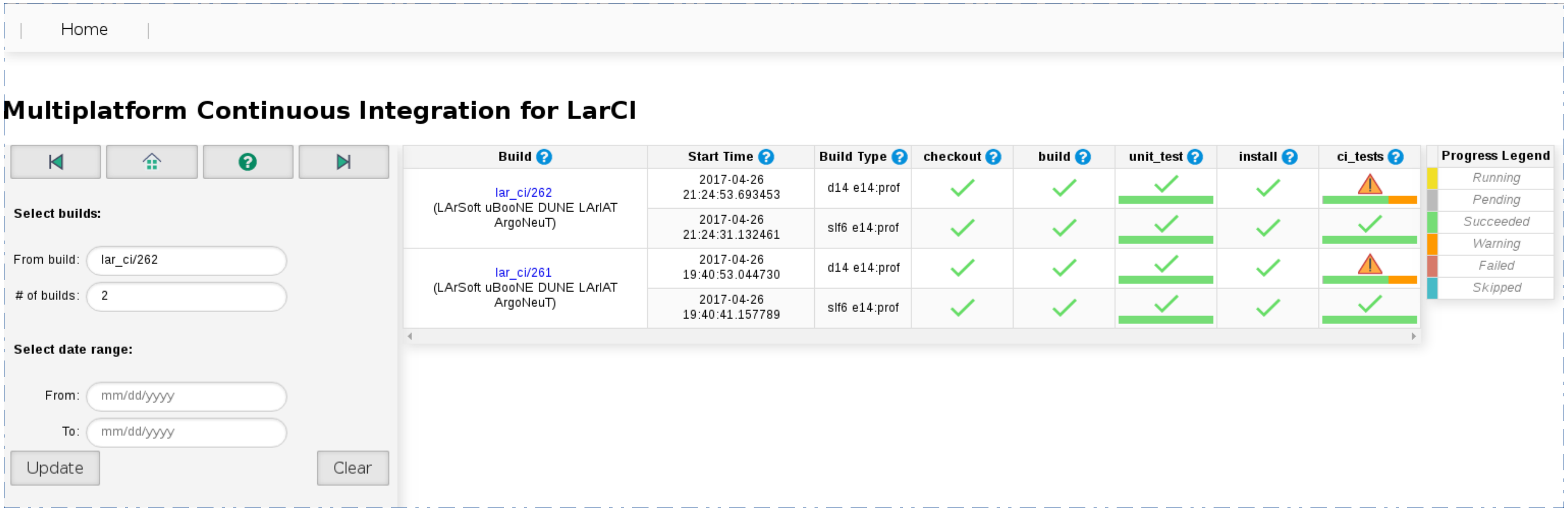
- Validation tests usually require thousands of events
  - for this purpose the **grid** can help to get the job done
- The CI allows to build a specific version of the code (tag, branch, ...) and uses it to run jobs on the grid
- Data produced by the CI validation are stored in a configurable dCache area for further analysis
  - also the code tarball and job logs are stored in dCache
- Provides stats about job usage resources
- Send an email report when the CI validation is complete and results are available
- Provide support to track jobs using POMS

# CI Web application for monitoring

## Useful to monitor past and current CI build status

[http://lar-ci-history.fnal.gov/LarCI/app]

- shows the status of each stage of the CI workflow;
- shows also the status for individual CI tests using a tool-tip;
- the status of each CI stage and CI test is identified by a color code;
- each bullet in the matrix provides a link to the logs;
- the Web pulls information from the LArCI DB.



Home

**Multiplatform Continuous Integration for LarCI**

**Select builds:**

| | |
|---|---|
| From build: | lar_ci/262 |
| # of builds: | 2 |

**Select date range:**

| | |
|---|---|
| From: | mm/dd/yyyy |
| To: | mm/dd/yyyy |

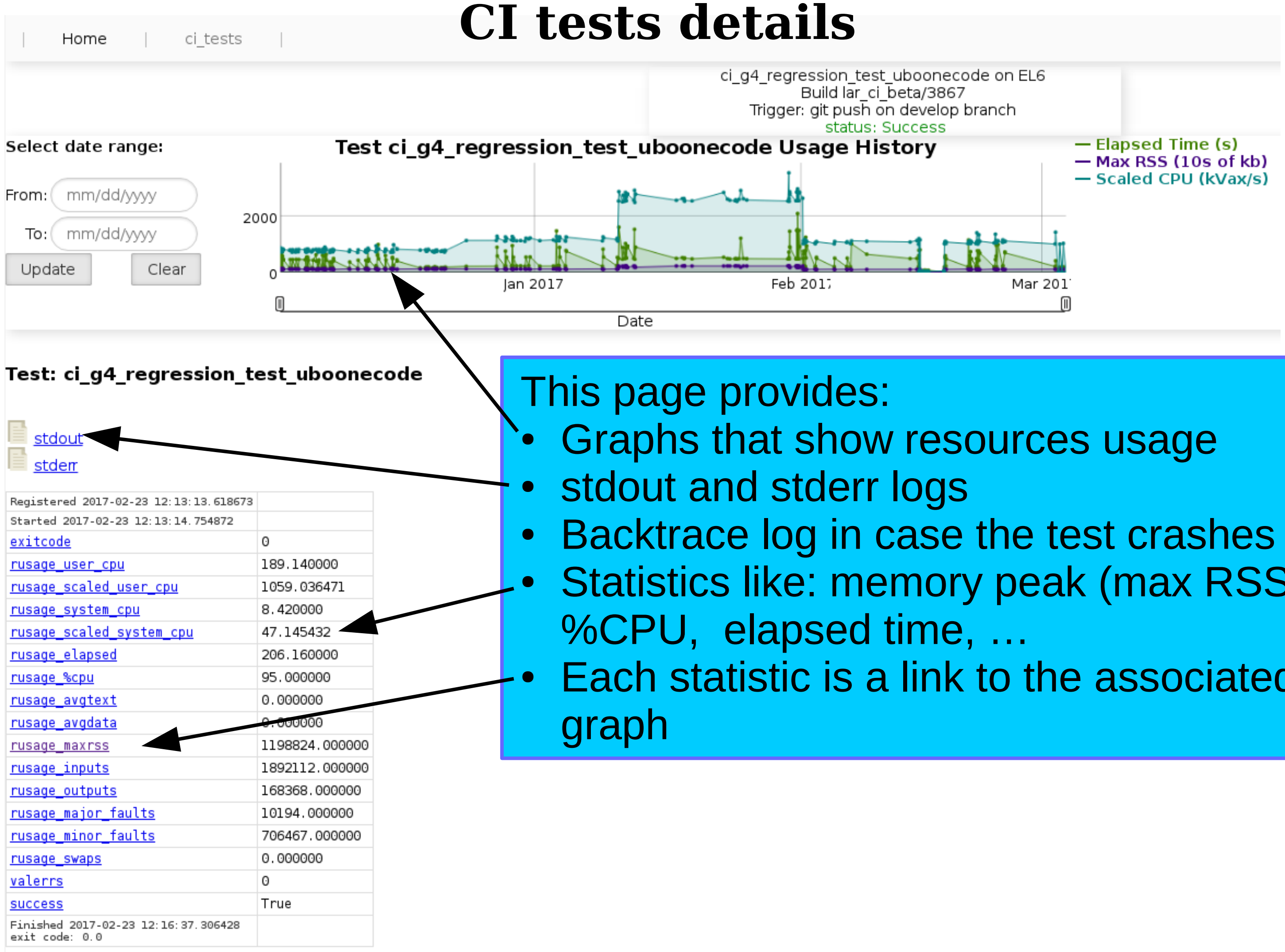
Update Clear

| Build | Start Time | Build Type | checkout | build | unit_test | install | ci_tests |
|---|---|---|---|---|---|---|---|
| lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 21:24:53.693453 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 21:24:31.132461 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |
| lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 19:40:53.044730 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 19:40:41.157789 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |

| Progress Legend | |
|---|---|
| 🟨 | Running |
| ⬜ | Pending |
| 🟩 | Succeeded |
| 🟧 | Warning |
| 🟥 | Failed |
| 🟦 | Skipped |

# CI Web application for monitoring

## CI tests details



ci_g4_regression_test_uboonecode on EL6
Build lar_ci_beta/3867
Trigger: git push on develop branch
status: Success

