

LArSoft CI System Overview

LArSoft Workshop

June 20, 2017

**Vito Di Benedetto
(Fermilab)**

**CI Team
Vito Di Benedetto
Michele Fattoruso
Vladimir Podstavkov**

Outline

- Introduction
- **Specifying and configuring tests**
- Running tests
- **Obtain results**
- Upcoming new features
- **Getting help**

Introduction

- **Continuous Integration (CI)**
- **why Continuous Integration**
- **definitions**
- **CI system overview**

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.

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.



Introduction: definitions

- **CI phase**

- Is a single activity executed by the CI system
- Examples are: checkout, build, install, ...

- **CI workflow**

- Is a collection of CI phases
- There are CI workflows used to test experiment code standalone, to update reference files used by CI tests, and so on

- **CI build**

- Is the job that executes a specific CI workflow on a build node

Introduction: definitions

- **Unit test**

- Is a test that verifies a single “unit” or logical concept of the system
- Is fully automated (you want be able to run the test in an automatic procedure)
- Is independent (you want be able to run the test in parallel)
- Runs fast (you want quick feedback)
- Is trustworthy (if the test fails you know that the code is broken)
- It is good practice to have at least one unit test for each functionality of the system

Introduction: definitions

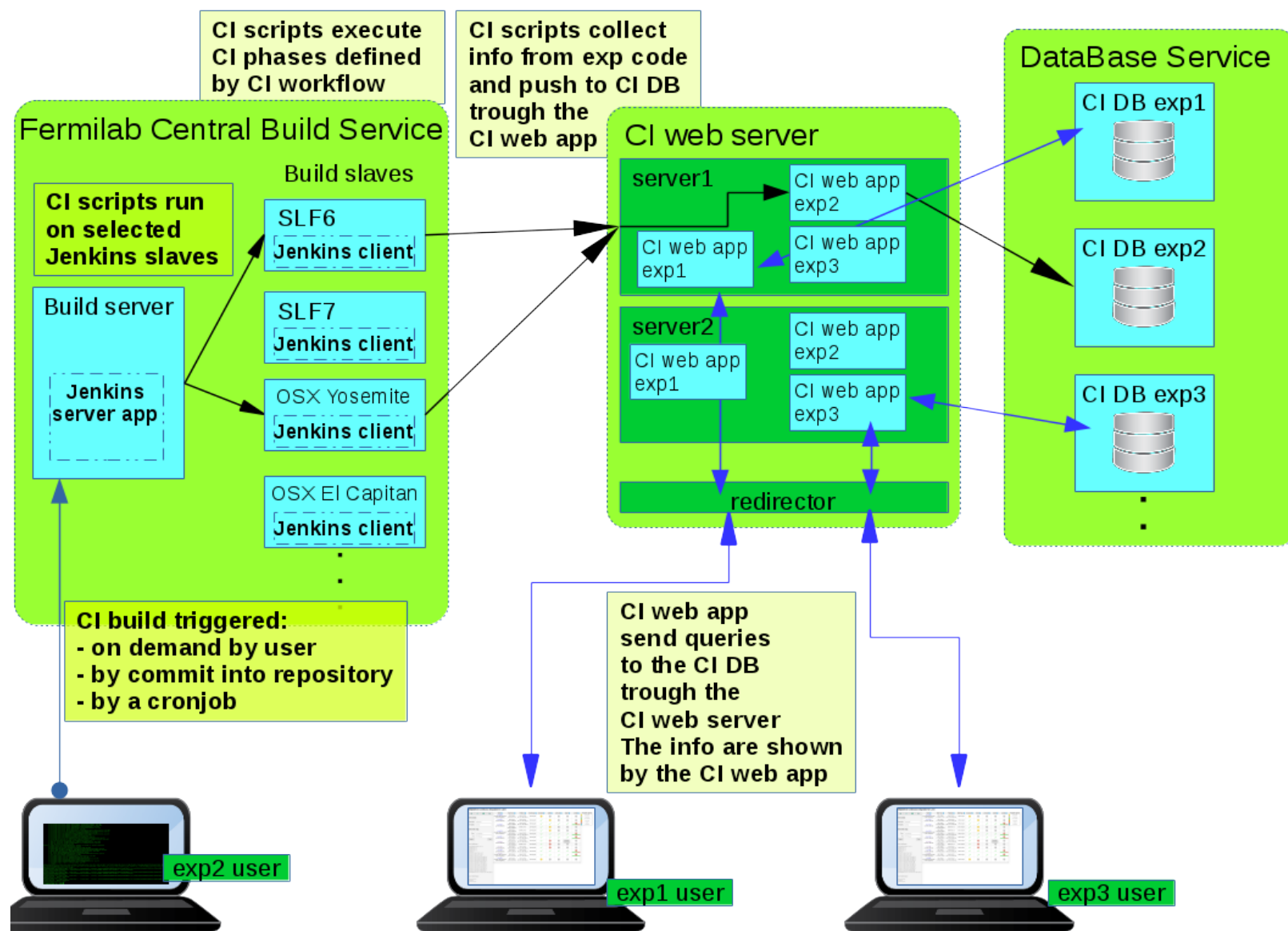
- **CI test or integration test**

- Is the logical extension of unit test
- Tests the behavior of “combined units” of the system
- It verifies that the (major) parts of the system work well together
- You can have:
 - **Regression test:** runs existing CI tests against modified code to make sure that what used to work doesn't break
 - **Reproducibility test:** makes sure that running the code using the same input **always** generate the same output
 - **Back-compatibility test:** make sure that new code is able to access data files produced by previous code releases
 - **Validation test:** make sure that new code produce meaningful results

- **Test suite**

- Is a collection (suite) of CI tests

Introduction: CI system overview



Specifying and configuring tests

- **CI workflow configuration**
- **Integration test configuration**
- **CI validation and grid support**

CI workflow configuration

More details at <https://cdcvcs.fnal.gov/redmine/projects/ci/wiki/Workflowcfg>

The CI workflow configuration drives all the actions to be performed by the CI build

Excerpt of the CI workflow configuration

- The CI workflow configuration has three basic sections:

- **default**: global section, selects the **CI workflow** to use, set the report mailing list, proxy VO, ...

- **workflow**: defines **personality**, list of **CI tests** to run, list of **CI phases**, code modules to test, **CI validation config file** (*grid_cfg*), ...

- **personality**: defines what each **CI phase** does

```
[default]
workflow      = ${LAR_WORKFLOW:-defaultwf}
notify_email_to = larsoft_build@fnal.gov
proxy_vo      = /fermilab/uboone
build_db_uri   = http://dbweb6.fnal.gov:8080/LarCI/app
```

```
[defaultwf] ### workflow section
experiment    = LArSoft uBooNE DUNE LArIAT ArgoNeuT
qualifier     = ${LAR_QUALS:-e14:prof}
ci_test_lists = quick_test_uboonecode quick_test_dunetpc
               quick_test_lariatsoft quick_test_argoneutcode
personality   = mrb
proxy_flag    = true
phases        = _eval_n checkout_x_modules build unit_test install ci_tests
grid_cfg      = ${LAR_GRIDWFCFG:-cfg/grid_workflow.cfg}
modules       = larsoft ... uboonecode dunetpc lariatsoft argoneutcode
```

```
[mrb] ### personality section
# define what the CI phases do:
# _eval_n: setup the code environment
...
#checkout: instruction to checkout the code
...
#build: instruction to build the code
...
# unit_test: instruction to run unit tests
...
# install: instruction to install the code
...
#ci_tests: instruction to run the CI tests
```

Integration test configuration

- The CI test configuration file is a INI-formatted file
 - It is parsed using the ***ConfigParser*** python module
- It is located in the experiment code repository at:
<exp code>/test/ci/ci_tests.cfg
- It has three type of sections
 - **[DEFAULT]** is used to define global variables that can be used in the test section
 - **[test <testname>]** is used to configure the test
<testname>
 - **[suite <suitename>]** is used to specify a collection of *tests* to run in the same job

Integration test configuration

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

- Basic layout for *ci_tests.cfg* file

Test definition blocks

Definition for a named 'testA' that uses the output from 'testB'

```
[test testA]
script=$<PRODUCT>_DIR/test/testA.sh
args= -a qualA -b qualB
requires= testB
```

Definition for 'testB'

```
[test testB]
script=$<PRODUCT>_DIR/test/testB.sh
...
```

Definition for 'testC'

```
[test testC]
script = $PRODUCT_DIR/test/testC.sh
...
```

Definition of test suite 'test_suiteA'

```
[suite test_suiteA]
testlist = testA testC
```

Test suite definition

Integration test configuration

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

- Basic layout for *ci_tests.cfg* file

```
# Definition for a named 'testA' that uses the output from 'testB'  
[test testA]
```

```
script=$<PRODUCT>_DIR/test/testA.sh
```

```
args= -a qualA -b qualB
```

```
requires= testB
```

Set dependencies
between tests →

```
# Definition for 'testB'
```

```
[test testB]
```

```
script=$<PRODUCT>_DIR/test/testB.sh
```

```
...
```

```
# Definition for 'testC'
```

```
[test testC]
```

```
script = $PRODUCT_DIR/test/testC.sh
```

```
...
```

```
# Definition of test suite 'test_suiteA'
```

```
[suite test_suiteA]
```

```
testlist = testA testC
```

“testB” will run as part
of the suite because it
is declared as
dependency of “testA” →

CI validation and grid support

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

- 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 to use 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
- Ability to track jobs using POMS
https://cdcv.s.fnal.gov/redmine/projects/prod_mgmt_db/wiki
(Production Operations Management Service) a service which will assist the Production Teams and the Analysis groups of the experiments in their scientific computational work

CI validation and grid support

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

- The CI validation phase has its own configuration file to set up
- It lives in *lar_ci/cfg/grid_workflow.cfg*
- 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_dccachedir        = /pnfs/uboone/scratch/users/vito/CI_tests_grid/
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 validation and grid support

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

stages

#jobs, #events/job

dCache area used to store
exp **code** used to be validated,
data, plots and logs produced by
the experiment workflow

mailing list to report the
completion of the CI validation phase

```
[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_dcache_dir        = /pnfs/uboone/scratch/users/vito/CI_tests_grid/
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 validation and grid support

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

Stage FHiCL file

Stage executable

Stage jobs resources:
expected lifetime, memory, disk

