SAND2022-6464C

Sandia National Laboratories

Exceptional service in the national interest

Multi-repository changeset testing with GitLab and Jenkins

Matthew Mosby & SIERRA/DevOps team

ASC S3C, ALBUQUERQUE, NM & VIRTUAL

May 24-26, 2022

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Acknowledgements: The SIERRA/DevOps Team

Sam Browne

Mark Hamilton

Jake Healy

Mario LoPrinzi

Matt Mosby

Tony Nguyen

Jon Sykora

Build → Test → Integrate: Only three steps – easy right?

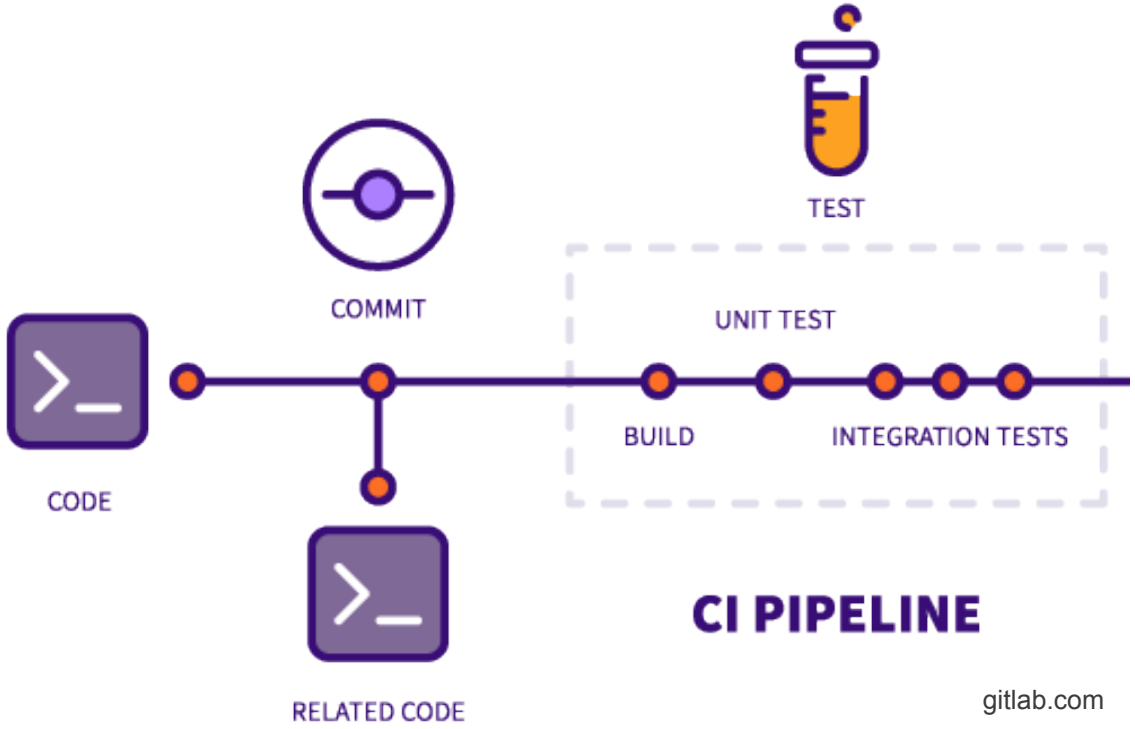
Many Continuous Integration/Continuous Delivery solutions exist

ASC is all-in on GitLab repository hosting and is expanding CI/CD capabilities

- GitLab/Jakamar runners
- Extensive containerization efforts

Many out-of-the-box solutions tailored for single repository workflows

CI/CD workflows coordinating changes across multiple repos requires custom infrastructure



SIERRA architecture & dev process poses a challenge

Multiple repositories protect need to know, but treated as a monolith

- Tight dependencies require coordinated testing across repositories
- Changes in repo A may cause failures in products contained in repo B