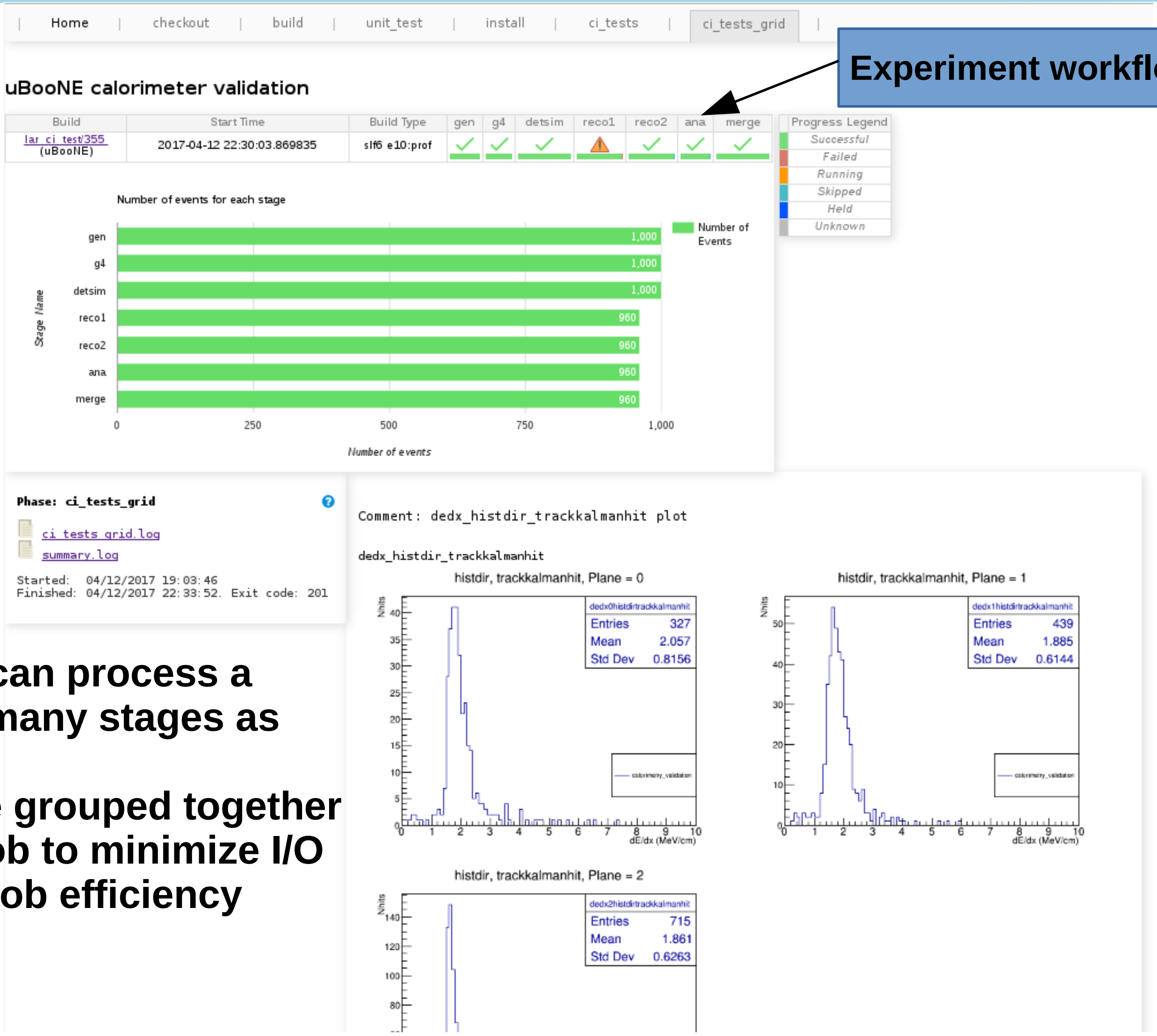
**Test: ci_g4_regression_test_uboonecode**

| | |
|---|---|
| Registered 2017-02-23 12:13:13.618673 | |
| Started 2017-02-23 12:13:14.754872 | |
| exitcode | 0 |
| rusage_user_cpu | 189.140000 |
| rusage_scaled_user_cpu | 1059.036471 |
| rusage_system_cpu | 8.420000 |
| rusage_scaled_system_cpu | 47.145432 |
| rusage_elapsed | 206.160000 |
| rusage_%cpu | 95.000000 |
| rusage_avgtext | 0.000000 |
| rusage_avgdata | 0.000000 |
| rusage_maxrss | 1198824.000000 |
| rusage_inputs | 1892112.000000 |
| rusage_outputs | 168368.000000 |
| rusage_major_faults | 10194.000000 |
| rusage_minor_faults | 706467.000000 |
| rusage_swaps | 0.000000 |
| valerrs | 0 |
| success | True |
| Finished 2017-02-23 12:16:37.306428 exit code: 0.0 | |

This page provides:
- Graphs that show resources usage
- stdout and stderr logs
- Backtrace log in case the test crashes
- Statistics like: memory peak (max RSS), %CPU, elapsed time, …
- Each statistic is a link to the associated graph

# CI Web application for monitoring

## CI validation



| Home | checkout | build | unit_test | install | ci_tests | ci_tests_grid |

uBooNE calorimeter validation

| Build | Start Time | Build Type | gen | g4 | detsim | reco1 | reco2 | ana | merge |
|---|---|---|---|---|---|---|---|---|---|
| lar_ci_test/355 (uBooNE) | 2017-04-12 22:30:03.869835 | slf6 e10:prof | ✓ | ✓ | ✓ | ⚠ | ✓ | ✓ | ✓ |

| Progress Legend |
|---|
| 🟩 Successful |
| 🟥 Failed |
| 🟧 Running |
| 🟦 Skipped |
| 🟦 Held |
| ⬜ Unknown |

```
Phase: ci_tests_grid

ci_tests_grid.log
summary.log

Started:  04/12/2017 19:03:46
Finished: 04/12/2017 22:33:52. Exit code: 201
```

```
Comment: dedx_histdir_trackkalmanhit plot

dedx_histdir_trackkalmanhit
```

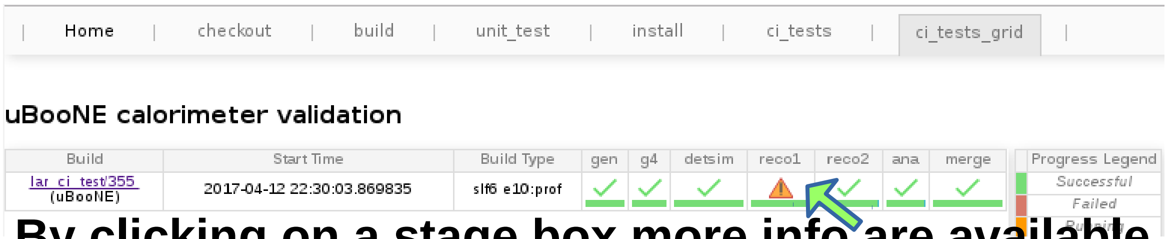
**Experiment workflow stages**

**Progress bars show the number of events available for each stage**

- **The CI validation can process a workflow with as many stages as needed**
- **The stages can be grouped together in the same grid job to minimize I/O and improve grid job efficiency**

# CI Web application for monitoring

## CI validation



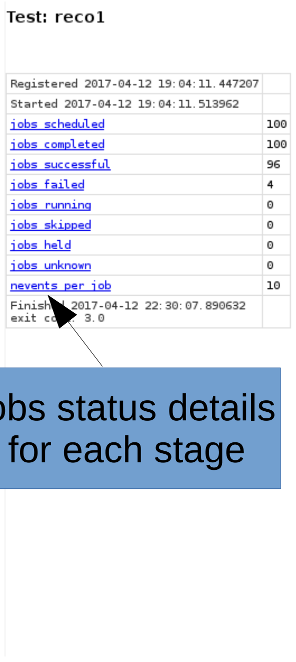
| Home | checkout | build | unit_test | install | ci_tests | ci_tests_grid |

uBooNE calorimeter validation

| Build | Start Time | Build Type | gen | g4 | detsim | reco1 | reco2 | ana | merge |
|---|---|---|---|---|---|---|---|---|---|
| lar_ci_test/355 (uBooNE) | 2017-04-12 22:30:03.869835 | slf6 e10:prof | ✓ | ✓ | ✓ | ⚠ | ✓ | ✓ | ✓ |

| Progress Legend |
|---|
| 🟩 Successful |
| 🟥 Failed |
| 🟧 Running |

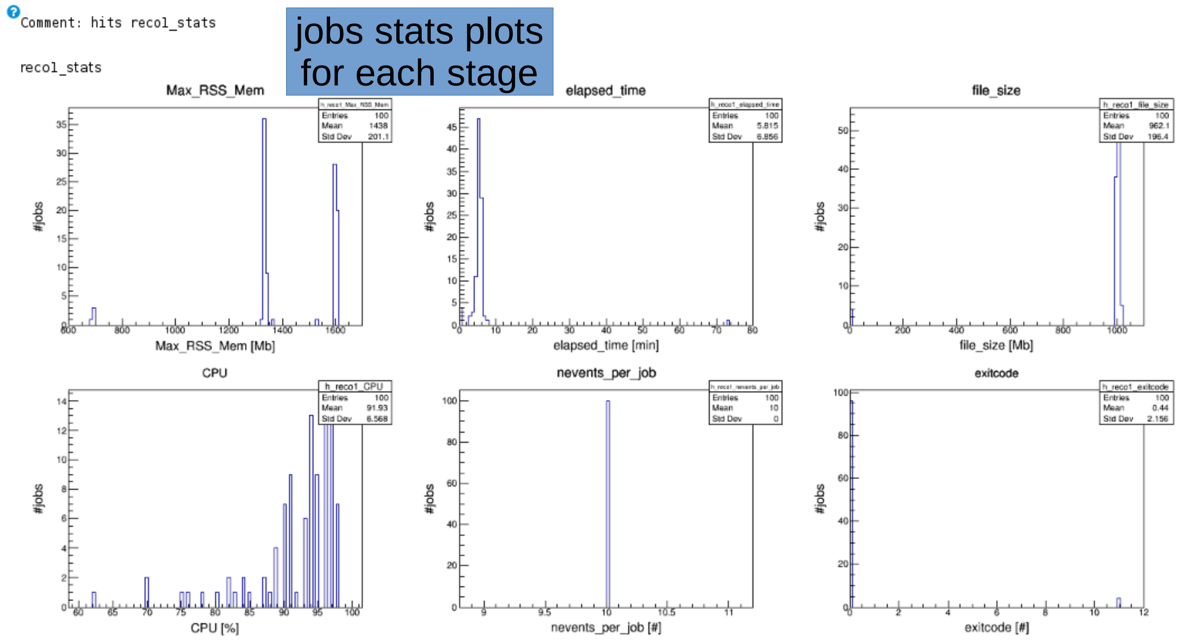
- **By clicking on a stage box more info are available**
  - jobs stats which include: resident memory peak, elapsed time, file size
  - job status details