```
[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_grid/
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
```

Running tests

More details at https://cdcv.sfnal.gov/redmine/projects/lar_ci/wiki

- There are three methods to test the code
 - **Automatic trigger of a CI build**
 - *Occurs when users **git push** in the develop branch of the repository*
 - Tests the develop branch of LArSoft + experiment
 - Default CI workflow consist of:
 - checkout, build, unit tests (174), CI tests (29 quick CI tests)
 - **Manual trigger of a CI build**
 - Developers can run a script to trigger a CI build to test specific branches
 - Developer can specify branches repository by repository
 - Default branch is “develop”
 - **NOTE: all code used for tests must be committed and pushed to the central repositories (a security requirement)**
 - **Running tests locally**
 - The code can be tested locally to verify that it doesn't break anything
 - This can be achieved using the **test_runner** script
see https://cdcv.sfnal.gov/redmine/projects/lar_ci/wiki for more details

Obtaining results

- **The CI web application components**
- **Which information you can get**
- **Email report**

Obtaining results

- The CI web application is a web interface that easily provides information about the status of the code.

It is available at the URL: <http://lar-ci-history.fnal.gov/LarCI/app>

Home

Multiplatform Continuous Integration for LarCI

Select builds:

From build:

of builds:

Select date range:

From:

To:

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

- In the next few slides we will walk through the CI web interface
- I'll describe:
 - its different components
 - the available information you can access

Wiki documentation available at

https://cdcvcs.fnal.gov/redmine/projects/lar_ci/wiki#How-to-monitor-the-status-of-your-build

Obtaining results

- In-line documentation:

Home

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

Multiplatform Continuous Integration for LarCI

Select builds:

From build:

of builds:

Select date range:

From:

To:

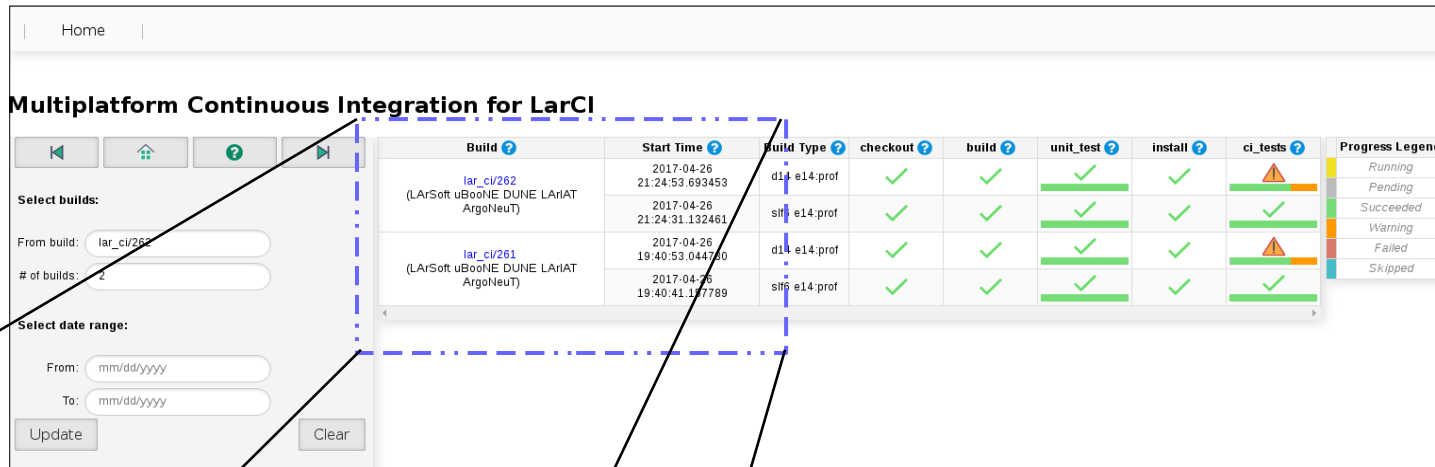
Build ?	Start Time ?	Build Type ?	checkout ?	build ?	unit_test ?	install ?	ci_tests ?	Progress Legend
lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT)	2017-04-26 21:24:53.693453	d14 e14:prof	✓	✓	✓	✓	⚠	Running