Large repo sizes require maintenance of on-disk assets

~50 active developers → ~20 integrations/day

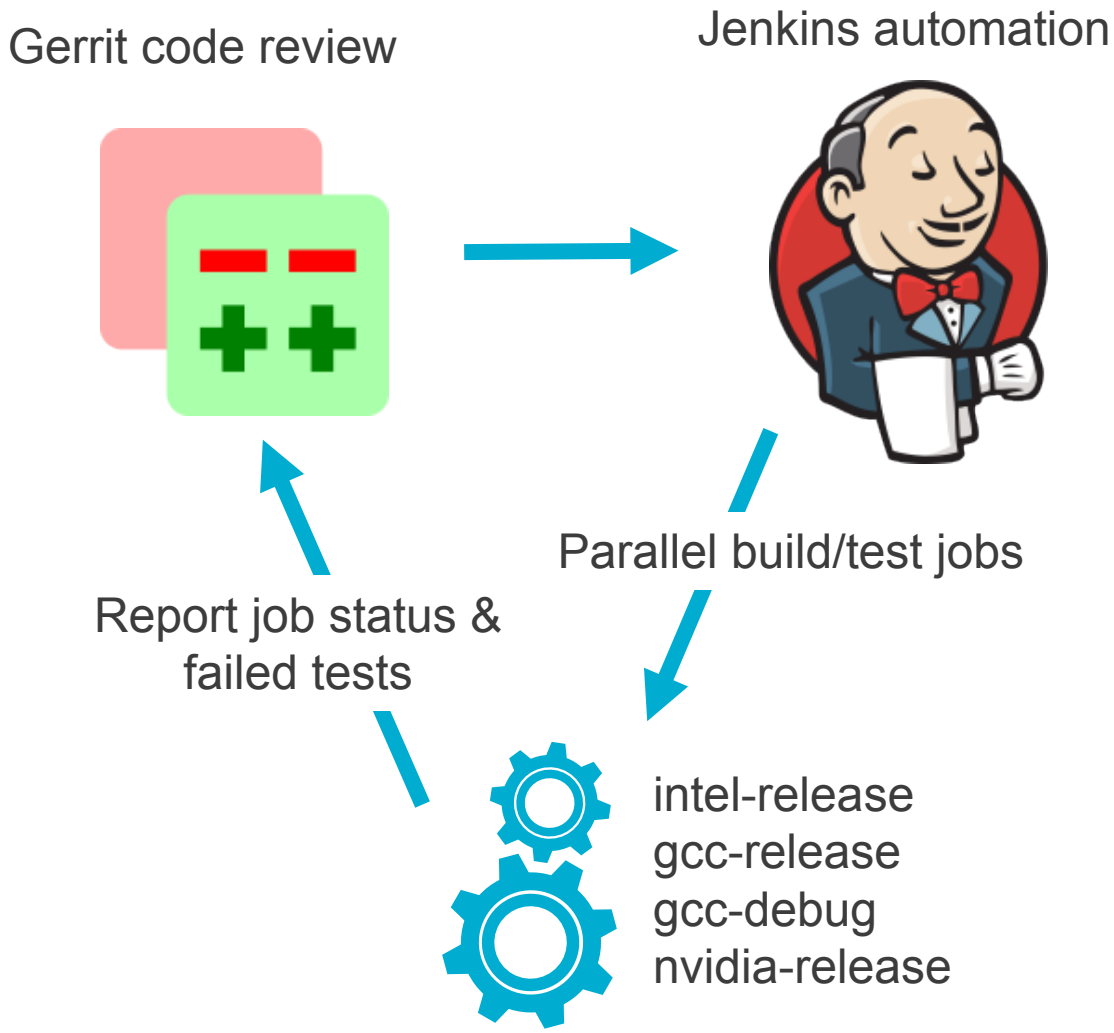
Past attempts at “CI” in SIERRA

Hourly builds with automated emails to authors of new changes with failed tests

- Slow turnaround of results
- Ambiguous ownership led to widespread filtering (deletion) of automated emails

Grassroots development of changeset testing using Gerrit code review framework

- Only supported testing in base repo
- Ambiguous *process* ownership/support
- Limited reporting capability – no artifacts



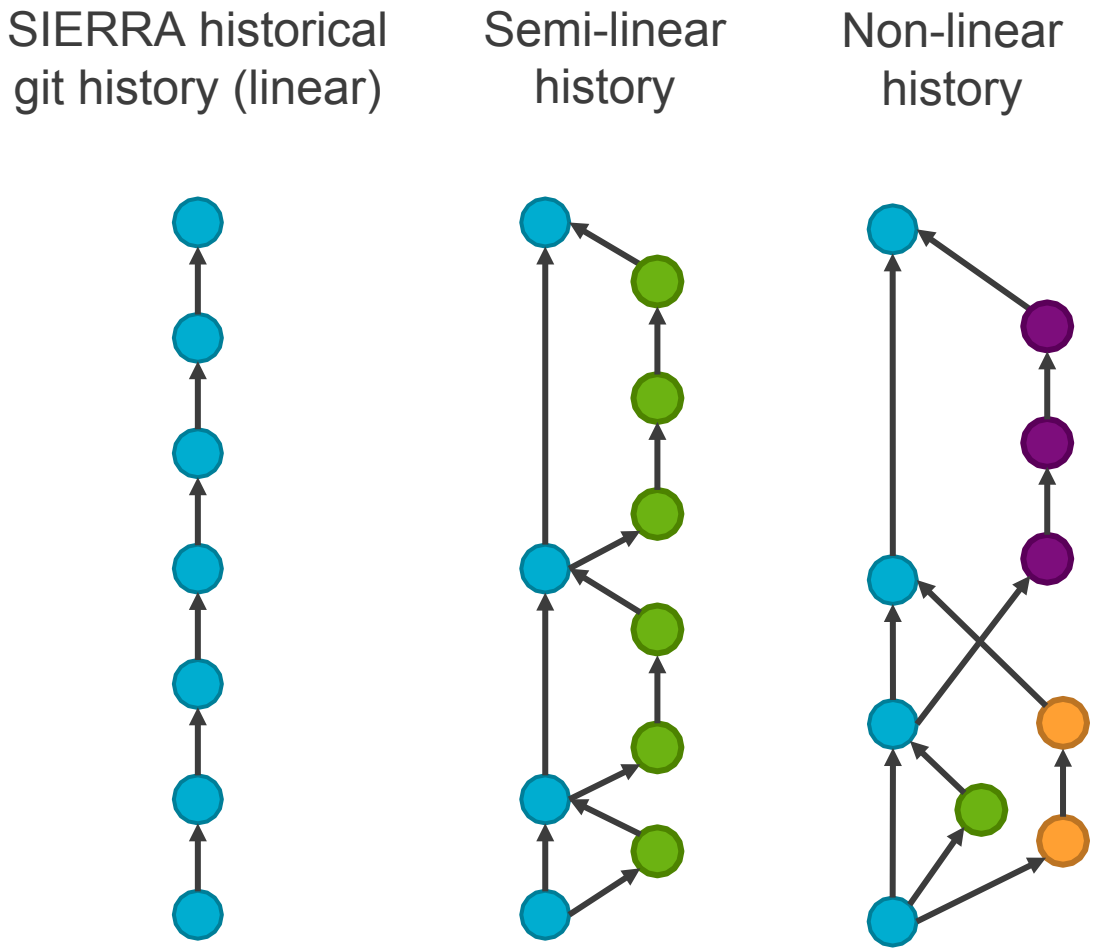
Prototype Gerrit “CI” proved value to SIERRA and need for more robust & supported capability

Migration to GitLab – Don’t discount workflow inertia!