Test: reco1

| | |
|---|---|
| Registered 2017-04-12 19:04:11.447207 | |
| Started 2017-04-12 19:04:11.513962 | |
| jobs scheduled | 100 |
| jobs completed | 100 |
| jobs successful | 96 |
| jobs failed | 4 |
| jobs running | 0 |
| jobs skipped | 0 |
| jobs held | 0 |
| jobs unknown | 0 |
| nevents per job | 10 |
| Finish 2017-04-12 22:30:07.890632 exit co 3.0 | |

jobs status details for each stage



jobs stats plots for each stage

# Continuous Integration highlights

- CI will help you to have a healthy code at all times
- CI workflow can handle code in ***git***, ***svn*** and ***cvs*** repositories
- CI workflow can build and test a list of mutually dependent modules together
- user can test any desired branch/tag of the code
- user can run CI tests locally using her/his own just built code
- users can add/implement their own CI tests.
- Experiments will be the stakeholder
- References:
  - the CI Project wiki

# Stats from CI user builds

| User | OS | #weeks | #builds | #builds/week | #warning | #failures |
|---|---|---|---|---|---|---|
| **LArSoft** | SLF6/MacOS | 14 | 744 | 54 | 49 | 36 |
| **NOvA** | SLF6 | 24 | 1035 | 43 | 48 | 59 |
| **GENIE** | SLF6/SLF7 | 11 | 330 | 30 | 0 | 0 |
| **MINERvA** | SLF6 | 7 | 63 | 7 | 0 | 4 |

- LArSoft and NOvA CI builds are triggered by commits
- GENIE and MINERvA CI builds are triggered nightly by a crontab
- Disclaimer
  - **“warning”** means that experiment code run fine, but some test on the output against a reference output is not successful
  - **“#failures”** includes also failures due to infrastructure issues (dCache unavailable, …)
  - In the case of LArSoft there are CI builds known to fail, experiment release managers need some time to update LArSoft version dependencies when a new LArSoft weekly tag is released

## Are you interested in the CI service?

Experiments can require the CI service through SNOW:
Scientific Computing Services / Scientific Production Processing / Continuous Integration Service

- Tomorrow there is the “Talk to expert: CI Support” session
- Basic requirements for the experiment code:
  - have a well defined and documented build chain;
  - have all software dependencies available on CVMFS;
  - have all needed accessory files (flux files, …) on dCache.

## Summary and Future plans

- The CI Project Team is glad to provide the CI service to IF experiments
- The CI practice has already been successfully adopted by LArSoft-based experiments and NOvA
- GENIE and  MINNERvA have been on-boarded since few months
- the plan is to on-board all IF experiments
  - CI service will provide a software facility to constantly monitor the status of the experiment code
    - will help to maintain a healthy code
    - will help to monitor resource usage
    - will help to monitor code performances
- New features are coming: memory profiling and more
- Feature requests from CI users are welcome!

Thank you!

## Back up slides

# Work flow configuration example

FABRIC FOR FRONTIER EXPERIMENTS

More details at https://cdcvs.fnal.gov/redmine/projects/ci/wiki/Workflowcfg

**cfg/workflow.cfg excerpt**

```
[default]
workflow       = ${GENIE_WORKFLOW:-GENIE_ROOT6}
notify_email_to   = vito@fnal.gov,perdue@fnal.gov,yarba_j@fnal.gov
notify_succeeded_email_to =
notify_success = true
notify_warning_email_to =
notify_failed_email_to =
notify_blame   = false
proxy_vo       = /fermilab/genie
build_db_uri=http://dbweb6.fnal.gov:8080/GenieCI/app
```

default configuration

```
[GENIE_ROOT6]
experiment    = GENIE
qualifier    = "ROOT6+e10:${BUILDTYPE}"
personality   = make
ci_test_lists = quick_test_genie
revision      = ${GENIE_REVISION:-trunk}
proxy_flag    = false
skip_phases   = *@slf7
phases        = _evalROOT6_n checkout build unit_test ci_tests
```

workflow configuration

```
[make]
# define what the stages do:
# _evalROOT6_n: setup the code environment
...
#checkout: instruction to checkout the code
...
#build: instruction to build the code
...
# unit_test:instruction to run unit tests
...
#ci_tests: instruction to run the CI tests
...
```

"personality" configuration using *make* as build tool

- The "***default*** configuration" selects the ***workflow*** to use
- The "***workflow*** configuration" selects the CI phases to run in the CI build, the ***personality*** and the list of code modules (repositories) to process
- The "***personality*** configuration" defines the CI phases using a particular build tool
- In the current implementation the GENIE CI workflow runs 5 CI phases: **_eval_n, checkout, build, unit_test, ci_tests.** The list of CI phases and their definition are arbitrary
- The CI phase is highly configurable, it can run an arbitrary sequence of commands

# CI test configuration example

More details at https://cdcvs.fnal.gov/redmine/projects/ci/wiki/Ci_testscfg

**test/ci_tests.cfg excerpt**

```
[DEFAULT]
EXPSCRIPT_NOVASOFT=ci_regression_test_novasoft.sh
INPUTFILEDIR_NOVASOFT=/pnfs/nova/persistent/users/novapro/ci_tests_inputfiles
INPUTFILEDIR_XROOT_NOVASOFT=xroot://fndca1.fnal.gov:1094//pnfs//fnal.gov/usr/nova/persistent/users/novapro/ci_tests_inputfiles
CI_EXP_CODE=NOVASOFT
IDENTIFIER_NOVASOFT=${build_identifier}
PLATFORM_NOVASOFT=${build_platform}
TESTMASK_NOVASOFT=%(RUN_TEST_NOVASOFT)s%(CHECK_PRODUCTS_NOVASOFT)s%(CHECK_PRODUCT_SIZE_NOVASOFT)s
stdargs=%(mcargs)s  --input-file %(INPUT_FILE)s --reference-files %(REFERENCE_FILE)s
```

Define global variables

```
[test ci_raw2root_nd_t00_regression_test_novasoft]
script=%(EXPSCRIPT_NOVASOFT)s
STAGE_NAME=raw2root_nd_t00
NEVENTS=1
FHiCL_FILE=daq2rawdigitjob.fcl
BASE=neardet_r00011552_s00_t00
INPUT_FILE=%(BASE)s.raw
FETCH_INPUT=%(INPUTFILEDIR_LOCAL_NOVASOFT)s/%(STAGE_NAME)s/%(BASE)s.raw
REFERENCE_FILE=%(INPUTFILEDIR_XROOT_NOVASOFT)s/%(STAGE_NAME)s/%(BASE)s_%(ref)s.artdaq.root
OUTPUT_STREAM=out1:%(BASE)s_%(cur)s.artdaq.root
args=%(stdargs)s --input-files-to-fetch %(FETCH_INPUT)s
```

CI test section

CI test suite section

```
[suite default]
testlist=ci_raw2root_nd_t00_regression_test_novasoft ci_raw2root_nd_t02_regression_test_novasoft ci_raw2root_fd_t00_regression_test_novasoft
ci_raw2root_fd_t02_regression_test_novasoft ci_fullchain_nd_data_regression_test_novasoft ci_fullchain_fd_data_regression_test_novasoft
ci_calib_nd_regression_test_novasoft ci_calib_fd_regression_test_novasoft ci_mcgen_nd_regression_test_novasoft ci_mcgen_fdoverlay_regression_test_novasoft
ci_mcgen_rock_regression_test_novasoft ci_mcgen_cry_regression_test_novasoft
```

- The “default section” initializes a set of global variables required to initialize the script that runs the CI tests.
- The “CI test section” sets specific configuration to run the CI test.
- The “CI test suite section” collects a list of tests to run all together.