	2017-04-26 21:24:31.132461	slf6 e14:prof	✓	✓	✓	✓	✓	Succeeded
lar_ci/261 (LArSoft uBooNE DUNE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

Progress Legend:

- Running
- Pending
- Succeeded
- Warning
- Failed
- Skipped

Obtaining results

- CI Build details



Multiplatform Continuous Integration for LarCI

Select builds:

From build:

of builds:

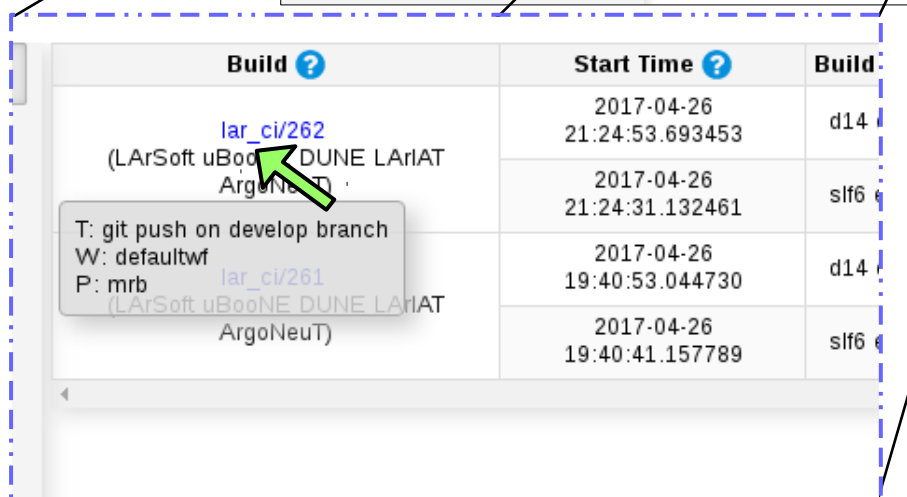
Select date range:

From:

To:

Update Clear

Build ?	Start Time ?	Build Type ?	checkout ?	build ?	unit_test ?	install ?	ci_tests ?	Progress Legend
lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 21:24:53.693453	d14_e14:prof	✓	✓	✓	✓	⚠	Running
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 21:24:31.132461	slf6_e14:prof	✓	✓	✓	✓	✓	Pending
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 19:40:53.044730	d14_e14:prof	✓	✓	✓	✓	⚠	Succeeded
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 19:40:41.157789	slf6_e14:prof	✓	✓	✓	✓	✓	Warning



Build ?	Start Time ?	Build
lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 21:24:53.693453	d14
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 21:24:31.132461	slf6
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 19:40:53.044730	d14
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 19:40:41.157789	slf6

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

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

Obtaining results

• Checkout details

The screenshot displays the 'Multiplatform Continuous Integration for LarCI' web interface. It features a sidebar with navigation links (Home, Build, Checkout, Unit Test, Install, CI Tests) and a main table of build results. The table has columns for Build Type, checkout, build, unit_test, install, and ci_tests. A tooltip is shown over the 'checkout' column for the 'slf6 e14:prof' build type, displaying a list of repository names and their corresponding git descriptions.

Build Table Data:

Build Type	checkout	build	unit_test	install	ci_tests
d14 e14:prof	✓	✓	✓	✓	⚠
slf6 e14:prof	✓	✓	✓	✓	✓
d14 e14:prof	✓	✓	✓	✓	⚠
slf6 e14:prof	✓	✓	✓	✓	✓

Checkout Details Tooltip:

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

Obtaining results

- Unit test details