SIERRA moved to GitLab from self-hosted git server during FY22

- Retirement of machine hosting SIERRA repositories
- One less thing to manage on the team
- Opportunity to integrate git hosting with code review and CI workflows

Simultaneously moved to a Merge Request workflow requiring branching

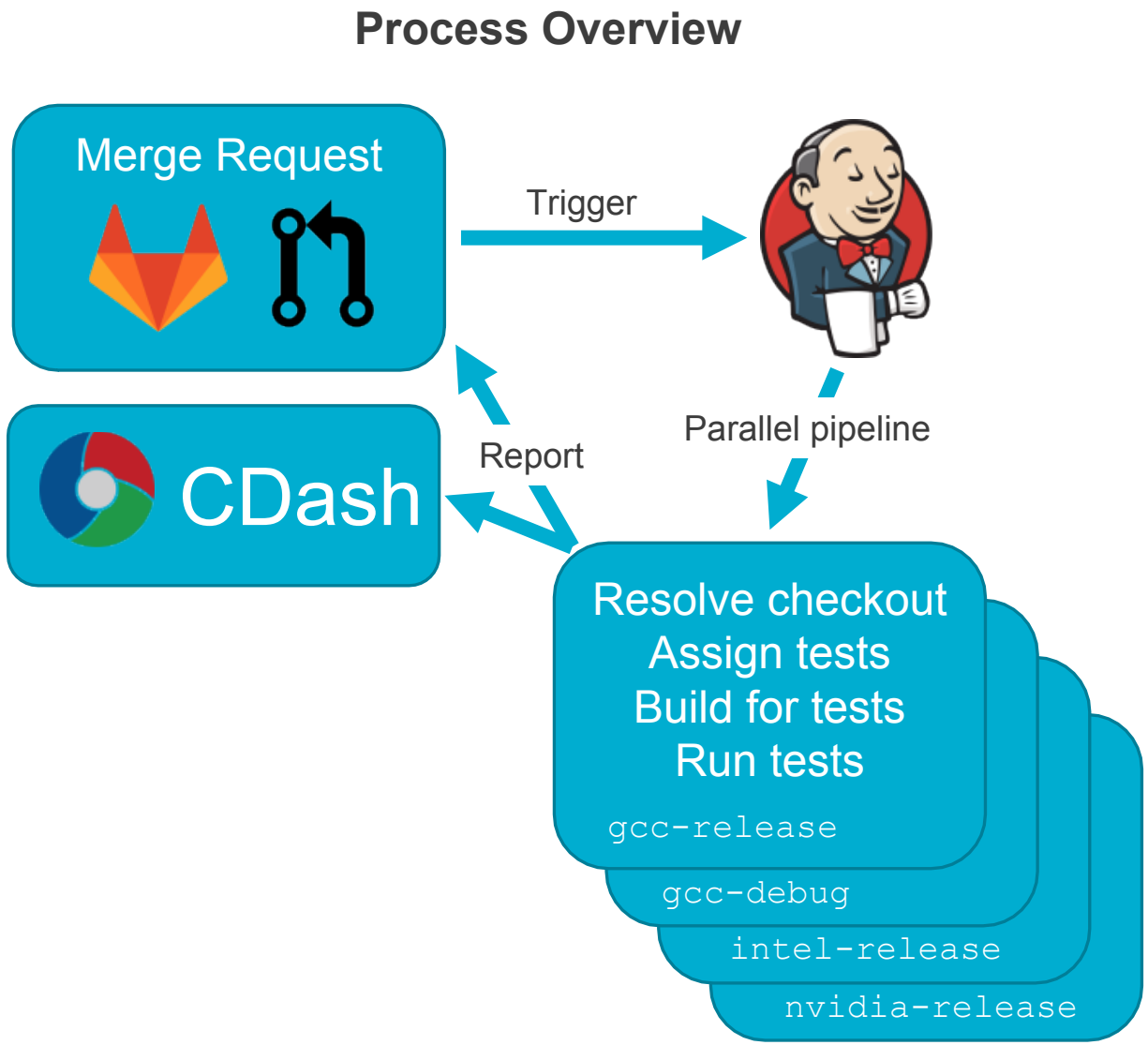


Reduced risk to productivity by first migrating repositories with less activity to build familiarity
(not covered by Gerrit CI anyhow – no gap in testing)

Branch-based multi-repo testing with GitLab & Jenkins

Goals of the system:

- Test associated changes from all repositories
- Minimize introduction of defects
- Min turnaround time / Max throughput
- Report build/test results consistently with nightly processes