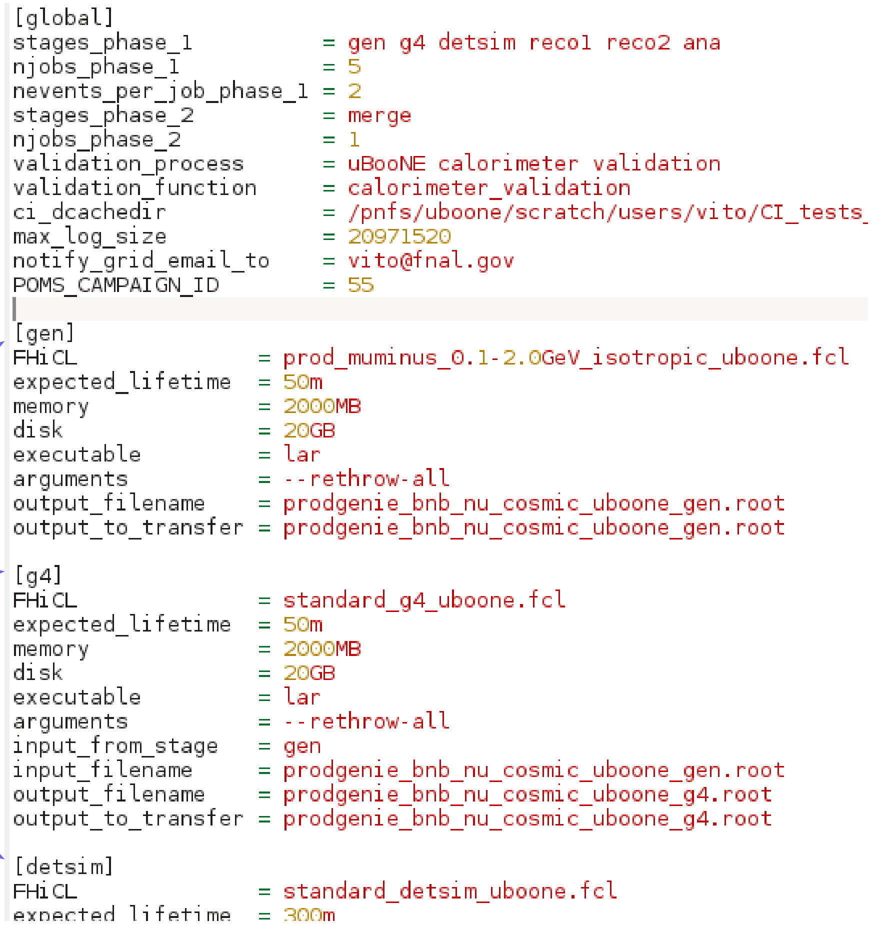
# CI validation configuration example

More details at https://cdcvs.fnal.gov/redmine/projects/ci/wiki/CI_validation_test_using_the_grid

- The CI validation phase has its own configuration file
- It consists of two types of sections:

**[global]** section that defines the experiment workflow

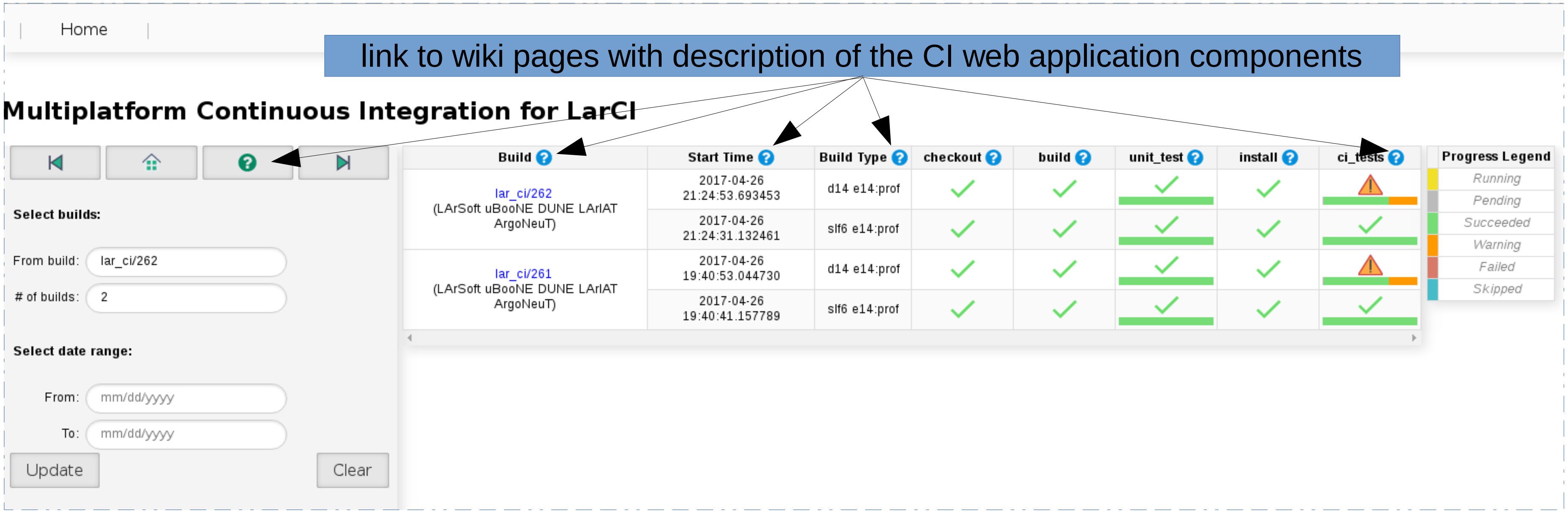
**[<stage>]** section that specifies stage properties



```
[global]
stages_phase_1           = gen g4 detsim reco1 reco2 ana
njobs_phase_1            = 5
nevents_per_job_phase_1  = 2
stages_phase_2           = merge
njobs_phase_2            = 1
validation_process       = uBooNE calorimeter validation
validation_function      = calorimeter_validation
ci_dcachedir             = /pnfs/uboone/scratch/users/vito/CI_tests_
max_log_size             = 20971520
notify_grid_email_to     = vito@fnal.gov
POMS_CAMPAIGN_ID         = 55

[gen]
FHiCL                 = prod_muminus_0.1-2.0GeV_isotropic_uboone.fcl
expected_lifetime     = 50m
memory                = 2000MB
disk                  = 20GB
executable            = lar
arguments             = --rethrow-all
output_filename       = prodgenie_bnb_nu_cosmic_uboone_gen.root
output_to_transfer    = prodgenie_bnb_nu_cosmic_uboone_gen.root

[g4]
FHiCL                 = standard_g4_uboone.fcl
expected_lifetime     = 50m
memory                = 2000MB
disk                  = 20GB
executable            = lar
arguments             = --rethrow-all
input_from_stage      = gen
input_filename        = prodgenie_bnb_nu_cosmic_uboone_gen.root
output_filename       = prodgenie_bnb_nu_cosmic_uboone_g4.root
output_to_transfer    = prodgenie_bnb_nu_cosmic_uboone_g4.root

[detsim]
FHiCL                 = standard_detsim_uboone.fcl
expected_lifetime     = 300m
```

# CI Web application for monitoring

- **In-line documentation:**



link to wiki pages with description of the CI web application components

Home

**Multiplatform Continuous Integration for LarCI**

**Select builds:**

From build: lar_ci/262

# of builds: 2

**Select date range:**

From: mm/dd/yyyy

To: mm/dd/yyyy

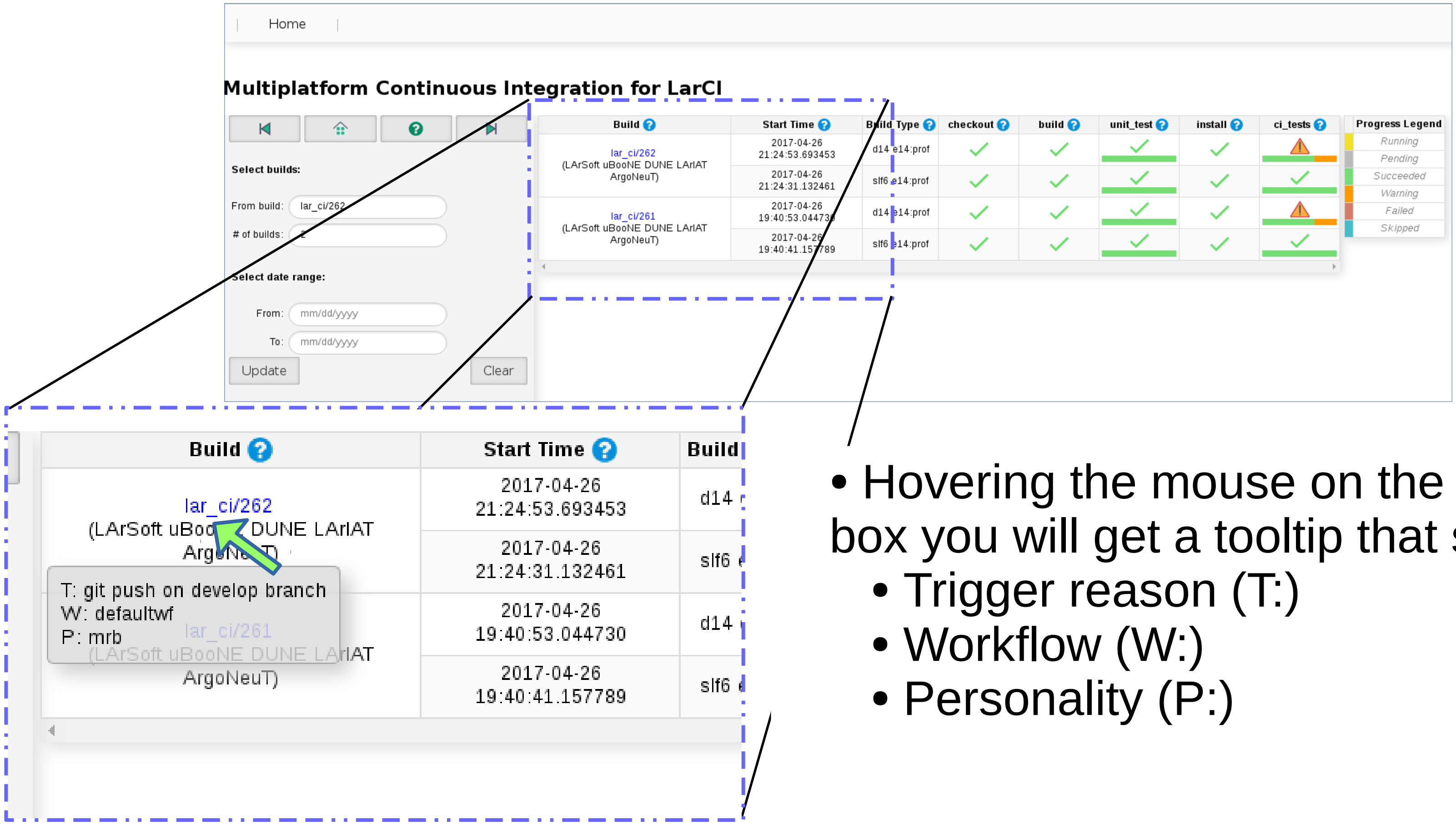
Update | Clear

| Build | Start Time | Build Type | checkout | build | unit_test | install | ci_tests |
|---|---|---|---|---|---|---|---|
| lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 21:24:53.693453 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 21:24:31.132461 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |
| lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 19:40:53.044730 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 19:40:41.157789 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |

| Progress Legend | |
|---|---|
| 🟨 | Running |
| ⬜ | Pending |
| 🟩 | Succeeded |
| 🟧 | Warning |
| 🟥 | Failed |
| 🟦 | Skipped |