The screenshot displays the 'Multiplatform Continuous Integration for LarCI' web interface. It features a sidebar with navigation links (Home, Build, Unit Test, Install, CI Tests) and a main table of build results. The table has columns for Build, Start Time, Build Type, Checkout, build, unit_test, install, ci_tests, and Progress Legend. A tooltip is shown over the 'unit_test' column for the first build, displaying the status 'ok' and statistics: 'total:168, succeeded:168, failed:0, skipped:0'.

Build	Start Time	Build Type	Checkout	build	unit_test	install	ci_tests	Progress Legend
lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 21:24:53.693453	d14 e14:prof	✓	✓	✓	✓	⚠	Running
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeUT)	2017-04-26 21:24:51.132461	slf6 e14:prof	✓	✓	✓	✓	✓	Pending
	2017-04-26 19:40:53.044730	d14 e14:prof	✓	✓	✓	✓	⚠	Succeeded
	2017-04-26 19:40:41.157789	slf6 e14:prof	✓	✓	✓	✓	✓	Warning

Unit Test Tooltip:

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.

Obtaining results

- CI tests details

The screenshot displays the 'Multiplatform Continuous Integration for LarCI' interface. It features a table of build results with columns for Build, Start Time, Build Type, checkout, build, unit_test, install, ci_tests, and a Progress Legend. A tooltip is shown for the 'ci_tests' column, providing a detailed status summary and a list of individual test results.

Build Results Table:

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

Progress Legend:

- Running
- Pending
- Succeeded
- Warning
- Failed
- Skipped

ci_tests Tooltip:

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_u Boonecode
- ci_reco2_regression_test_u Boonecode
- ci_reco_regression_test_dune35t
- ci_reco_regression_test_dunefd
- ci_reco_regression_test_protoDUNE
- ci_sim_regression_test_argoneutcode
- ci_bireco_RUN1_regression_test_lariatsoft
- ci_bireco_RUN2_regression_test_lariatsoft
- ci_detsim_regression_test_dune35t
- ci_detsim_regression_test_dunefd
- ci_detsim_regression_test_protoDUNE
- ci_detsim_regression_test_u Boonecode
- ci_gen_regression_test_dune35t
- ci_gen_regression_test_dunefd
- ci_gen_regression_test_protoDUNE
- ci_gen_regression_test_u Boonecode
- ci_mergeana_regression_test_dune35t
- ci_mergeana_regression_test_dunefd
- ci_mergeana_regression_test_protoDUNE
- ci_mergeana_regression_test_u Boonecode
- ci_reco1_regression_test_u Boonecode