The CI Testing Process

Checkout Resolution

Test Assignment

Build

Test

Checkout resolution

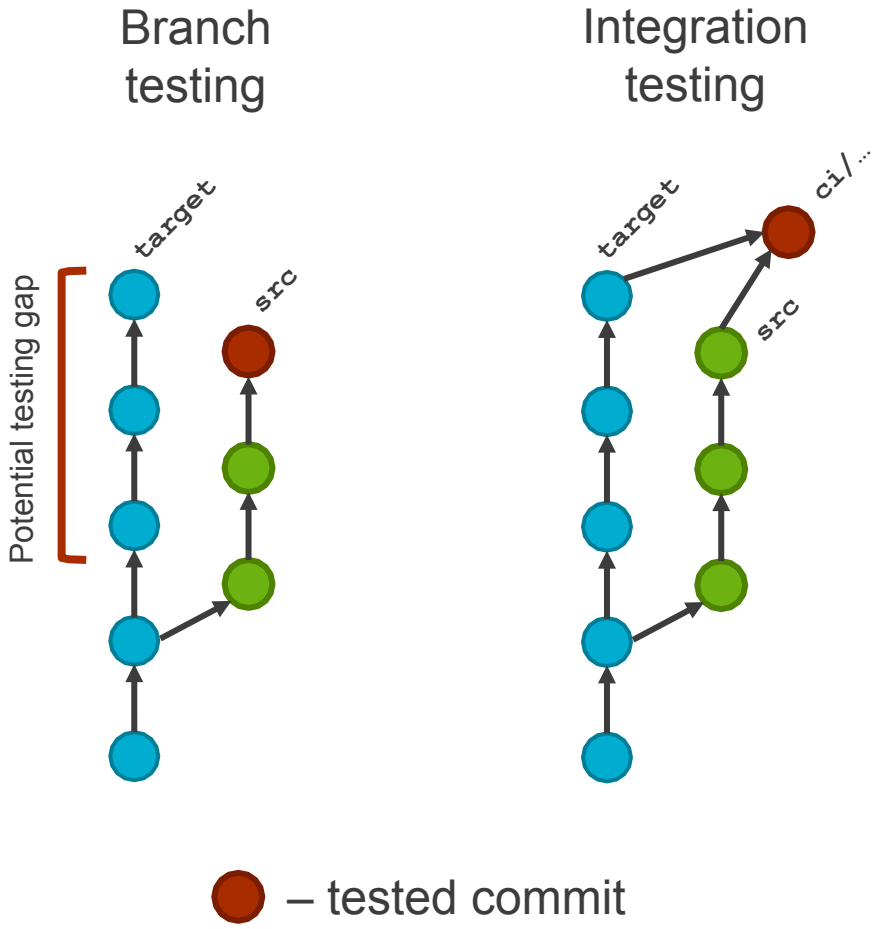
Checkout source branch with fallback to target branch in each repository

Opened gap in testing where changes integrated to target branch not considered

- “Spurious” failures degraded trust in process
- Exacerbated by multi-repo workflow

Now use intermediate merge commit on temporary CI branch

Repositories with no source branch still create a temporary CI branch



Temporary CI branch is pushed to repositories for developer reproducibility

CI build/test performance

SIERRA is **huge**

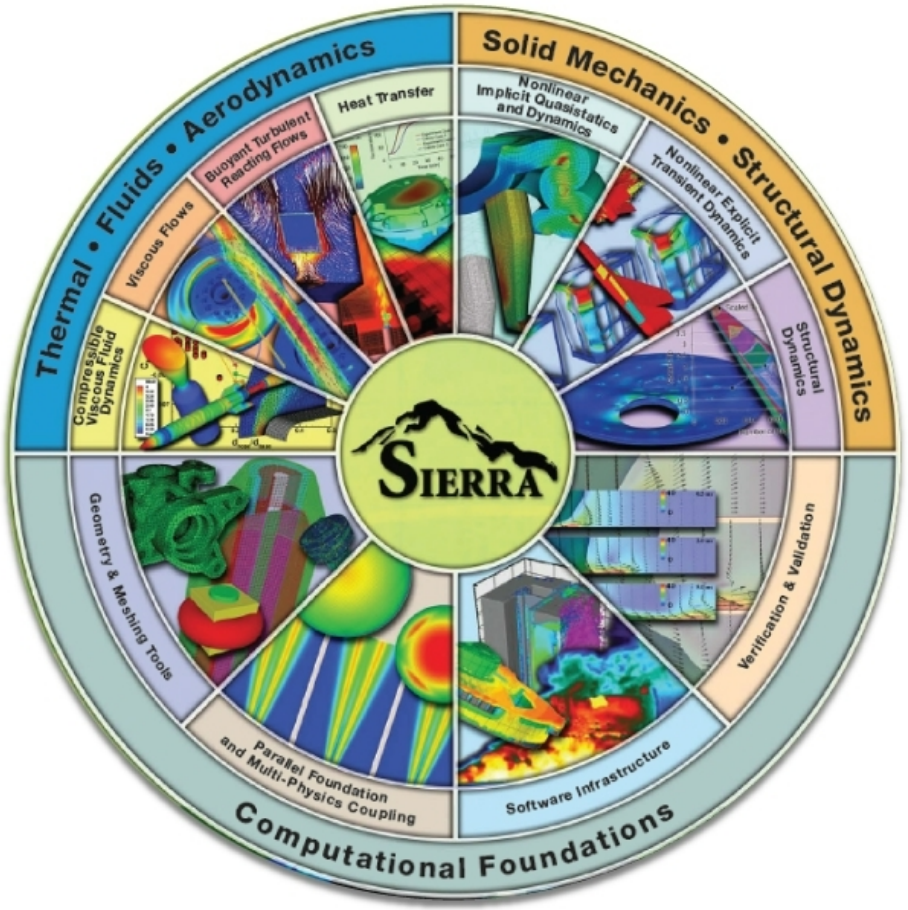
- ~700 developer build executable targets
- ~20k regression tests

What tests to assign for isolated changes?

What executables need to be built to run tests?

How do we build those executables quickly?

How do we efficiently turn around testing?