# CI Web application for monitoring

- **CI Build details**



Home

Multiplatform Continuous Integration for LarCI

Select builds:

From build: lar_ci/262

# of builds: 2

Select date range:

From: mm/dd/yyyy

To: mm/dd/yyyy

Update Clear

| Build | Start Time | Build Type | checkout | build | unit_test | install | ci_tests |
|---|---|---|---|---|---|---|---|
| lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 21:24:53.693453 | d14:e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 21:24:31.132461 | slf6:e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |
| lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 19:40:53.044730 | d14:e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 19:40:41.157789 | slf6:e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |

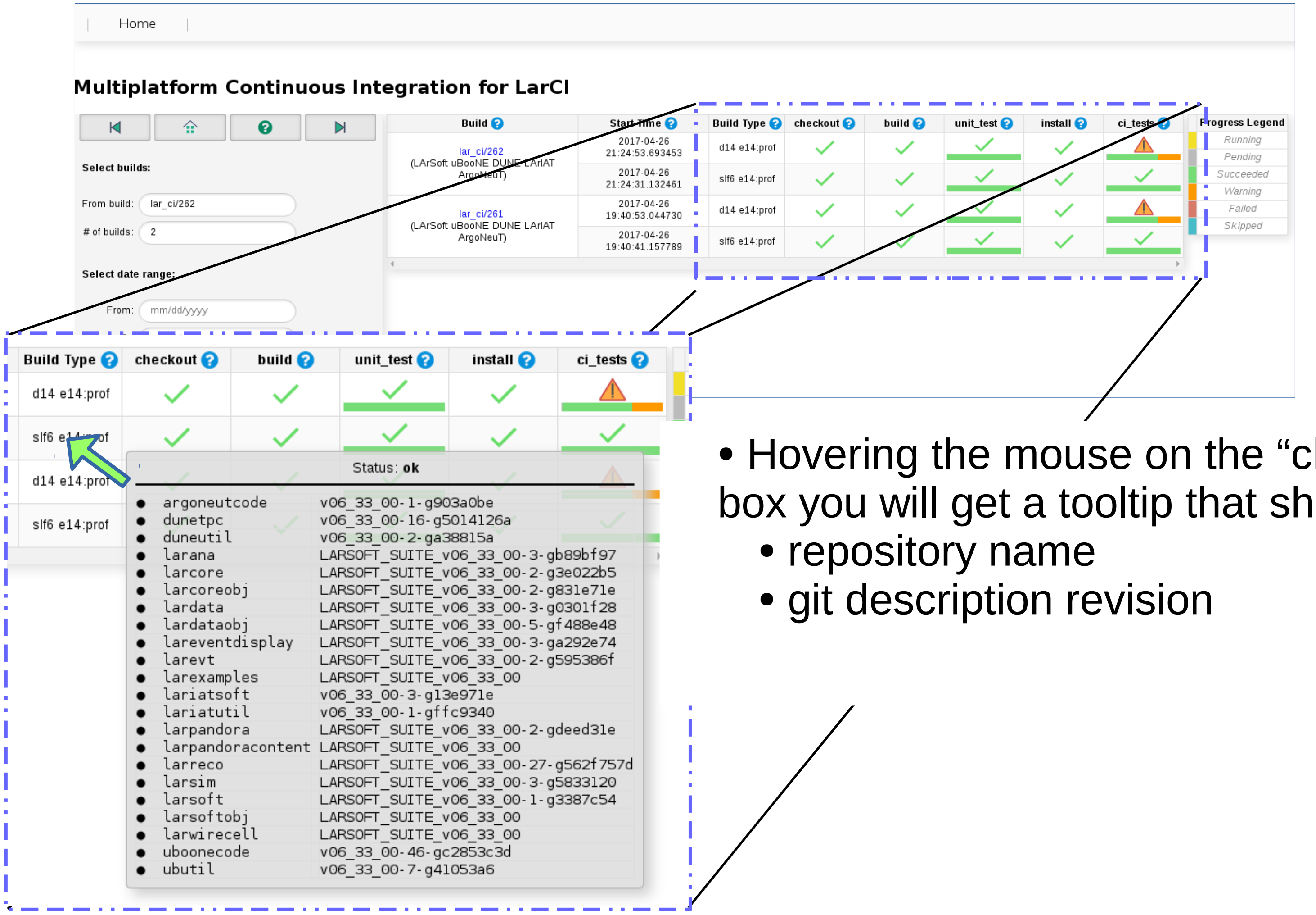
| Progress Legend |
|---|
| 🟨 Running |
| ⬜ Pending |
| 🟩 Succeeded |
| 🟧 Warning |
| 🟥 Failed |
| 🟦 Skipped |

T: git push on develop branch
W: defaulttwf
P: mrb

- Hovering the mouse on the "Build" box you will get a tooltip that shows:
  - Trigger reason (T:)
  - Workflow (W:)
  - Personality (P:)

# CI Web application for monitoring

- **Checkout details**



Multiplatform Continuous Integration for LarCI

| Build | Start Time | Build Type | checkout | build | unit_test | install | ci_tests |
|---|---|---|---|---|---|---|---|
| lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 21:24:53.693453 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 21:24:31.132461 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |
| lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 19:40:53.044730 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 19:40:41.157789 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |

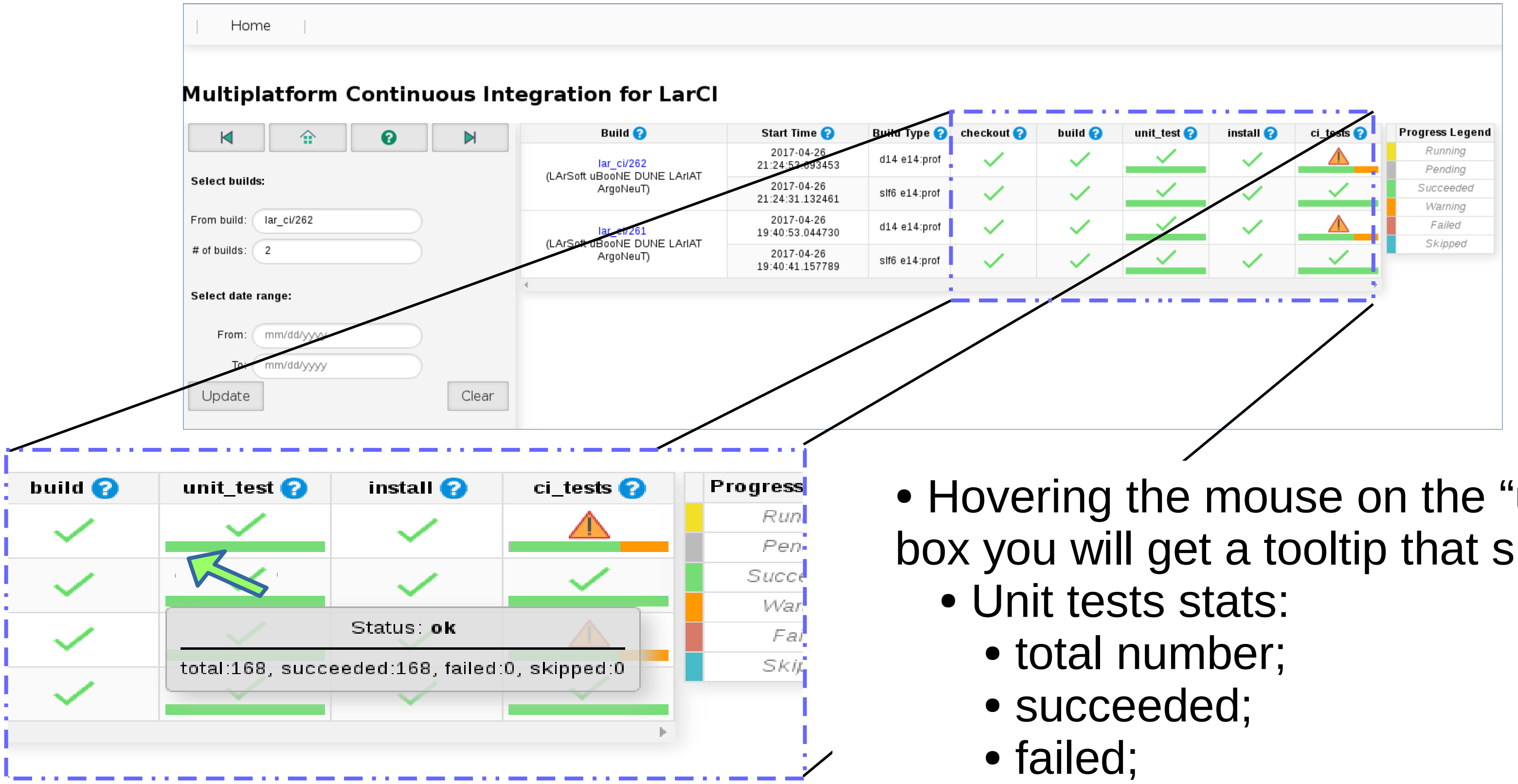
Status: **ok**

```
● argoneutcode        v06_33_00-1-g903a0be
● dunetpc             v06_33_00-16-g5014126a
● duneutil            v06_33_00-2-ga38815a
● larana              LARSOFT_SUITE_v06_33_00-3-gb89bf97
● larcore             LARSOFT_SUITE_v06_33_00-2-g3e022b5
● larcoreobj          LARSOFT_SUITE_v06_33_00-2-g831e71e
● lardata             LARSOFT_SUITE_v06_33_00-3-g0301f28
● lardataobj          LARSOFT_SUITE_v06_33_00-5-gf488e48
● lareventdisplay     LARSOFT_SUITE_v06_33_00-3-ga292e74
● larevt              LARSOFT_SUITE_v06_33_00-2-g595386f
● larexamples         LARSOFT_SUITE_v06_33_00
● lariatsoft          v06_33_00-3-g13e971e
● lariatutil          v06_33_00-1-gffc9340
● larpandora          LARSOFT_SUITE_v06_33_00-2-gdeed31e
● larpandoracontent   LARSOFT_SUITE_v06_33_00
● larreco             LARSOFT_SUITE_v06_33_00-27-g562f757d
● larsim              LARSOFT_SUITE_v06_33_00-3-g5833120
● larsoft             LARSOFT_SUITE_v06_33_00-1-g3387c54
● larsoftobj          LARSOFT_SUITE_v06_33_00
● larwirecell         LARSOFT_SUITE_v06_33_00
● uboonecode          v06_33_00-46-gc2853c3d
● ubutil              v06_33_00-7-g41053a6
```