- 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.

Obtaining results: CI tests view

- The “CI tests view”
- Which information you can get here

Home

Multiplatform Continuous Integration for LarCI

◀ ⏏ ? ▶

Select builds:

From build:

of builds:

Select date range:

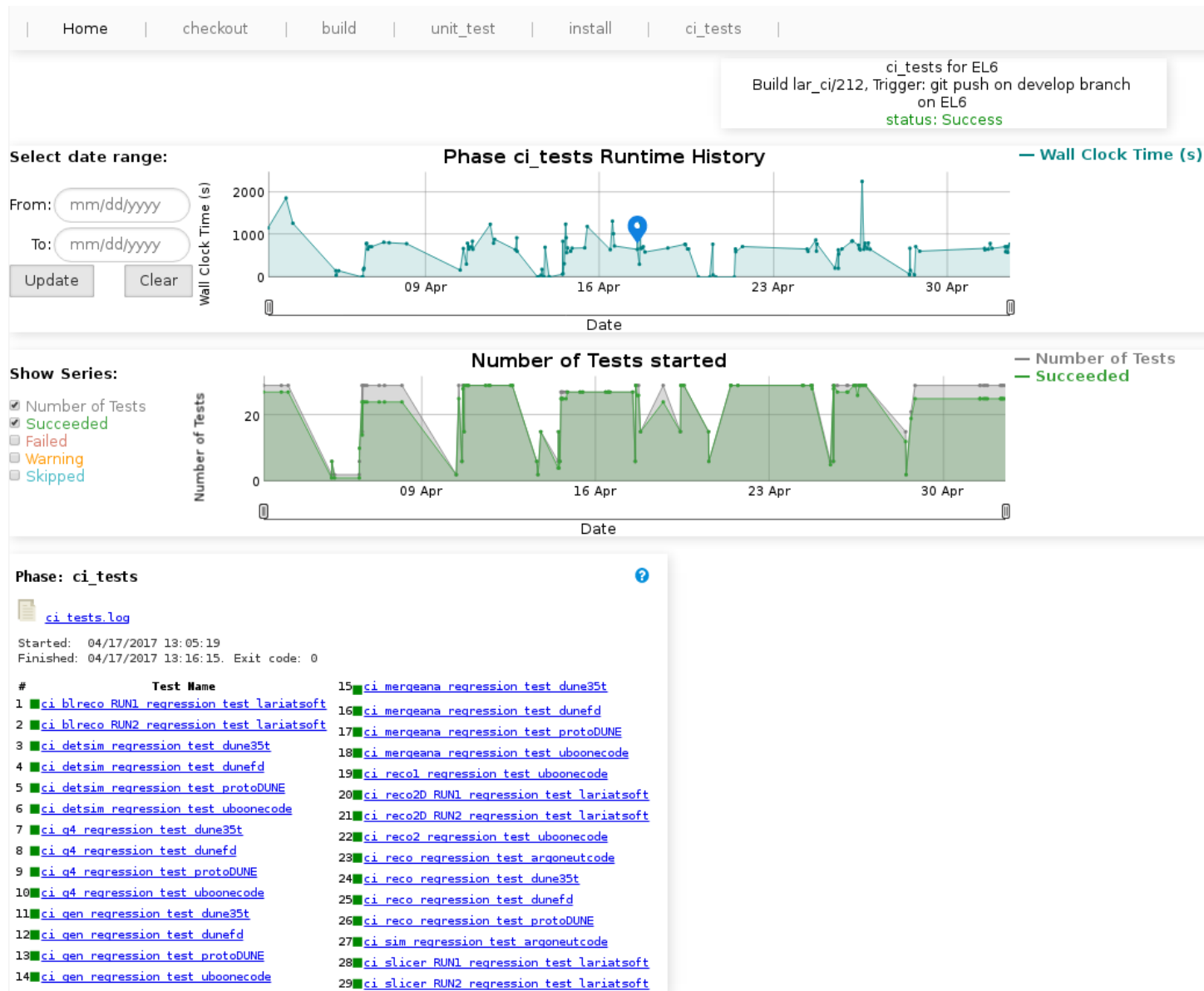
From:

To:

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

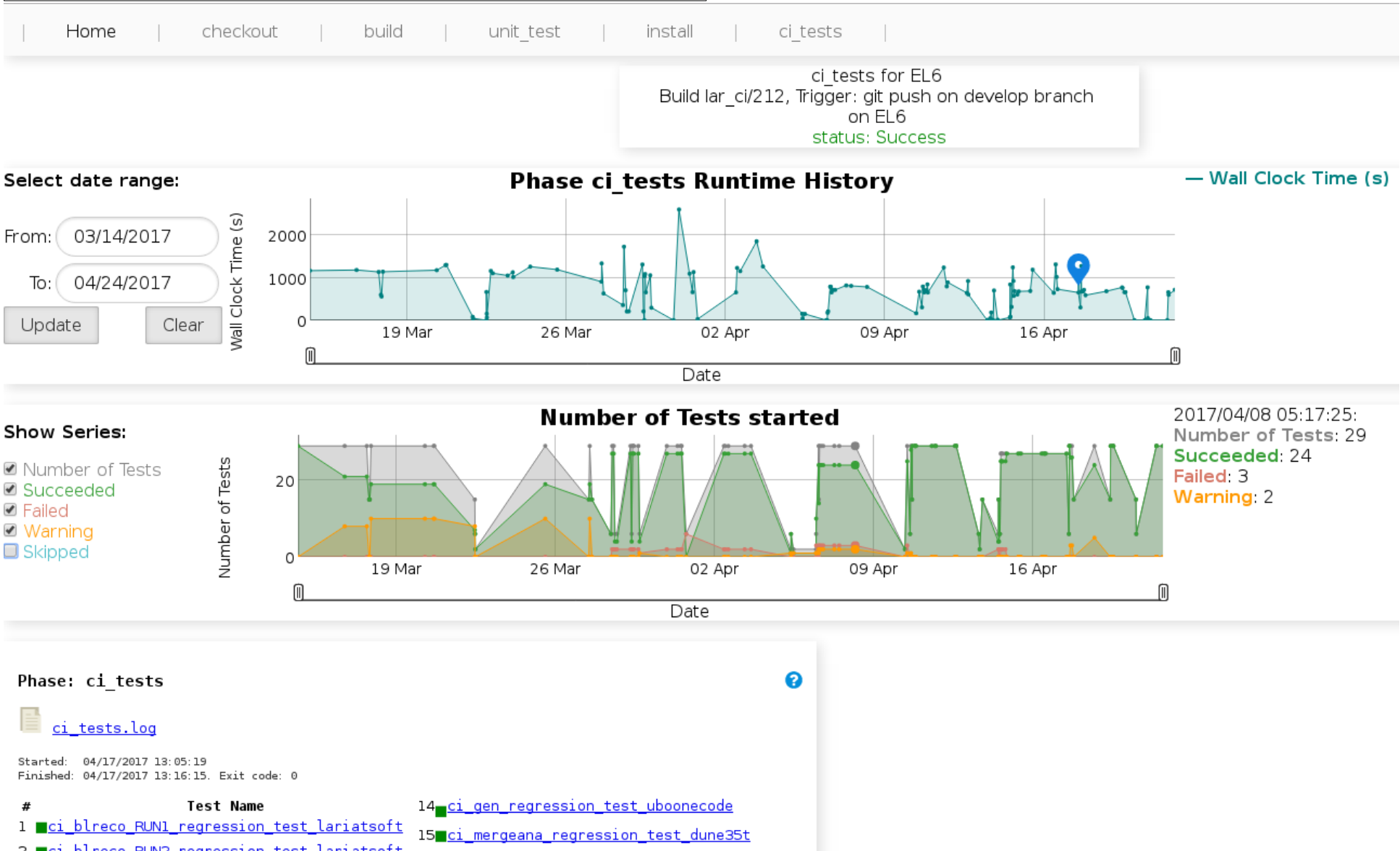
To access the CI tests view, click on the **ci_tests** box

Obtaining results: CI tests view

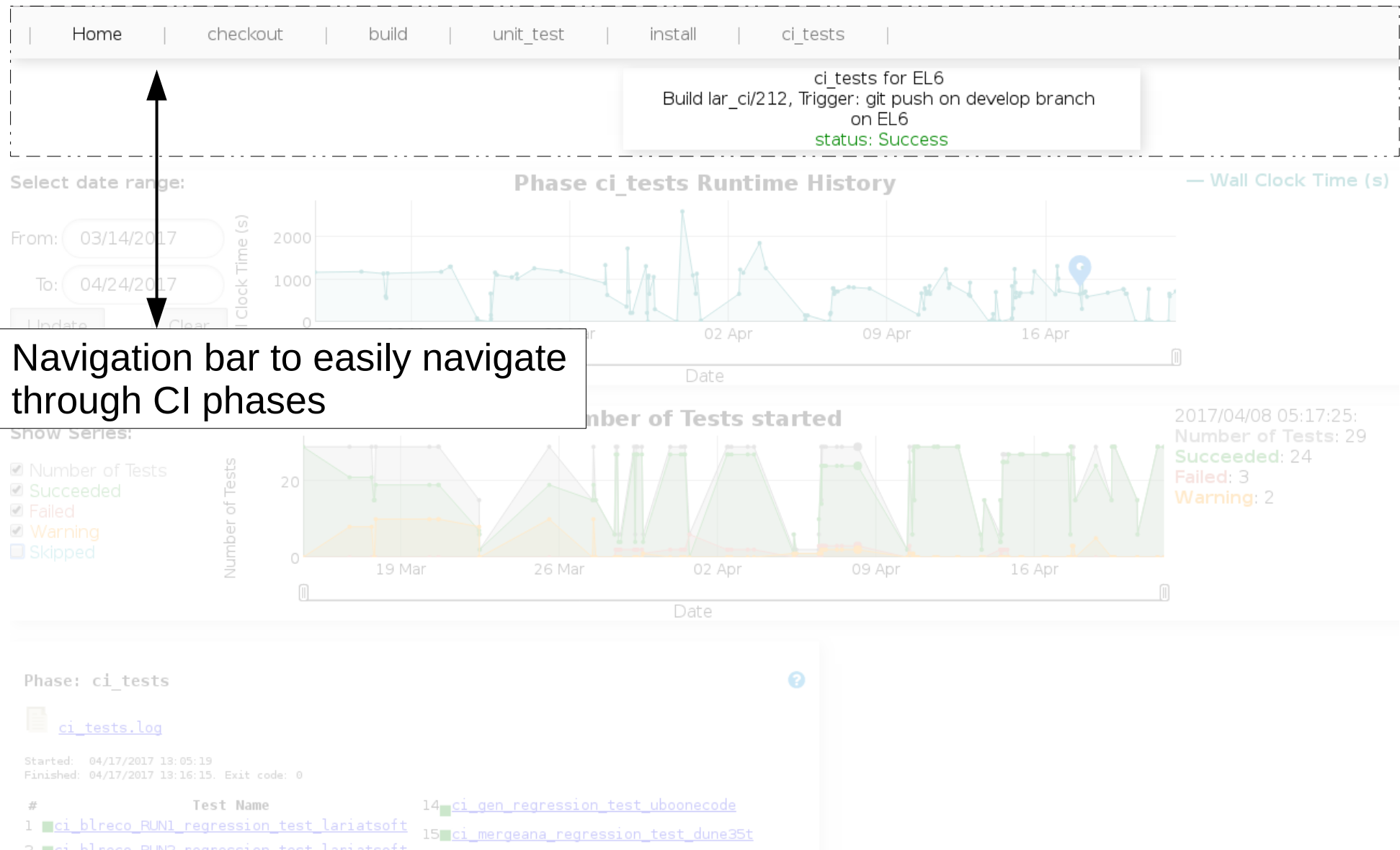


Obtaining results: CI tests view

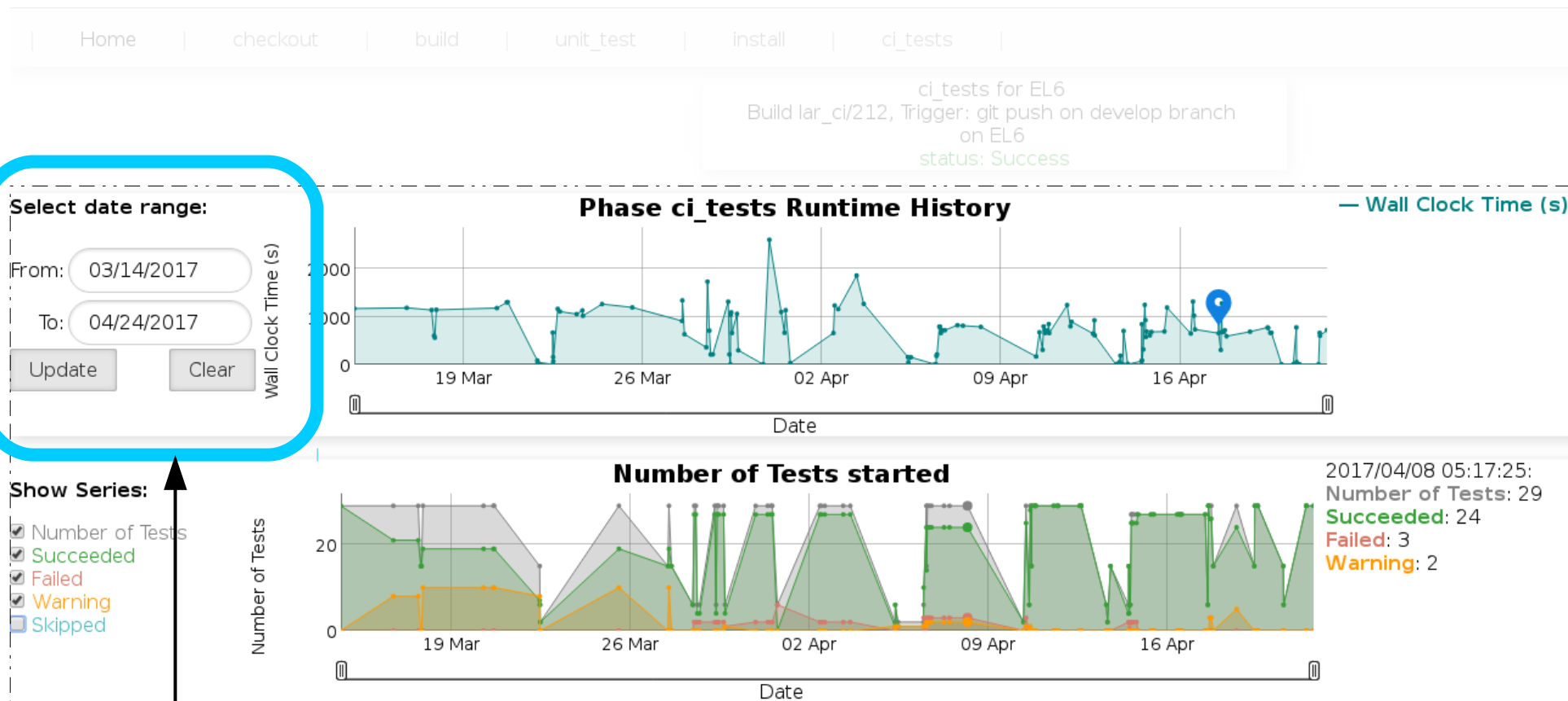
Zoom out the view to easily see details



Obtaining results: CI tests view



Obtaining results: CI tests view



Time picker to select data range

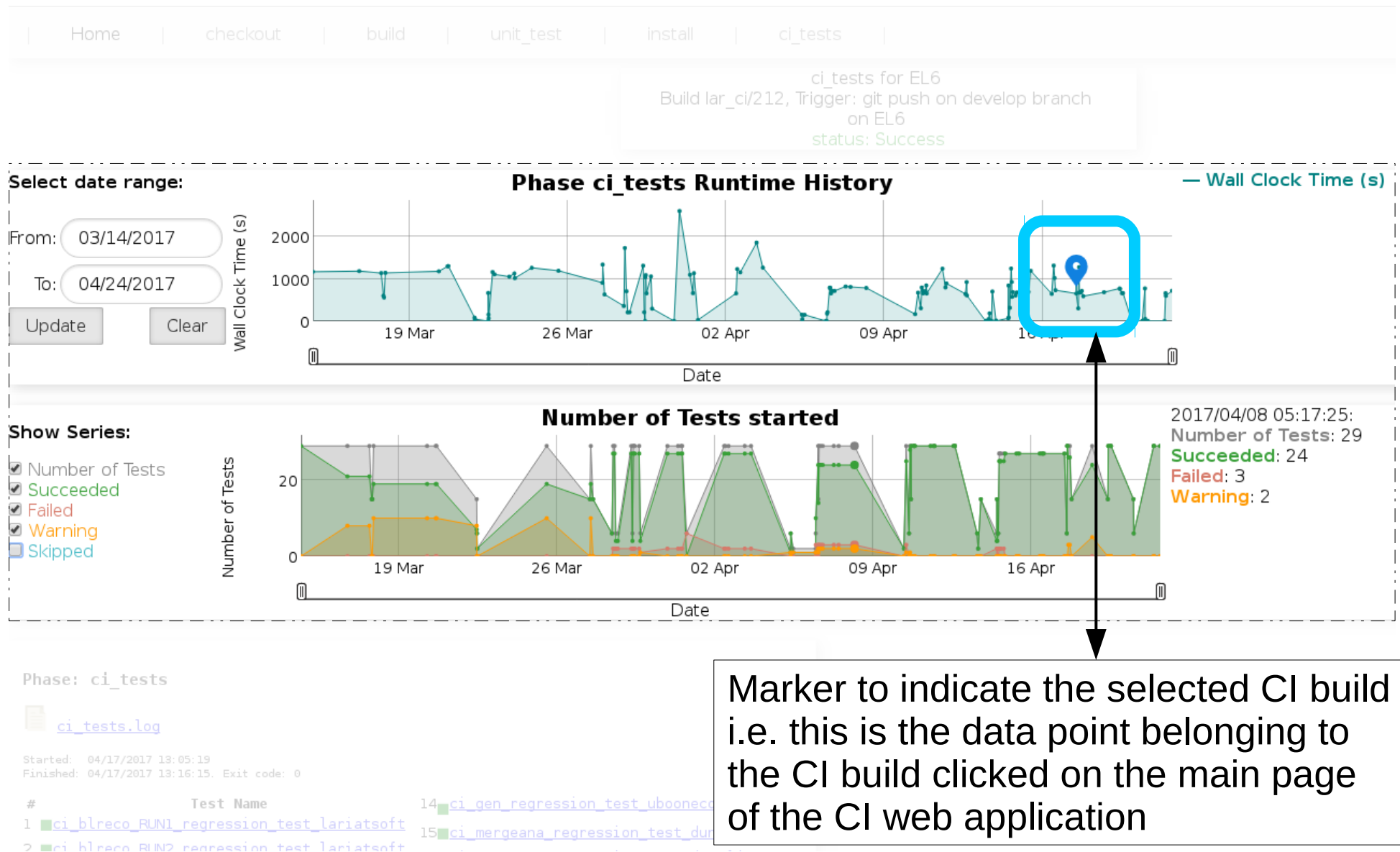
ci_tests.log

Started: 04/17/2017 13:05:19

Finished: 04/17/2017 13:16:15. Exit code: 0

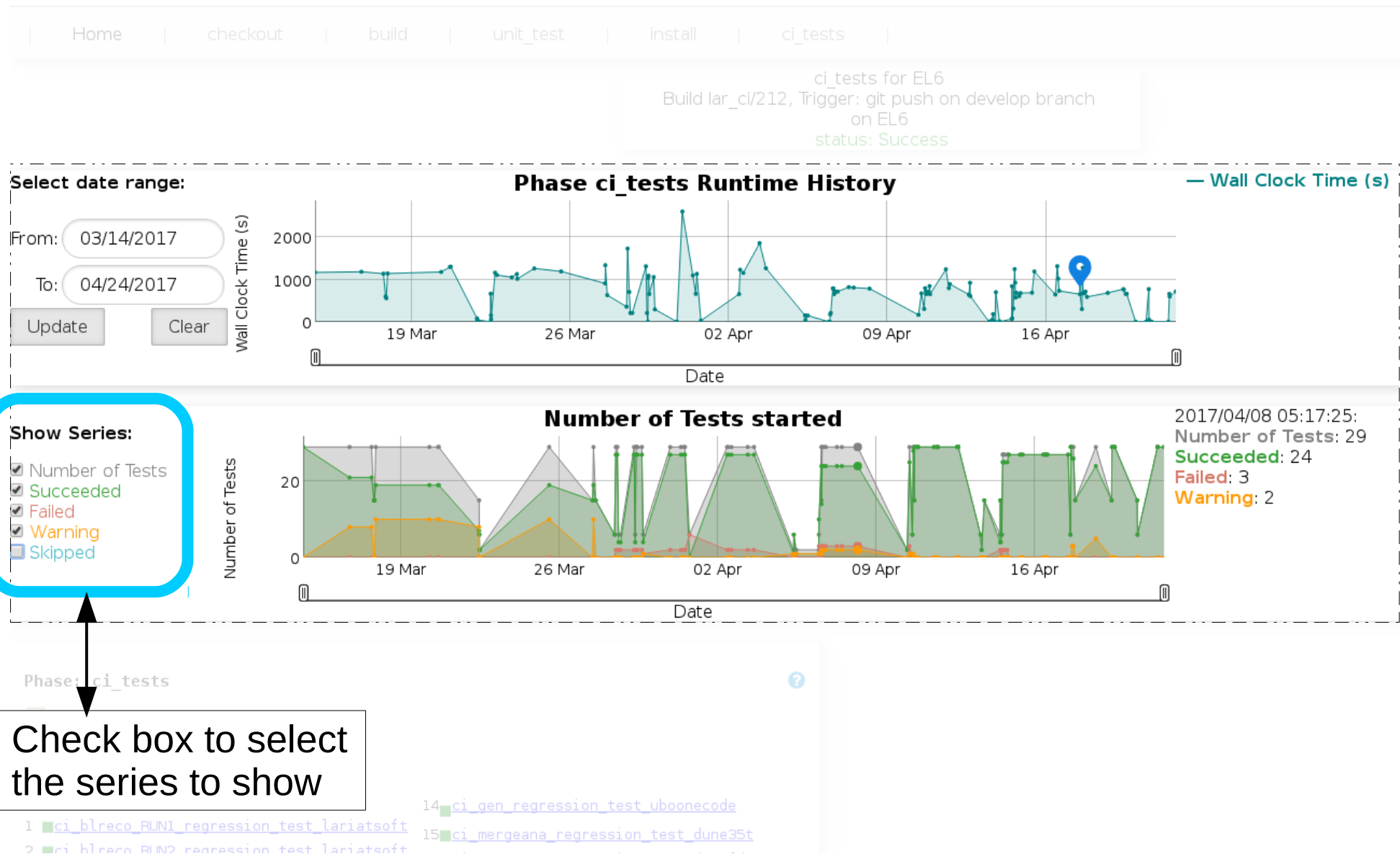
```
# Test Name
1 ci_hlreco_RUN1_regression_test_lariatsoft
2 ci_hlreco_RUN2_regression_test_lariatsoft
14 ci_gen_regression_test_uboonecode
15 ci_mergeana_regression_test_dune35t
```

Obtaining results: CI tests view



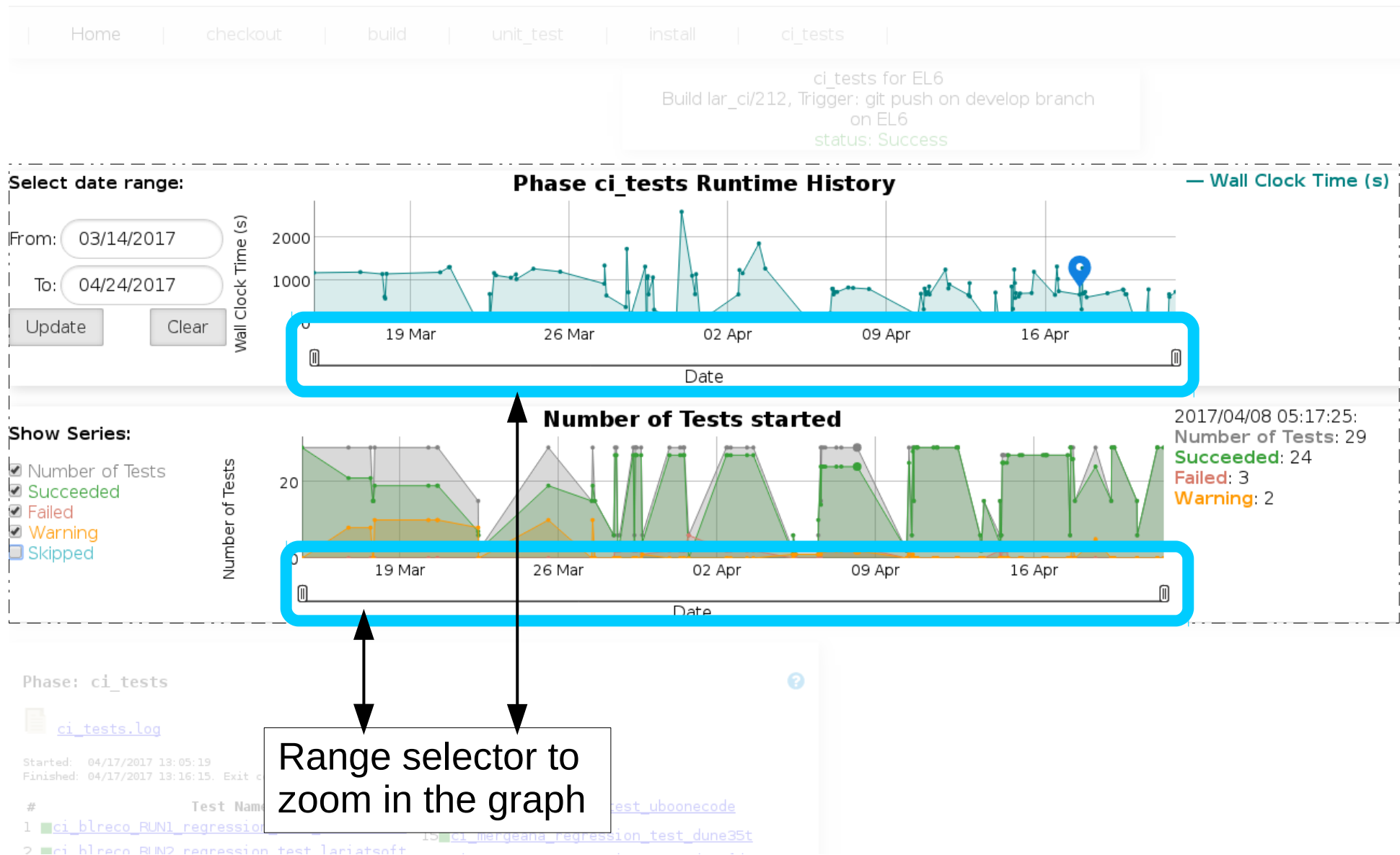
Marker to indicate the selected CI build
i.e. this is the data point belonging to
the CI build clicked on the main page
of the CI web application

Obtaining results: CI tests view



Check box to select the series to show

Obtaining results: CI tests view



Obtaining results: CI tests view

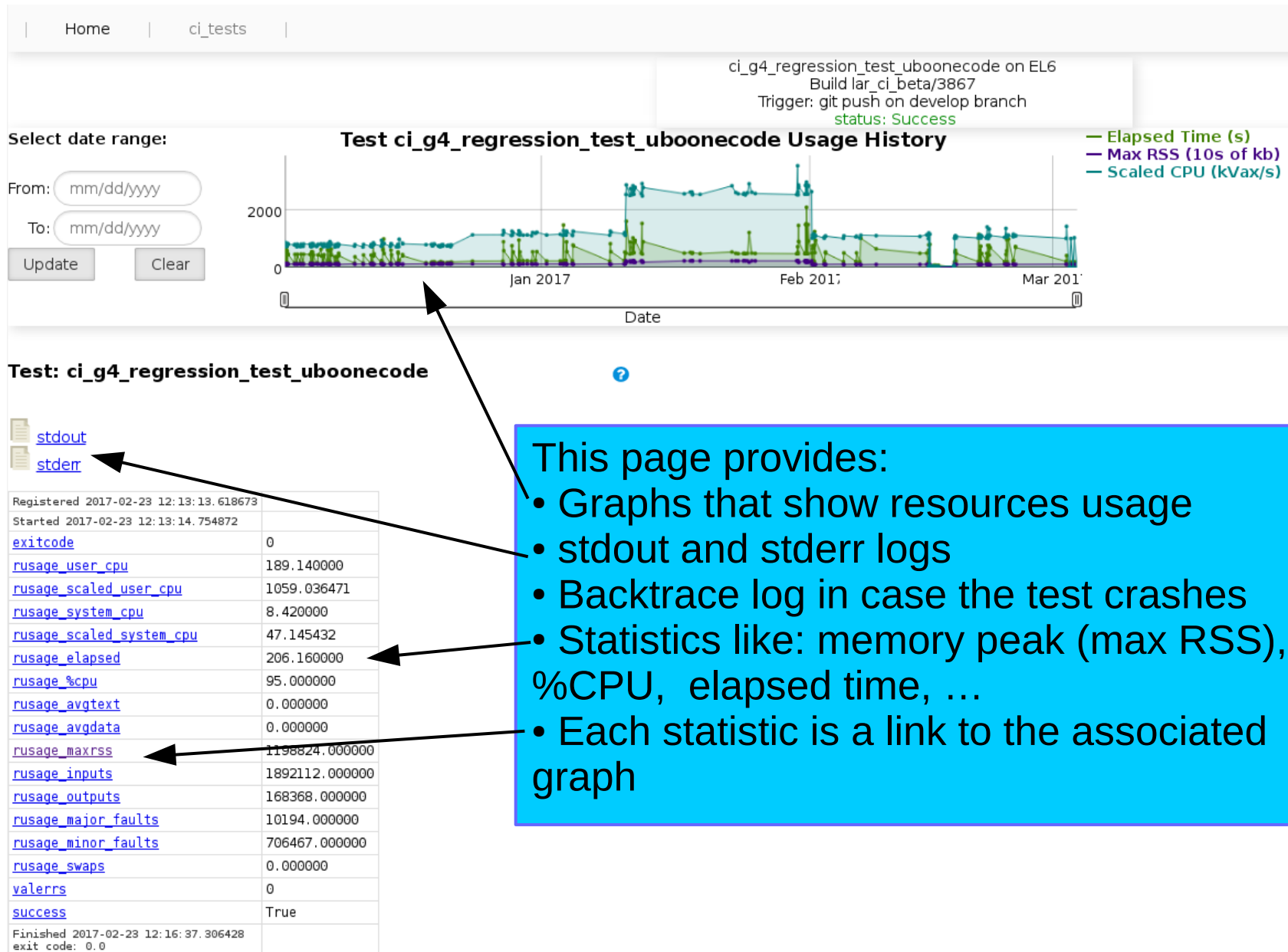
- Access specific CI test logs and stats



Each CI phase has its own log

To access ci test details
click on the specific ci test link

Obtaining results: CI tests details

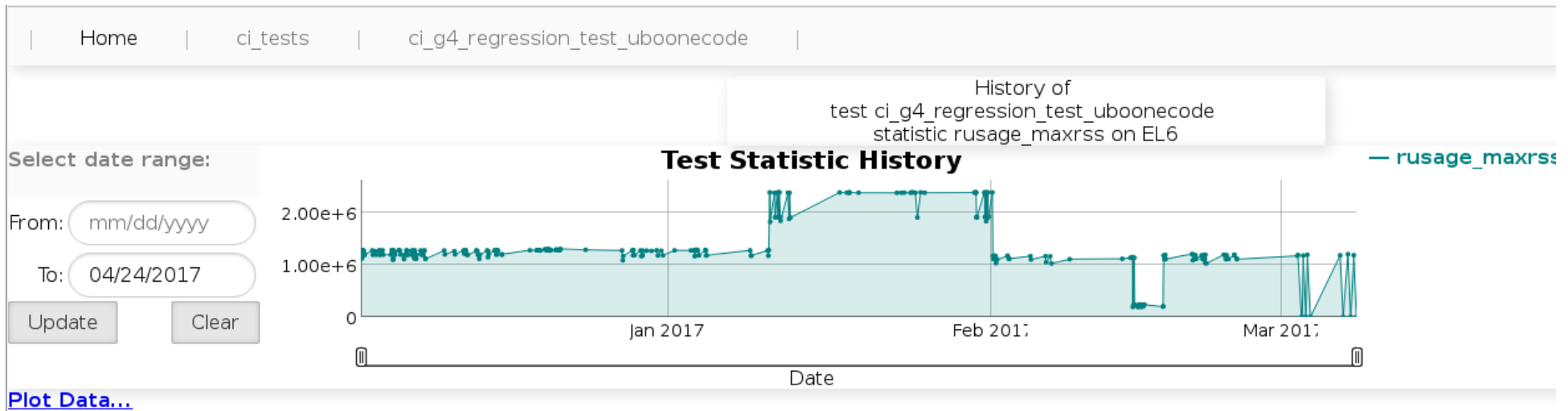


This page provides:

- 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

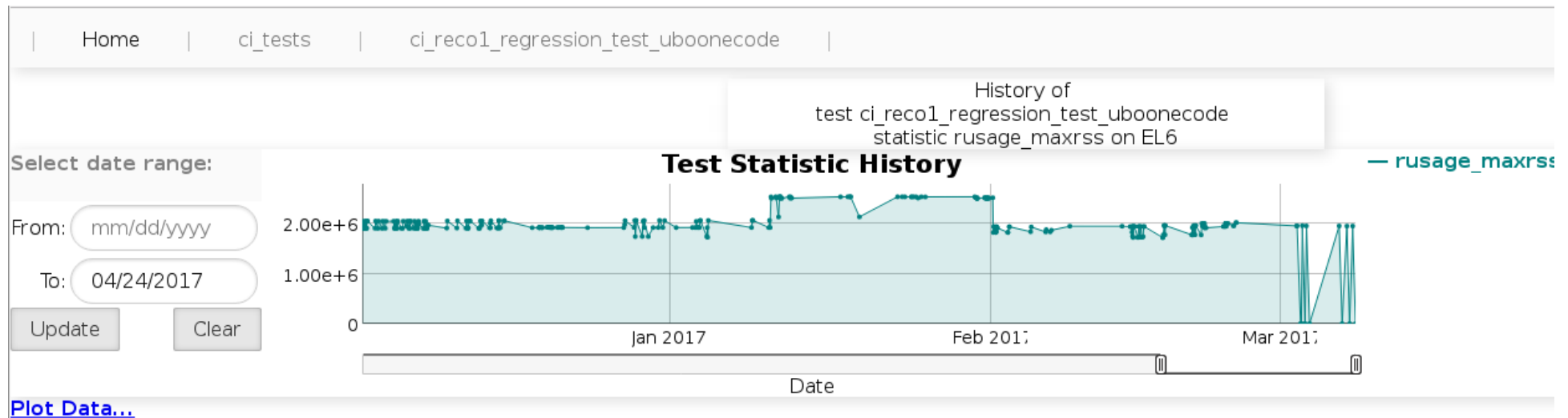
Obtaining results: CI tests details

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



Obtaining results: CI tests details

- Graph of RSS memory peak: uboonecode reco stage 1 as an example



Obtaining results: CI tests view

- The list of CI tests is sorted by status severity
 - The CI test status corresponds to a color:
red=failure; **orange=warning**; **green=succeeded**
- Each CI test link provides logs for the test