BRUCE – Assign tests based on what code changed

BRUCE is a backronym for Basic Reverse Unittes Coverage Evaluator

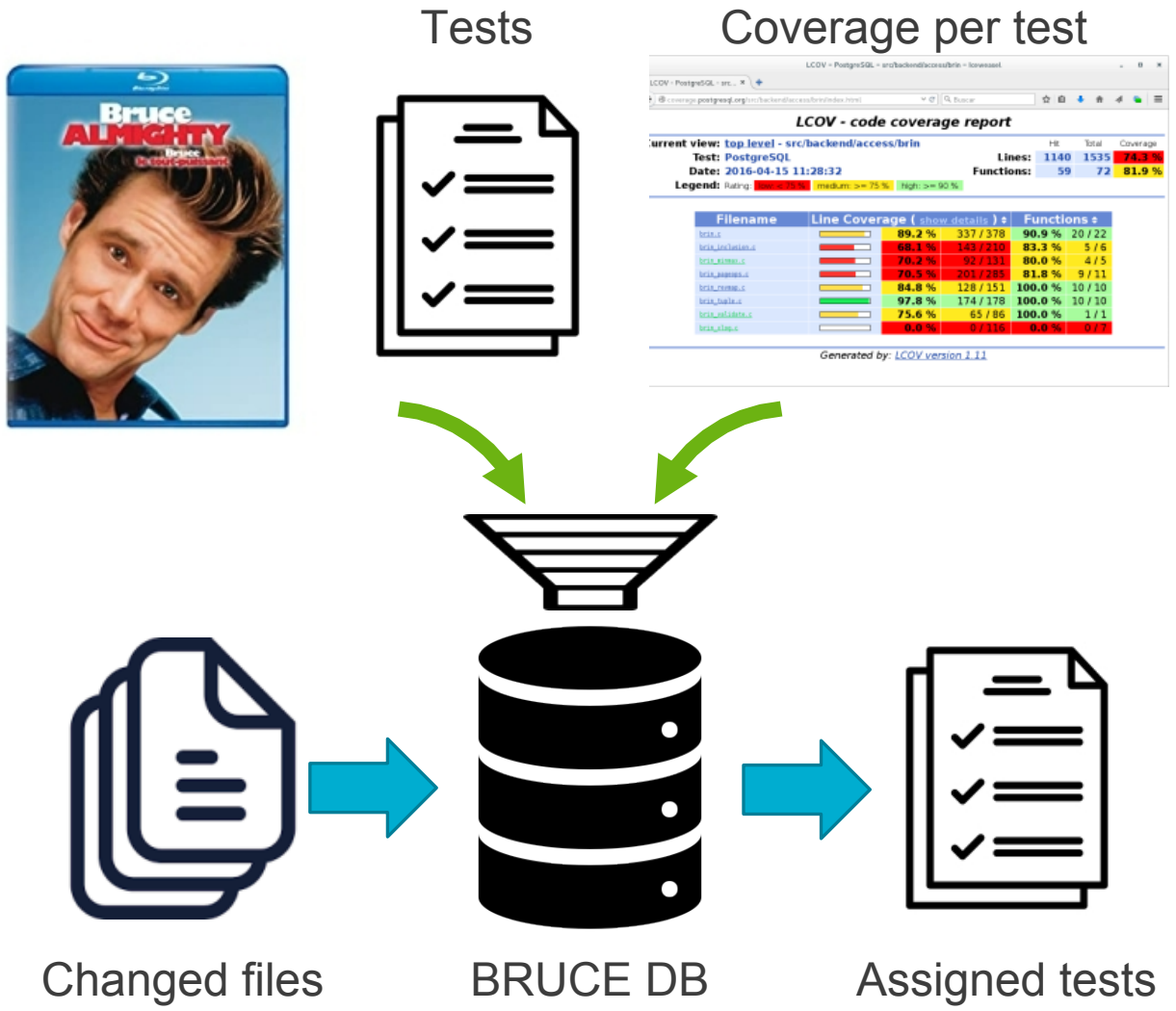
Run full test suite with code coverage enabled

Build relational database mapping files to tests that cover them

Use ‘git whatchanged’ to query database for which tests to assign

Also assign any modified tests

Modifying a unit tester now only assigns the unit test rather than ~20k tests



BRUCE + changed tests assignment results in minimum tests to run and targets to build

Build performance – cache is king!

Full build of SIERRA from scratch takes ~3-4 hrs (gcc, using 36 cores)

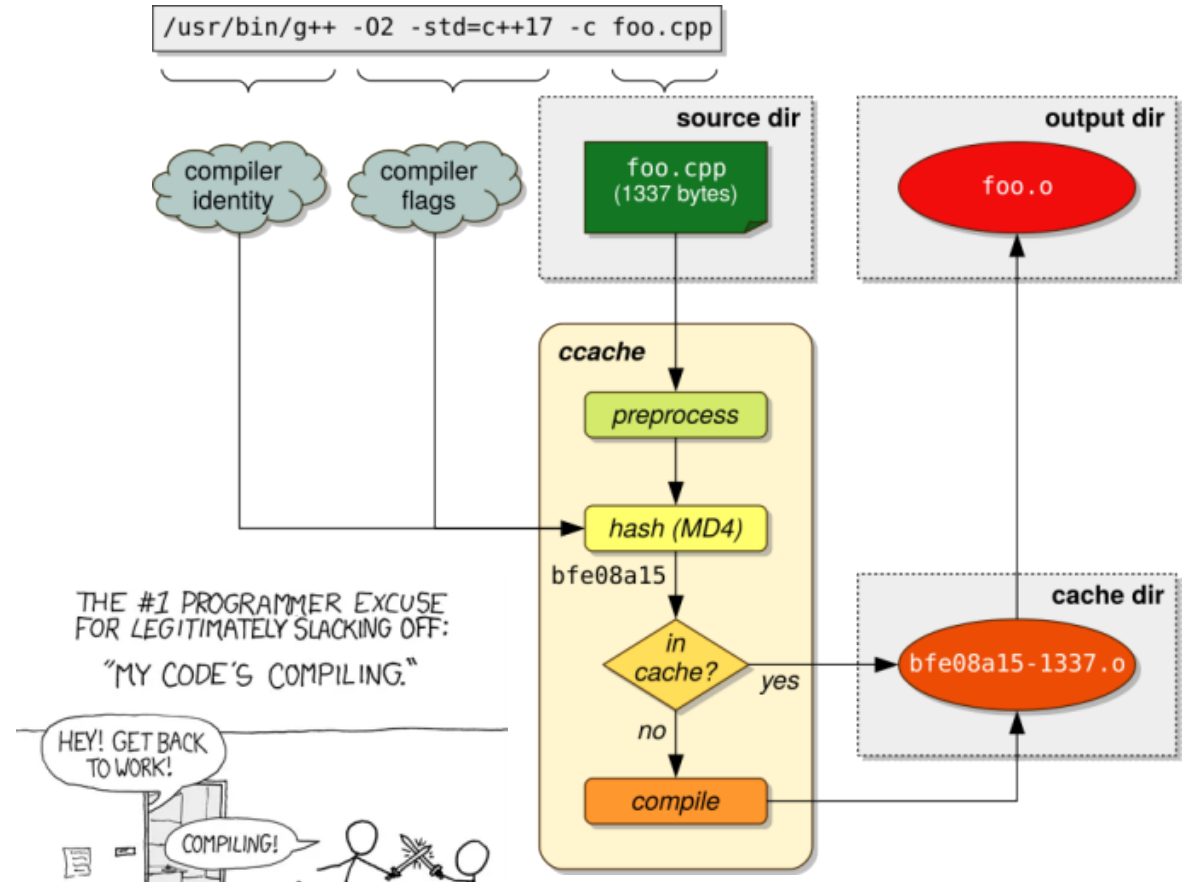
Only build targets used in assigned tests