- Hovering the mouse on the “checkout” box you will get a tooltip that shows:
  - repository name
  - git description revision

# CI Web application for monitoring

- **Unit test details**



| Home |

**Multiplatform Continuous Integration for LarCI**

Select builds:

From build: lar_ci/262

# of builds: 2

Select date range:

From: mm/dd/yyyy

To: mm/dd/yyyy

Update | Clear

| Build | Start Time | Build Type | checkout | build | unit_test | install | ci_tests |
|---|---|---|---|---|---|---|---|
| lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 21:24:52.693453 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 21:24:31.132461 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |
| lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 19:40:53.044730 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 19:40:41.157789 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |

| Progress Legend | |
|---|---|
| 🟨 | Running |
| ⬜ | Pending |
| 🟩 | Succeeded |
| 🟧 | Warning |
| 🟥 | Failed |
| 🟦 | Skipped |

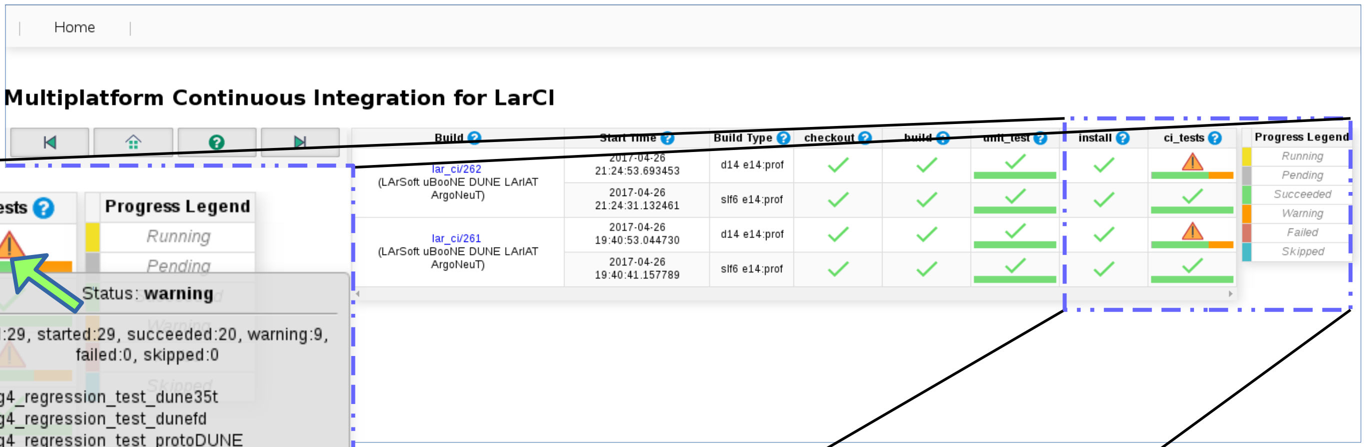
Status: **ok**

total:168, succeeded:168, failed:0, skipped:0

- Hovering the mouse on the “unit_test” box you will get a tooltip that shows:
  - Unit tests stats:
    - total number;
    - succeeded;
    - failed;
    - skipped.

# CI Web application for monitoring

- **CI tests details**

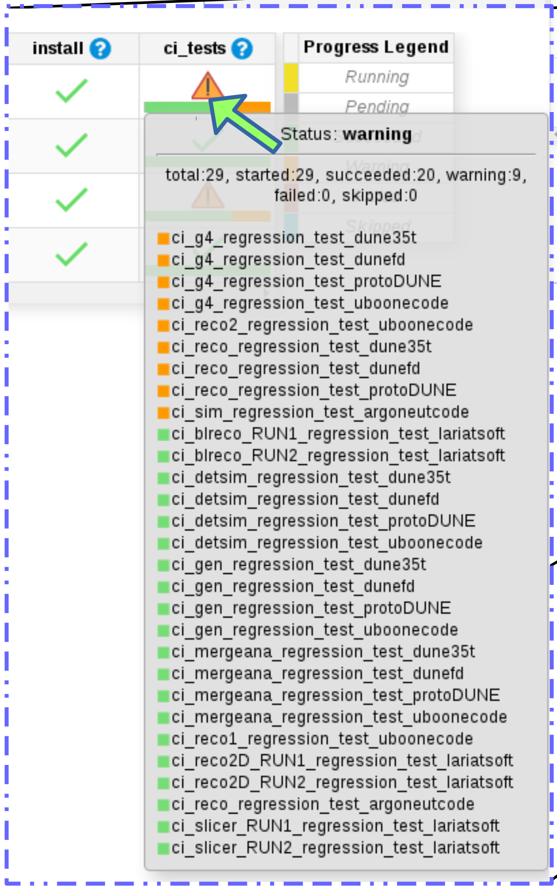


Home

**Multiplatform Continuous Integration for LarCI**

| Build | Start Time | Build Type | checkout | build | unit_test | install | ci_tests |
|---|---|---|---|---|---|---|---|
| lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 21:24:53.693453 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 21:24:31.132461 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |
| lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT) | 2017-04-26 19:40:53.044730 | d14 e14:prof | ✓ | ✓ | ✓ | ✓ | ⚠ |
| | 2017-04-26 19:40:41.157789 | slf6 e14:prof | ✓ | ✓ | ✓ | ✓ | ✓ |

| Progress Legend |
|---|
| Running |
| Pending |
| Succeeded |
| Warning |
| Failed |
| Skipped |