```
Phase: ci_tests ?  
  
ci_tests.log  
  
Started: 05/05/2017 17:00:47  
Finished: 05/05/2017 17:15:21. Exit code: 202  
  
#          Test Name  
1 ■ ci_reco_regression_test_dune35t  
2 ■ ci_g4_regression_test_dune35t  
3 ■ ci_g4_regression_test_dunefd  
4 ■ ci_g4_regression_test_protoDUNE  
5 ■ ci_reco_regression_test_dunefd  
6 ■ ci_reco_regression_test_protoDUNE  
7 ■ ci_bireco_RUN1_regression_test_lariatsoft  
8 ■ ci_bireco_RUN2_regression_test_lariatsoft  
9 ■ ci_detsim_regression_test_dune35t  
10 ■ ci_detsim_regression_test_dunefd  
11 ■ ci_detsim_regression_test_protoDUNE  
12 ■ ci_detsim_regression_test_uboonecode  
13 ■ ci_g4_regression_test_uboonecode  
14 ■ ci_gen_regression_test_dune35t  
15 ■ ci_gen_regression_test_dunefd  
16 ■ ci_gen_regression_test_protoDUNE  
17 ■ ci_gen_regression_test_uboonecode  
18 ■ ci_mergeana_regression_test_dune35t  
19 ■ ci_mergeana_regression_test_dunefd  
20 ■ ci_mergeana_regression_test_protoDUNE  
21 ■ ci_mergeana_regression_test_uboonecode  
22 ■ ci_reco1_regression_test_uboonecode  
23 ■ ci_reco2D_RUN1_regression_test_lariatsoft  
24 ■ ci_reco2D_RUN2_regression_test_lariatsoft  
25 ■ ci_reco2_regression_test_uboonecode  
26 ■ ci_reco_regression_test_argoneutcode  
27 ■ ci_sim_regression_test_argoneutcode  
28 ■ ci_slicer_RUN1_regression_test_lariatsoft  
29 ■ ci_slicer_RUN2_regression_test_lariatsoft
```

Obtaining results: Unit test view

- The “Unit test view”
- Which information you can get here

Home

Multiplatform Continuous Integration for LarCI

Select builds:

From build:

of builds:

Select date range:

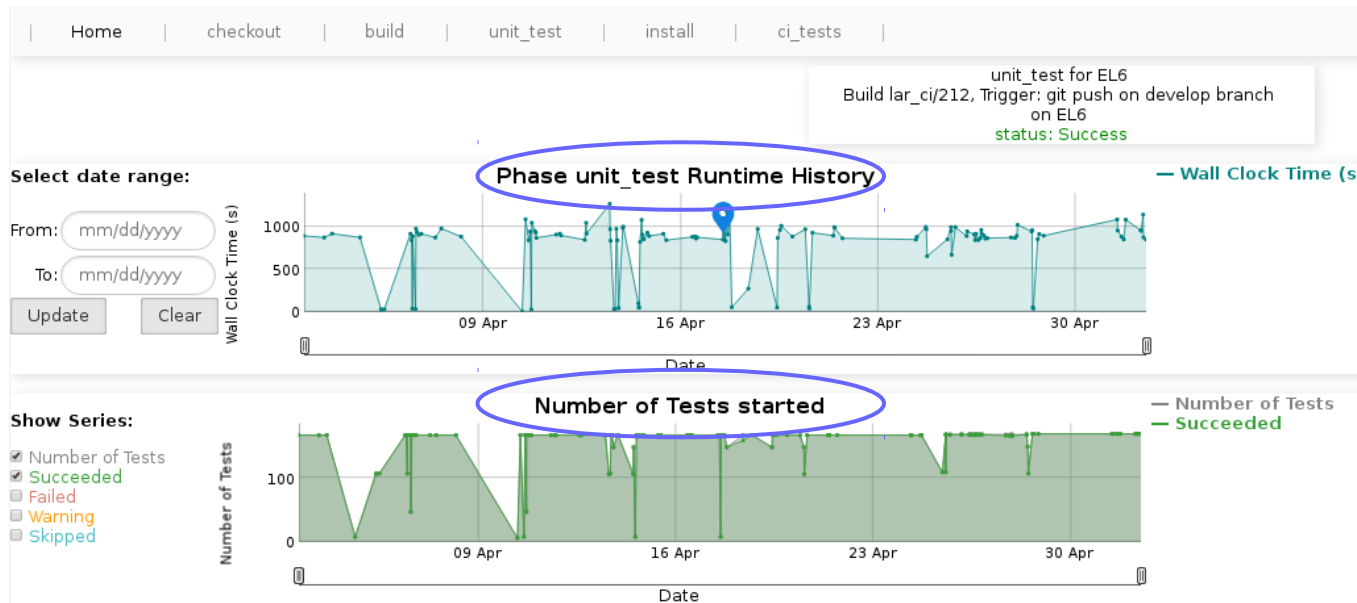
From:

To:

Build ?	Start Time ?	Build Type ?	checkout ?	build ?	unit_test ?	install ?	ci_tests ?	Progress Legend
lar_ci/262 (LArSoft uBooNE DUNE LArIAT ArgoNeuT)	2017-04-26 21:24:53.693453	d14 e14:prof	✓	✓	✓	✓	⚠	Running
	2017-04-26 21:24:31.132461	slf6 e14:prof	✓	✓	✓	✓	✓	Pending
lar_ci/261 (LArSoft uBooNE DUNE LArIAT ArgoNeuT)	2017-04-26 19:40:53.044730	d14 e14:prof	✓	✓	✓	✓	⚠	Succeeded
					✓	✓	✓	Warning
						✓	✓	Failed
							✓	Skipped

To access the unit tests view, click on the ***unit_tests*** box

Obtaining results: Unit test view



Logs

Phase: unit_test

[unit_test.log](#)
[CTestCostData.txt](#)
[LastTest.log](#)

Started: 04/17/2017 12:49:19
Finished: 04/17/2017 13:08:28. Exit code: 0

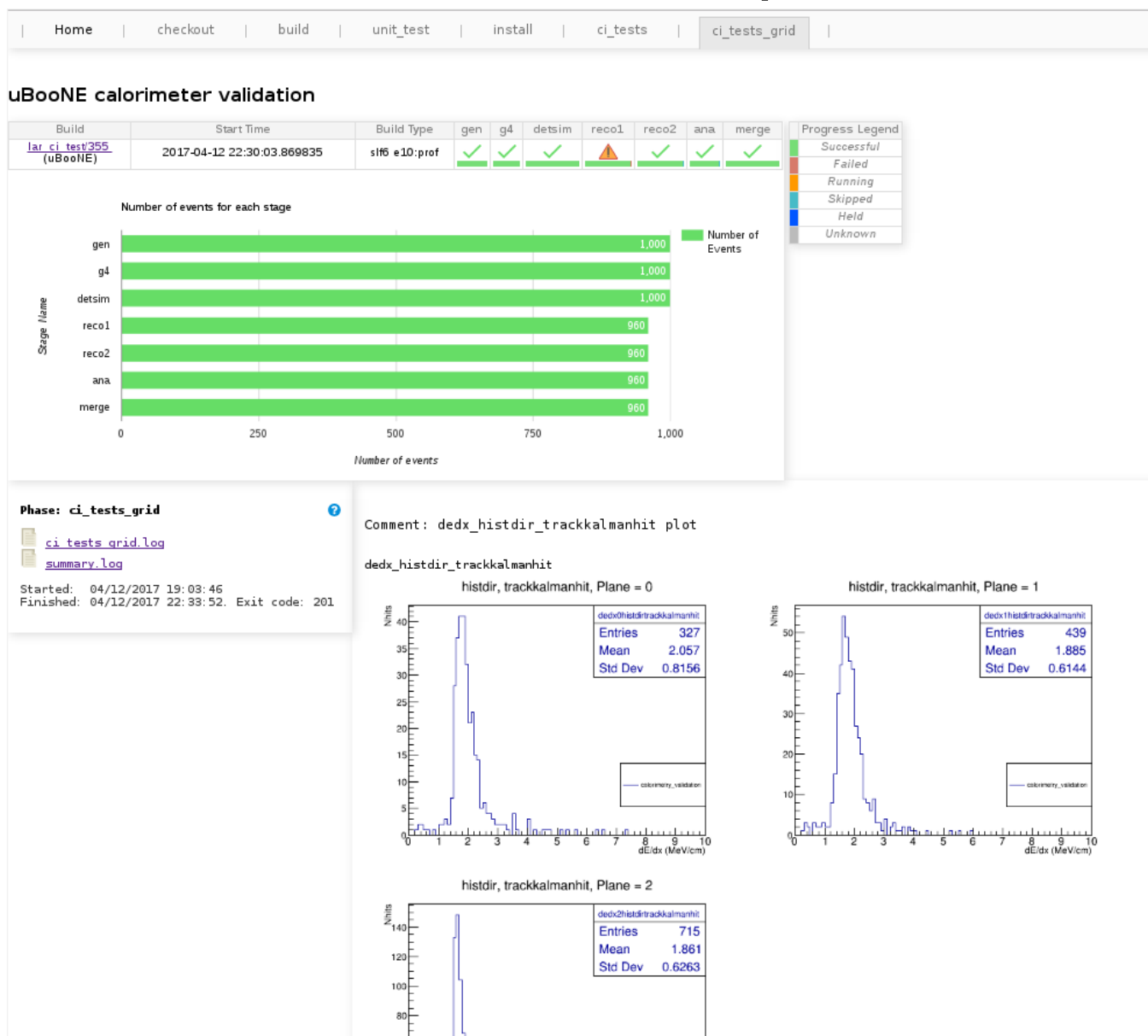
#	Test Name
1	AtomicNumberService test
2	AtomicNumber test
3	BoostedAtomicNumber test
4	BulkAllocator test
5	ChannelStatusServiceTest
6	Cluster test
7	CountersMap test
8	CreateTestShowerCalibrationFromPID test
9	Decomposer test
10	DereferenceIterator test
11	Dereference test
12	DetactorClocks test

Stats

Start Time	Duration (s)	Test Name	Start Time	Duration (s)	Test Name
04/17/2017 12:49:45	5.77	donothing_dune35t	04/17/2017 12:50:04	17.75	
04/17/2017 12:49:45	0.57	donothing_dune35t	04/17/2017 12:50:06	18.81	
04/17/2017 12:49:45	1.26	donothing_simul_dune35t	04/17/2017 12:50:04	17.04	