Have a bespoke jam-based build system with sketchy incremental builds

- Fine for developers – not robust enough for automated processes

Performant builds via caching and parallelism

- Spack TPL builds cache & ccache for apps
- Large (36+ core) build machines



https://www.methodpark.de/blog/the-c-c-developers-guide-to-avoiding-office-swordfights-part-1-ccache/

https://xkcd.com/303/

With caching, builds typically less <1hr – also using ccache in CI → in high hit rate for devs

Test performance – scale baby scale!

Lots of SIERRA devs with beefy workstations

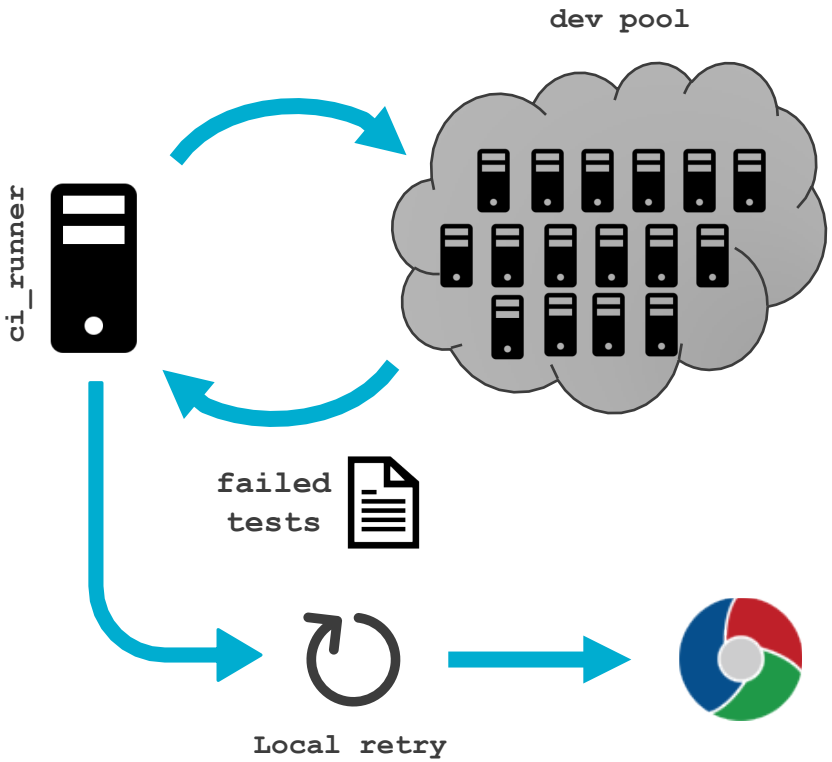
Dev opt-in to create a Beowulf cluster/pool

Custom test harness dispatches tests to cluster

- Short per-test time limit (5 min)
- 90 min overall time limit

Re-run failed/remaining tests locally

- Resolves intermittent network issues
- 20 min/test time limit to catch timeouts



Maximize throughput in pool, local re-try for robustness

Developer productivity – is the failure my fault?