Status: **warning**

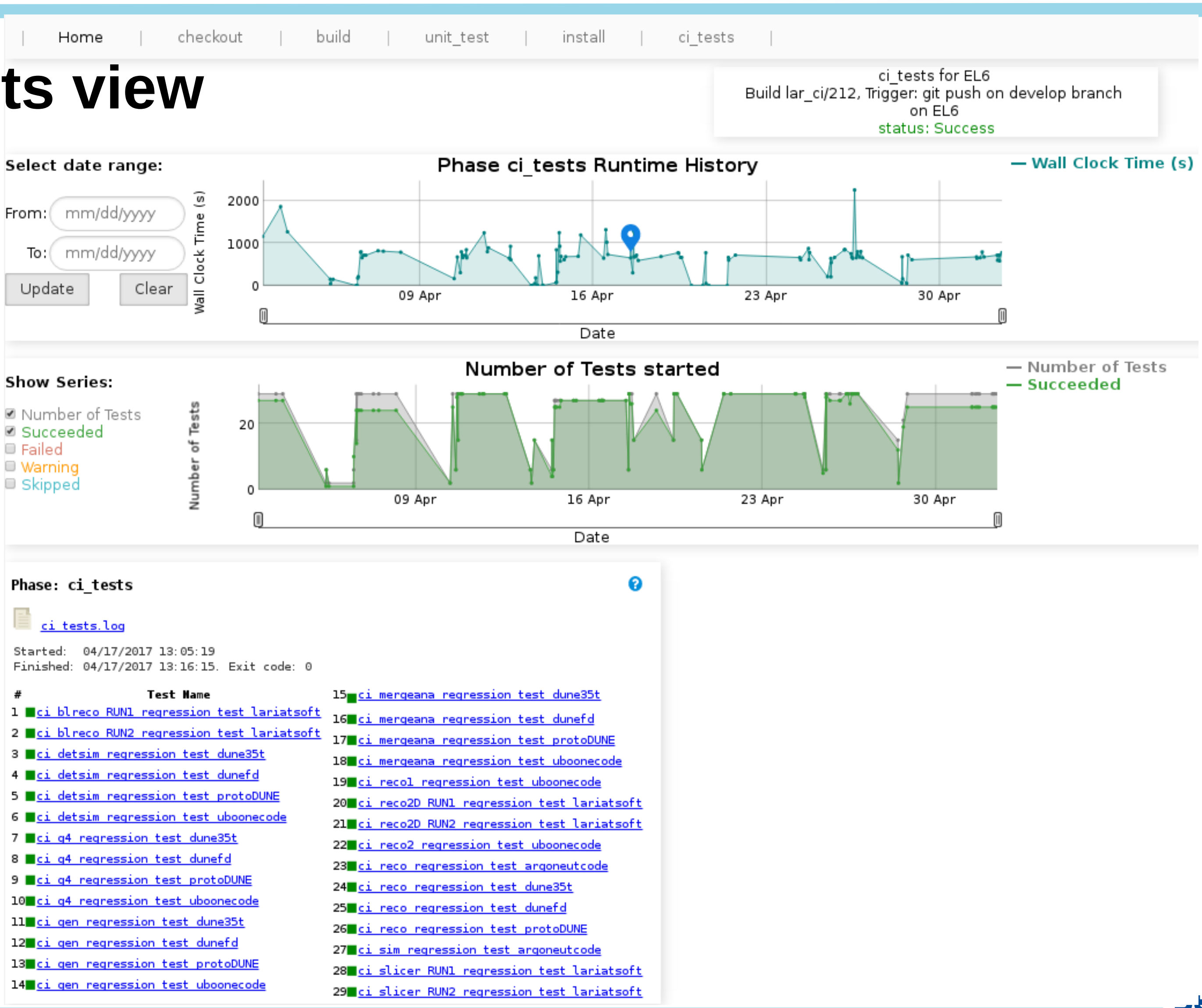
total:29, started:29, succeeded:20, warning:9, failed:0, skipped:0

🟧ci_g4_regression_test_dune35t
🟧ci_g4_regression_test_dunefd
🟧ci_g4_regression_test_protoDUNE
🟧ci_g4_regression_test_uboonecode
🟧ci_reco2_regression_test_uboonecode
🟧ci_reco_regression_test_dune35t
🟧ci_reco_regression_test_dunefd
🟧ci_reco_regression_test_protoDUNE
🟧ci_sim_regression_test_argoneutcode
🟩ci_blreco_RUN1_regression_test_lariatsoft
🟩ci_blreco_RUN2_regression_test_lariatsoft
🟩ci_detsim_regression_test_dune35t
🟩ci_detsim_regression_test_dunefd
🟩ci_detsim_regression_test_protoDUNE
🟩ci_detsim_regression_test_uboonecode
🟩ci_gen_regression_test_dune35t
🟩ci_gen_regression_test_dunefd
🟩ci_gen_regression_test_protoDUNE
🟩ci_gen_regression_test_uboonecode
🟩ci_mergeana_regression_test_dune35t
🟩ci_mergeana_regression_test_dunefd
🟩ci_mergeana_regression_test_protoDUNE
🟩ci_mergeana_regression_test_uboonecode
🟩ci_reco1_regression_test_uboonecode
🟩ci_reco2D_RUN1_regression_test_lariatsoft
🟩ci_reco2D_RUN2_regression_test_lariatsoft
🟩ci_reco_regression_test_argoneutcode
🟩ci_slicer_RUN1_regression_test_lariatsoft
🟩ci_slicer_RUN2_regression_test_lariatsoft

- Hovering the mouse on the “ci_test” box you will get a tooltip that shows:
  - CI tests stats:
    - total number;
    - succeeded;
    - warning;
    - failed;
    - Skipped.
  - Summary of CI tests status.

# CI Web application for monitoring

- **CI tests view**



ci_tests for EL6
Build lar_ci/212, Trigger: git push on develop branch on EL6
status: Success

**Phase: ci_tests**

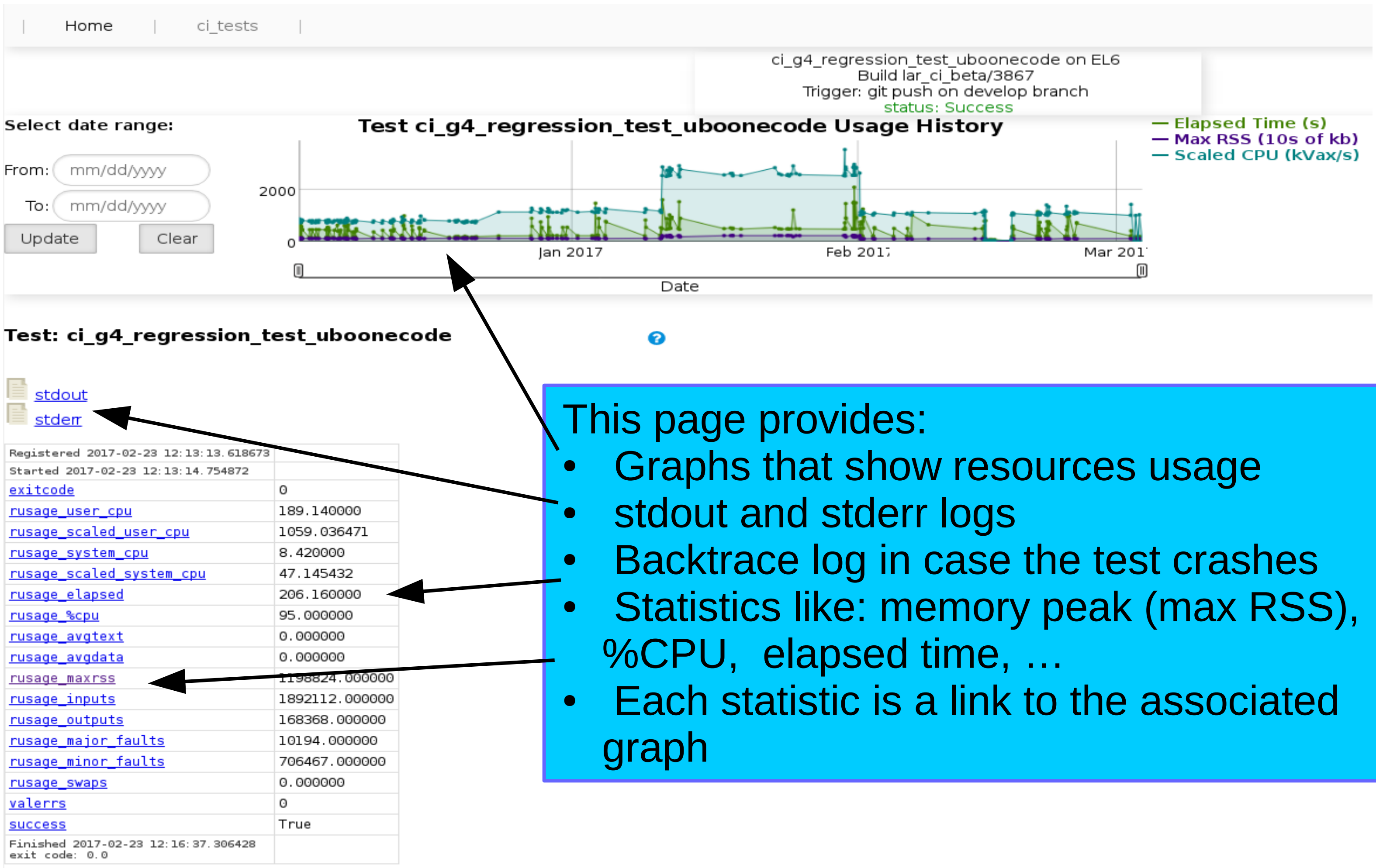
ci_tests.log

```
Started:  04/17/2017 13:05:19
Finished: 04/17/2017 13:16:15. Exit code: 0
```

| # | Test Name |
|---|---|
| 1 | 🟩 ci_blreco_RUN1_regression_test_lariatsoft |
| 2 | 🟩 ci_blreco_RUN2_regression_test_lariatsoft |
| 3 | 🟩 ci_detsim_regression_test_dune35t |
| 4 | 🟩 ci_detsim_regression_test_dunefd |
| 5 | 🟩 ci_detsim_regression_test_protoDUNE |
| 6 | 🟩 ci_detsim_regression_test_uboonecode |
| 7 | 🟩 ci_g4_regression_test_dune35t |
| 8 | 🟩 ci_g4_regression_test_dunefd |
| 9 | 🟩 ci_g4_regression_test_protoDUNE |
| 10 | 🟩 ci_g4_regression_test_uboonecode |
| 11 | 🟩 ci_gen_regression_test_dune35t |
| 12 | 🟩 ci_gen_regression_test_dunefd |
| 13 | 🟩 ci_gen_regression_test_protoDUNE |
| 14 | 🟩 ci_gen_regression_test_uboonecode |
| 15 | 🟩 ci_mergeana_regression_test_dune35t |
| 16 | 🟩 ci_mergeana_regression_test_dunefd |
| 17 | 🟩 ci_mergeana_regression_test_protoDUNE |
| 18 | 🟩 ci_mergeana_regression_test_uboonecode |
| 19 | 🟩 ci_reco1_regression_test_uboonecode |
| 20 | 🟩 ci_reco2D_RUN1_regression_test_lariatsoft |
| 21 | 🟩 ci_reco2D_RUN2_regression_test_lariatsoft |
| 22 | 🟩 ci_reco2_regression_test_uboonecode |
| 23 | 🟩 ci_reco_regression_test_argoneutcode |
| 24 | 🟩 ci_reco_regression_test_dune35t |
| 25 | 🟩 ci_reco_regression_test_dunefd |
| 26 | 🟩 ci_reco_regression_test_protoDUNE |
| 27 | 🟩 ci_sim_regression_test_argoneutcode |
| 28 | 🟩 ci_slicer_RUN1_regression_test_lariatsoft |
| 29 | 🟩 ci_slicer_RUN2_regression_test_lariatsoft |