04/17/2017 12:49:30	21.90	donothing_simul_dune35t	04/17/2017 12:50:06	18.80	
04/17/2017 12:49:44	11.19	dump_channel_map_dune35t test	04/17/2017 12:50:07	15.83	
04/17/2017 12:49:41	1.58	dump_channel_map_dune35t test	04/17/2017 12:50:07	17.32	
04/17/2017 12:49:30	35.17	dump_channel_map test	04/17/2017 12:49:22	13.36	
04/17/2017 12:49:45	2.80	gensingle	04/17/2017 12:49:51	18.08	
04/17/2017 12:49:22	1.30	geo_types test	04/17/2017 12:49:24	0.34	
04/17/2017 12:49:24	1.72	geo_vectors test	04/17/2017 12:49:25	1.34	
04/17/2017 12:49:30	1.81	geometry	04/17/2017 12:49:21	27.40	
04/17/2017 12:49:35	2.41	geometry_argoneut	04/17/2017 12:50:11	8.96	

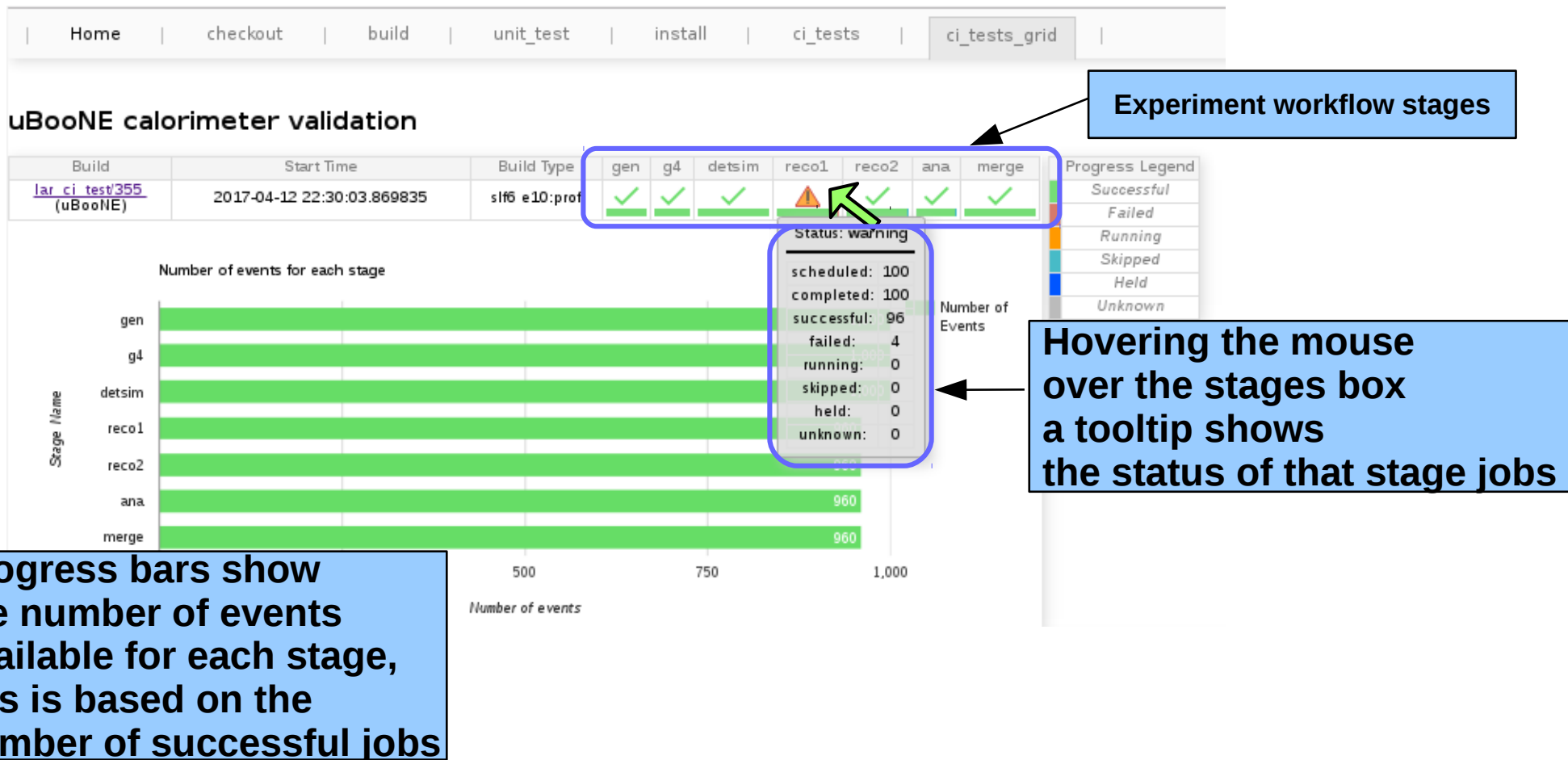
Obtaining results: CI validation view

- uBooNE calorimeter validation as an example



Obtaining results: CI validation view

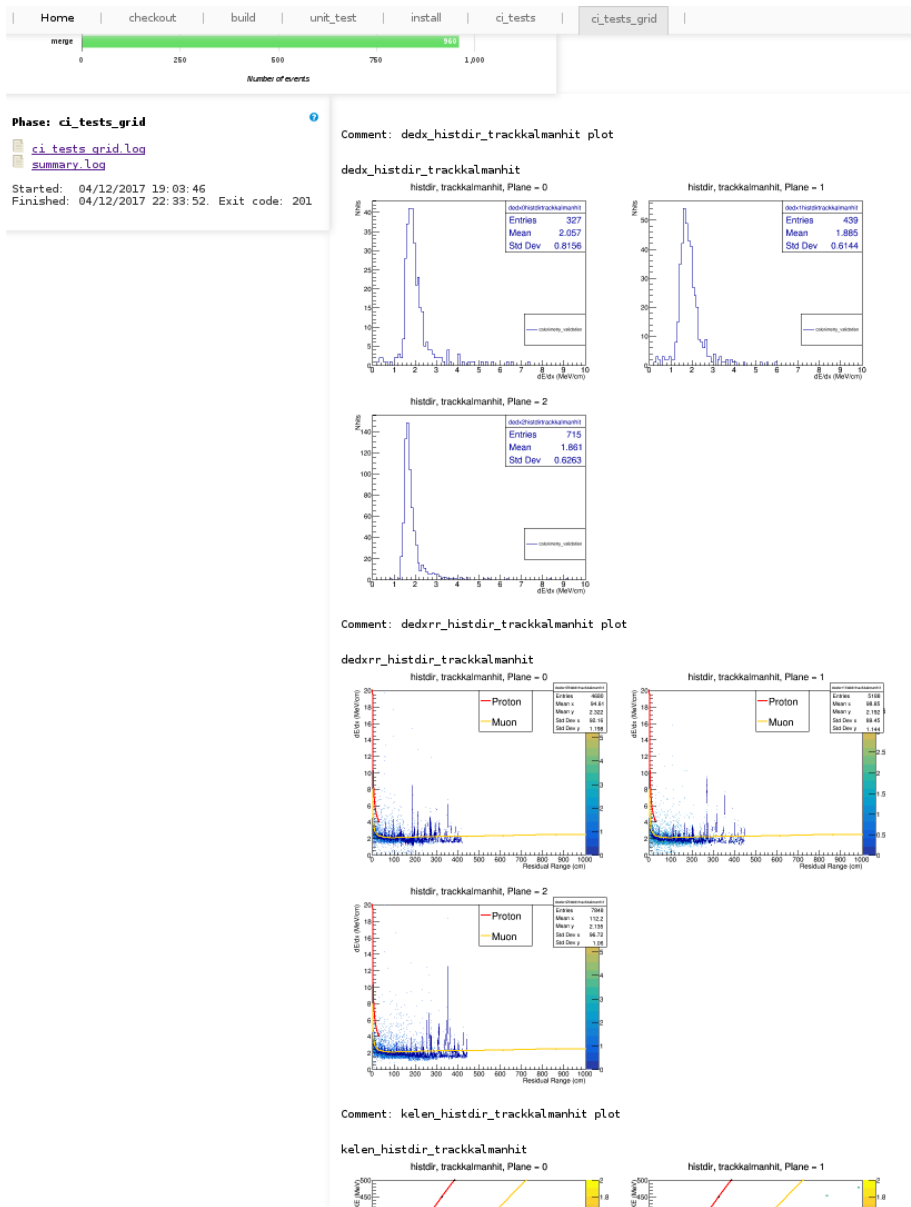
- uBooNE calorimeter validation as an example



- 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

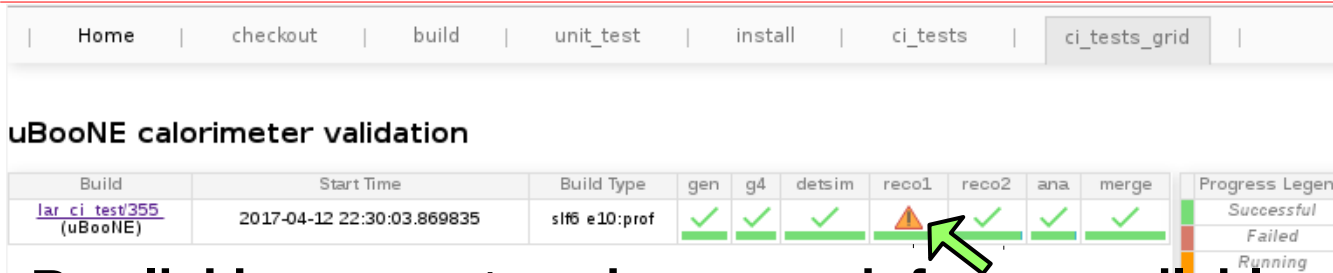
Obtaining results: CI validation view

- uBooNE calorimeter validation as an example

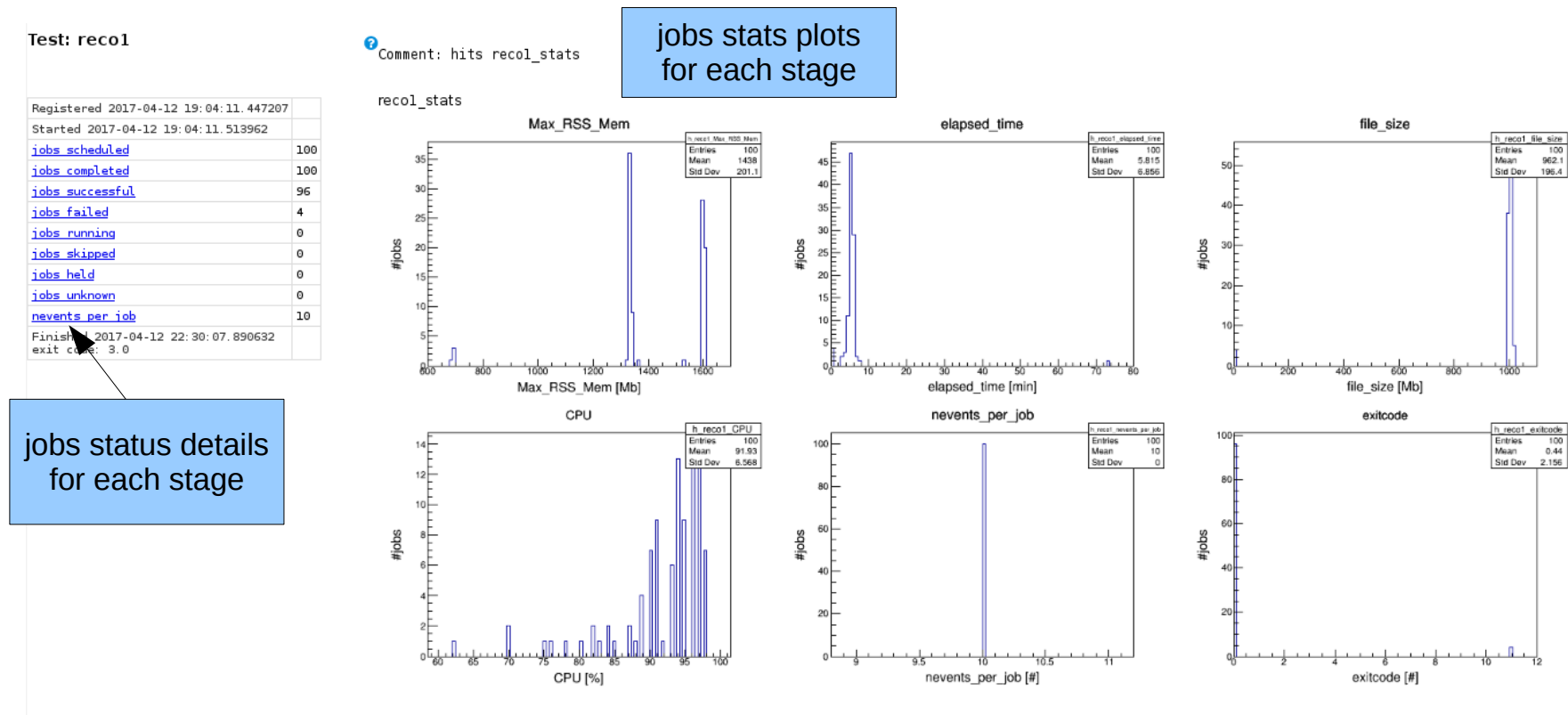


- CI validation phase inherits the CI functionality to upload and show plots
- This provides the user an easy access to CI validation results
- If something looks suspicious the user have access to the job outputs for a further analysis

Obtaining results: CI validation view



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



Obtaining results: email report

- Use different mailing lists according to the CI build status.
- Possible CI build status are:
 - **successful**: any issue reported
 - **warning**: the code run properly, but some check on the output is not successful
 - **failed**: something is not working properly, it can be a failure in the build of the code, or running unit/integration tests
 - There is still the possibility to send a report whatever is the status of the CI build
- Each experiment can choose a set of mailing lists to receive email reports according to the CI build status
- The mailing list can also be a list of individual user email
- Users that trigger a manual CI build will receive an email report

Obtaining results: email report

- Provides quick information in the subject;
- details about the CI build;



CI build of defaultwf workflow for slf6 Succeeded

Experiment: LArSoft uBooNE DUNE LArIAT ArgoNeuT

Code revision: develop

Build type: e10:prof

Platform: slf6

Workflow: defaultwf

Personality: mrb

Jenkins project: lar_ci, build 58

Build slave: buildservice002.fnal.gov

Trigger: git push on develop branch

For more details about this CI build status and to access the logs, please, see [the web monitoring application](#).

To access more logs see the [Jenkins project](#) web page.

Test Suite: quick_test_uboonecode quick_test_dunetpc quick_test_lariatsoft quick_test_argoneutcode

Start 1: ci_gen_regression_test_uboonecode