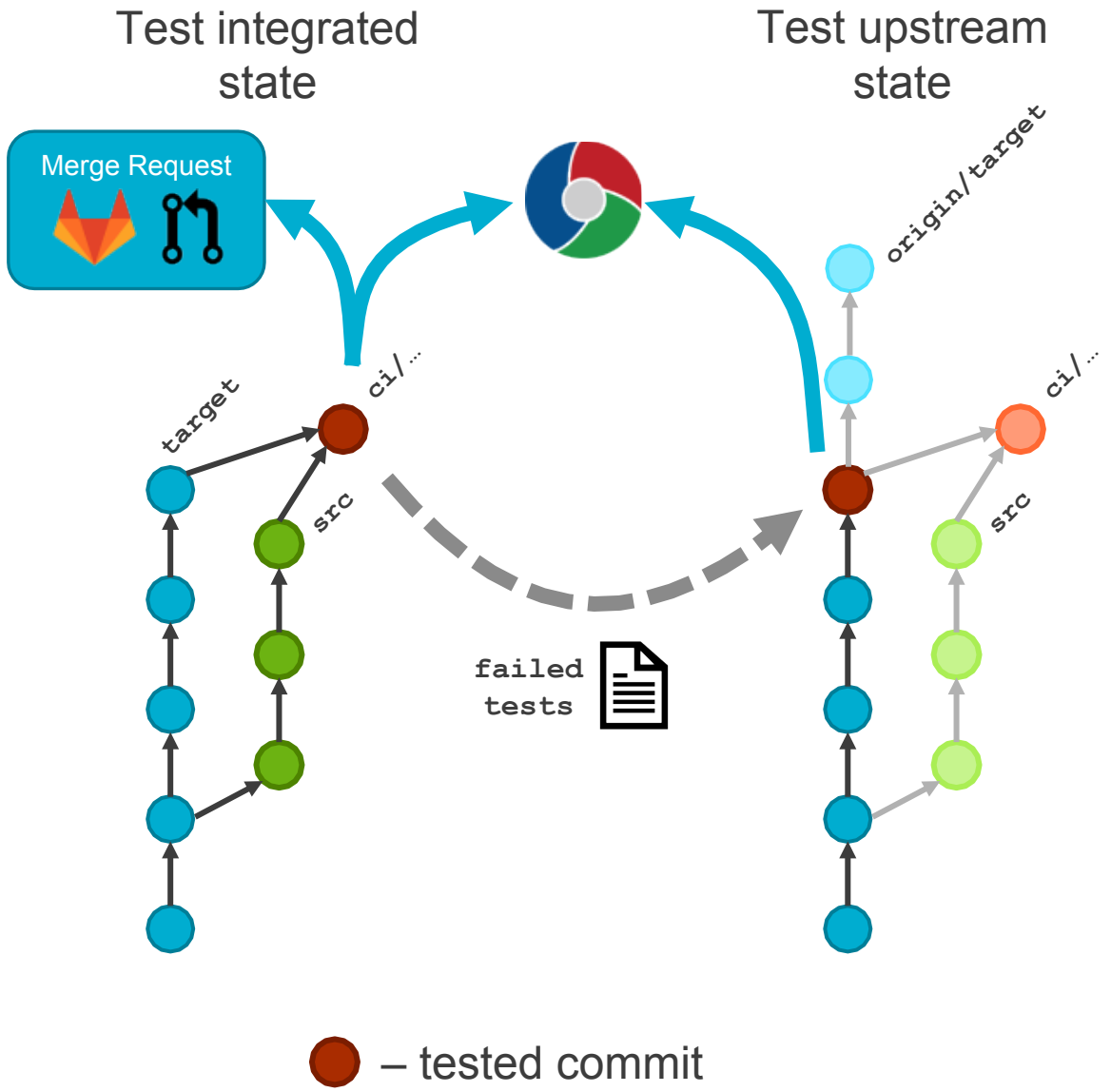
Prototype Gerrit process identified whether failures were pre-existing and reported **success** if no ***new*** failures were added

Large philosophical disagreement on what the new system should do

Agreed that the system should reduce burden of reproducing failures

Only succeed if the integrated state is clean!

> "The only way to go fast is to go clean"
>
> -- Robert "Uncle Bob" Martin



Reporting of CI Results

Feedback to GitLab Merge Request

CDash reporting

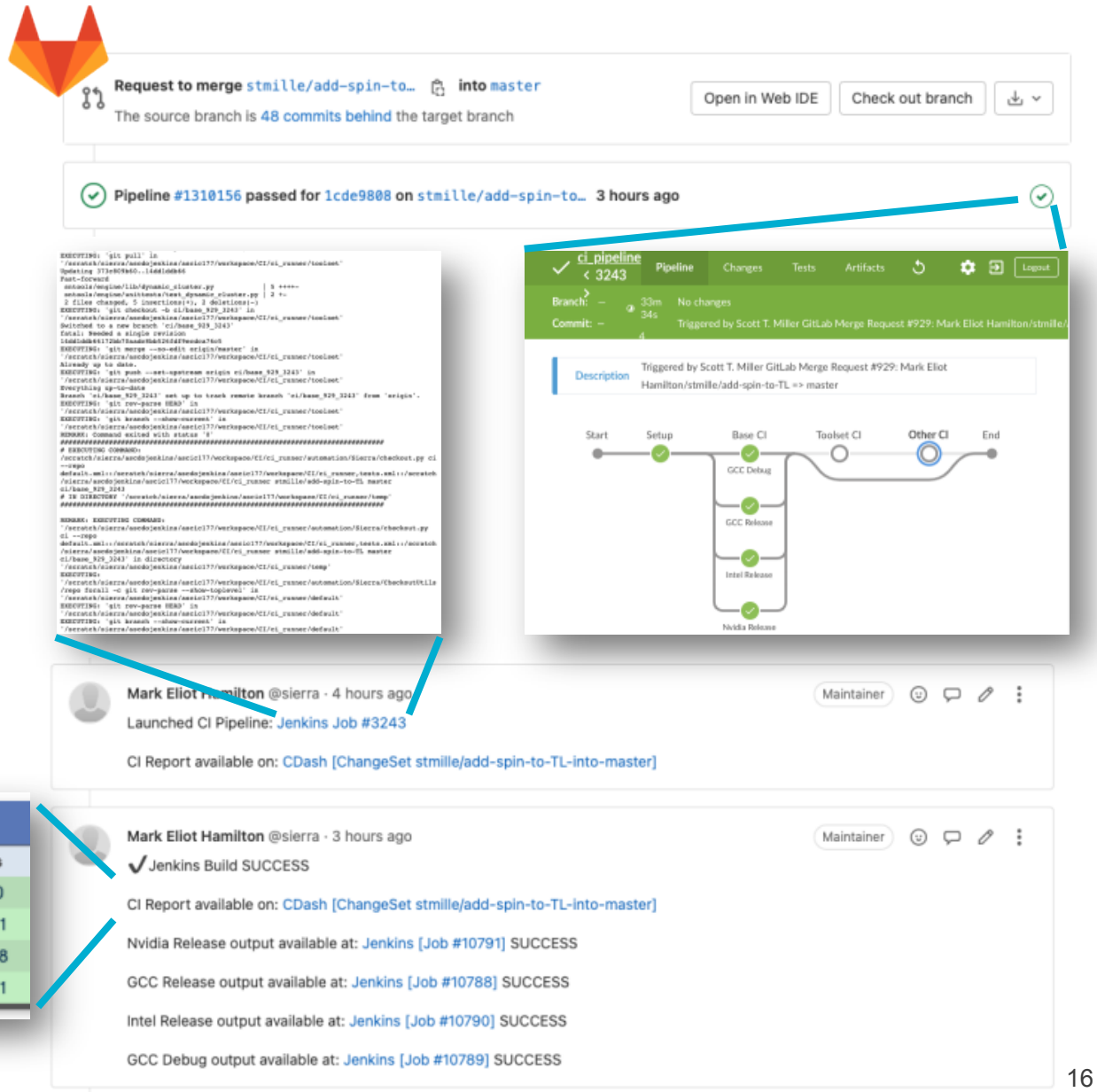
Feedback to GitLab MergeRequest

Comment-based feedback to MR when pipeline launched

- GitLab integrated pipeline link
- Links to pipeline / “process vomit” logging
- Links to dashboard