# CI Web application for monitoring

- **CI tests details**



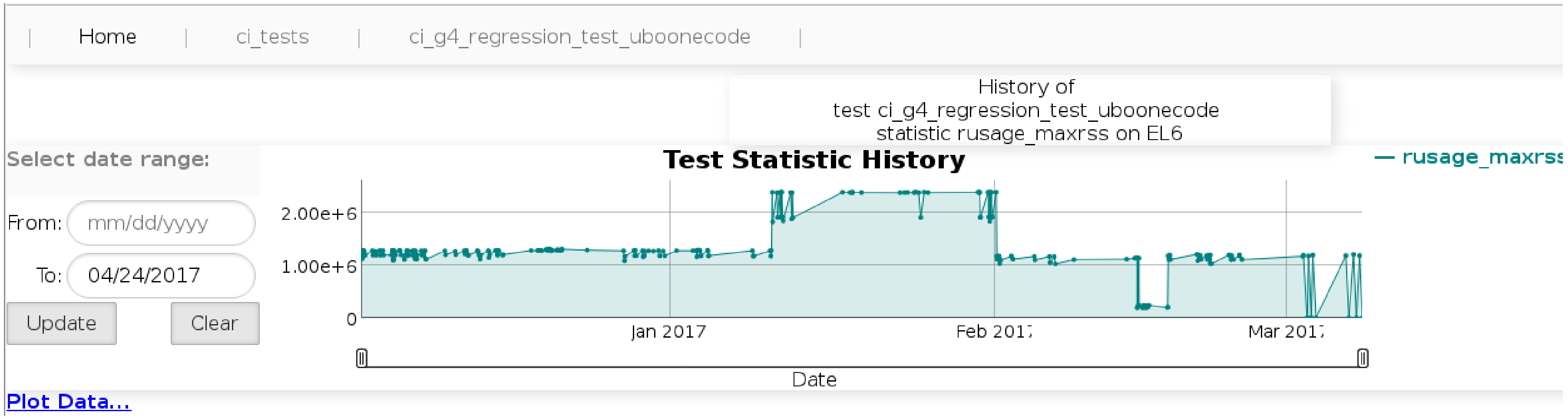
**Test: ci_g4_regression_test_uboonecode**

| | |
|---|---|
| Registered 2017-02-23 12:13:13.618673 | |
| Started 2017-02-23 12:13:14.754872 | |
| exitcode | 0 |
| rusage_user_cpu | 189.140000 |
| rusage_scaled_user_cpu | 1059.036471 |
| rusage_system_cpu | 8.420000 |
| rusage_scaled_system_cpu | 47.145432 |
| rusage_elapsed | 206.160000 |
| rusage_%cpu | 95.000000 |
| rusage_avgtext | 0.000000 |
| rusage_avgdata | 0.000000 |
| rusage_maxrss | 1198824.000000 |
| rusage_inputs | 1892112.000000 |
| rusage_outputs | 168368.000000 |
| rusage_major_faults | 10194.000000 |
| rusage_minor_faults | 706467.000000 |
| rusage_swaps | 0.000000 |
| valerrs | 0 |
| success | True |
| Finished 2017-02-23 12:16:37.306428<br>exit code: 0.0 | |

This page provides:
- Graphs that show resources usage
- stdout and stderr logs
- Backtrace log in case the test crashes
- Statistics like: memory peak (max RSS), %CPU, elapsed time, …
- Each statistic is a link to the associated graph

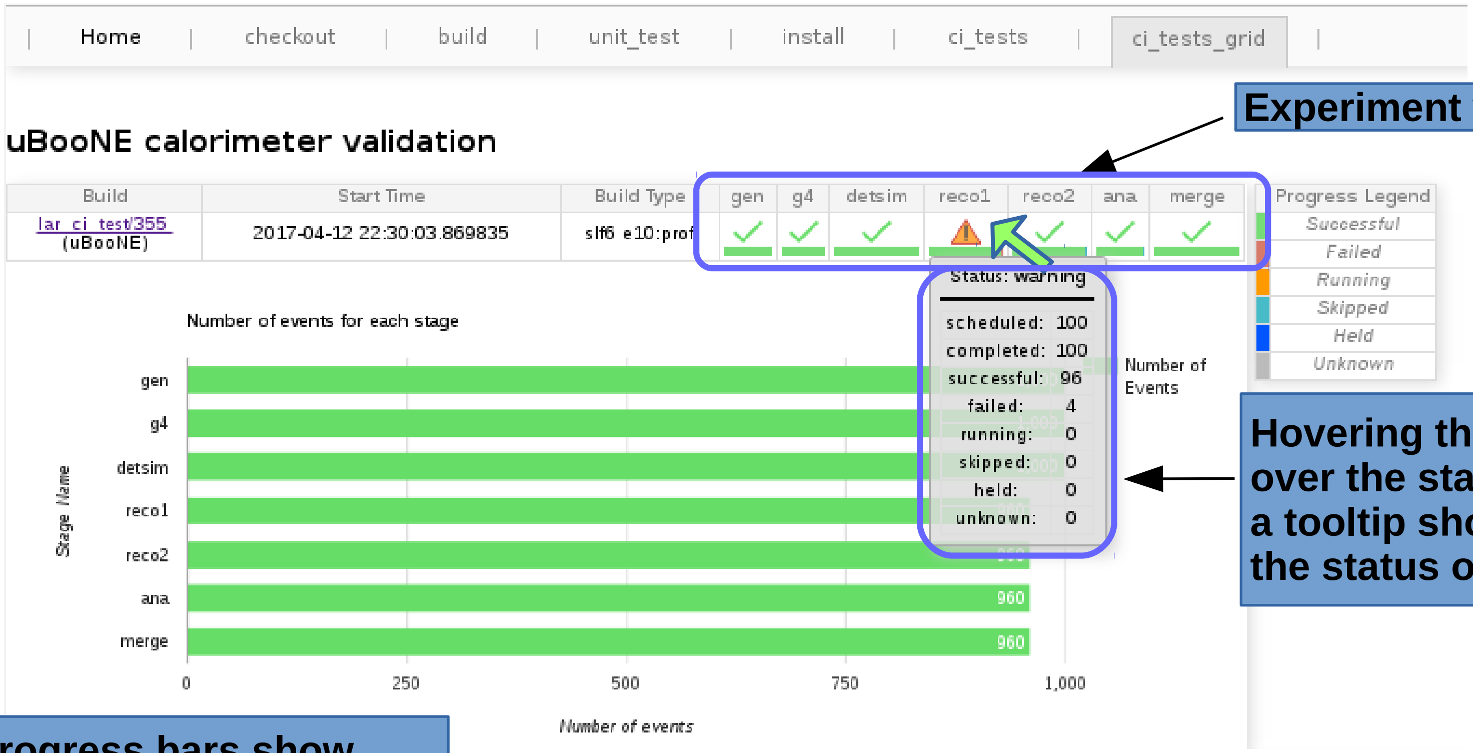
# CI Web application for monitoring

- Graph of RSS memory peak:
uboonecode g4 stage as an example



# CI validation view

- **uBooNE calorimeter validation as an example**



| Home | checkout | build | unit_test | install | ci_tests | ci_tests_grid |

## uBooNE calorimeter validation

| Build | Start Time | Build Type | gen | g4 | detsim | reco1 | reco2 | ana | merge |
|---|---|---|---|---|---|---|---|---|---|
| lar_ci_test/355 (uBooNE) | 2017-04-12 22:30:03.869835 | slf6 e10:prof | ✓ | ✓ | ✓ | ⚠ | ✓ | ✓ | ✓ |

| Progress Legend |
|---|
| 🟩 Successful |
| 🟥 Failed |
| 🟧 Running |
| 🟦 Skipped |
| 🟦 Held |
| ⬜ Unknown |

Status: warning

| | |
|---|---|
| scheduled: | 100 |
| completed: | 100 |
| successful: | 96 |
| failed: | 4 |
| running: | 0 |
| skipped: | 0 |
| held: | 0 |
| unknown: | 0 |

**Experiment workflow stages**

**Hovering the mouse over the stages box a tooltip shows the status of that stage jobs**

**Progress bars show the number of events available for each stage**

- **The CI validation can process a workflow with as many stages as needed**
- **The stages can be grouped together in the same grid job to minimize I/O and improve grid job efficiency**