Start 2: ci_g4_regression_test_uboonecode

Start 3: ci_detsim_regression_test_uboonecode

Start 4: ci_reco1_regression_test_uboonecode

Start 5: ci_reco2_regression_test_uboonecode

Start 6: ci_mergeana_regression_test_uboonecode

Start 7: ci_gen_regression_test_dune35t

Start 8: ci_g4_regression_test_dune35t

Start 9: ci_detsim_regression_test_dune35t

Start 10: ci_reco_regression_test_dune35t

Start 11: ci_mergeana_regression_test_dune35t

Start 12: ci_gen_regression_test_dunefd

Start 13: ci_g4_regression_test_dunefd

Obtaining results: email report

- Support “warning” status for CI tests;
- details about CI tests that are not successful

Warning means:
experiment code runs successful,
but some check on the output
is not successful

CI build of defaultwf workflow for slf6 Warning at phase ci_tests

■ Failures ■ Warnings

CI Test Status	Error Message	Required Action
■ ci_reco2D_RUN1_regression_test_lariatsoft	Differences in products names	Request new reference files
■ ci_reco2D_RUN2_regression_test_lariatsoft	Differences in products names	Request new reference files

For more details about the warning phase, check the [ci_tests phase logs](#)

Experiment: LArSoft uBooNE DUNE LArIAT ArgoNeuT

Code revision: develop

Build type: e10:prof

Platform: slf6

Workflow: defaultwf

Personality: mrb

Jenkins project: lar_ci, build 54

Build slave: buildservice002.fnal.gov

Trigger: git push on develop branch

For more details about this CI build status and to access the logs, please, see [the web monitoring application](#).

To access more logs see the [jenkins project](#) web page.

Test Suite: quick_test_uboonecode quick_test_dunetpc quick_test_lariatsoft quick_test_argoneutcode
Start 1: ci_gen_regression_test_uboonecode

If some phase of the CI build
is not successful
there is a link that points
to the logs in the CI Web App

Upcoming new features

- **CI workflow reorganization**

- each experiment code will be tested independently each other, but still depend on the build/test of LArSoft
- CI build will test only code from experiments that can be potentially affected by a given commit
- CI email report will provide independent info for each experiment
- Support to get email report for individual CI tests

- **CI Web App reorganization**

- LArSoft and experiment code test information are shown in dedicated tabs

- **Add support for memory profiling to intercept memory leak**

Getting help

- The redmine wiki documentation is available at:
CI: <http://cdcv.s.fnal.gov/redmine/projects/ci/wiki>
LArCI: http://cdcv.s.fnal.gov/redmine/projects/lar_ci/wiki
- The CI web application provides tooltips and online documentation
- To request new features open a SNOW ticket
[Scientific Computing Services > Scientific Production Processing > Continuous Integration](#)
- For support/question send an email to: ci_team@fnal.gov
- As last resort, talk to LArSoft Team

**Thank you
for your attention**