Moving in direction of reduced output to focus on code review in MergeRequest

- Workflow inertia/familiarity from Gerrit



		Update	Build		Test		
Site	Build Name	Files	Error	Warn	Not Run	Fail	Pass
ascicgpu059	stmille/add-spin-to-TL-into-master-nvidia-release	9	0	25	0	0	20
ascic177	stmille/add-spin-to-TL-into-master-intel-release	9	0	1	0	0	841
ascic163	stmille/add-spin-to-TL-into-master-gcc-debug	9	0	0	0	0	608
ascic156	stmille/add-spin-to-TL-into-master-gcc-release	9	0	0	0	0	841

CDash reporting

Same as in the nightly cross-platform testing

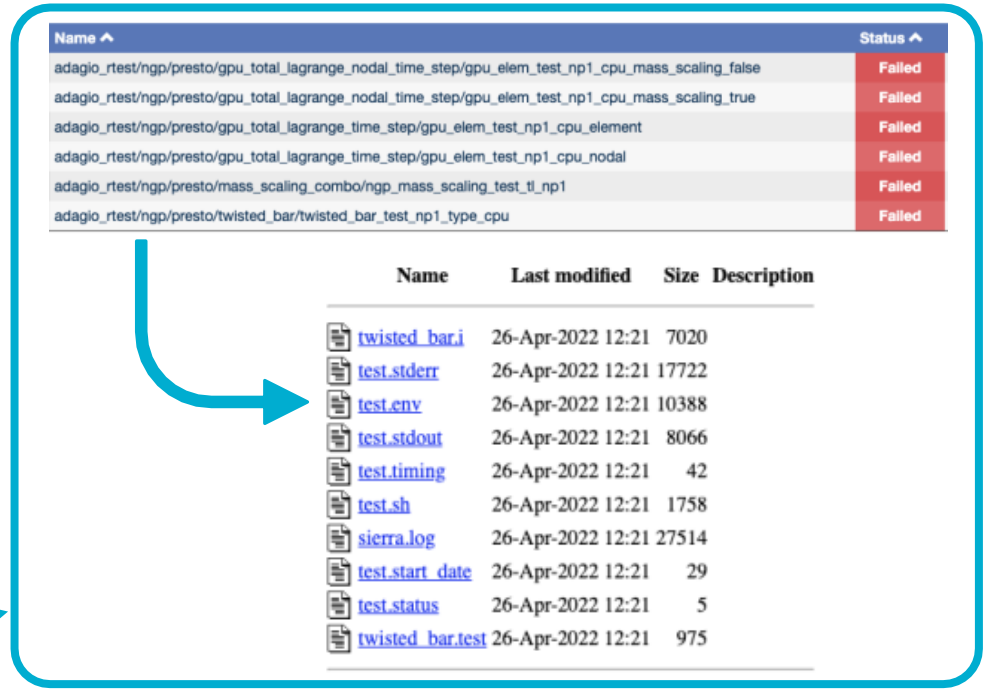
- Presentation familiar to dev community

Test artifacts archived for failing tests

Links in GitLab MR resolve to timestamp-filtered results on overall CI dashboard project

`checkout manifest`

`failed tests artifacts`

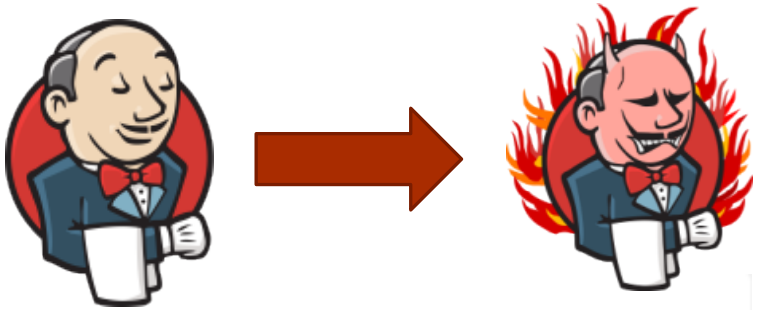


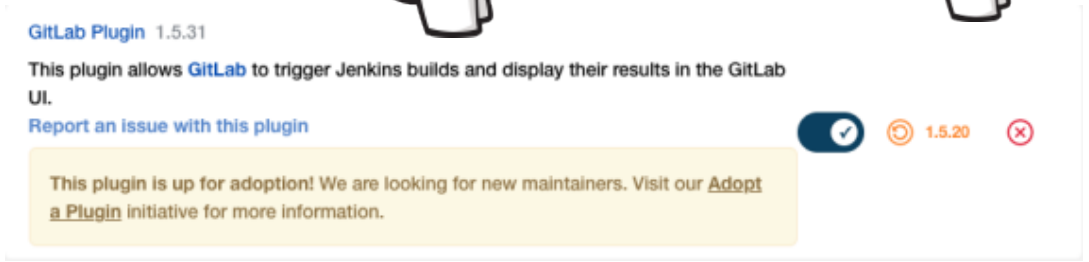
		Update	Test	
Site	Build Name	Files	Fail	Pass
ascic151	stmille/TL-retain-local-fields-into-master-gcc-debug-target-branch-state	9	0	287
ascicgpu061	stmille/TL-retain-local-fields-into-master-nvidia-release-target-branch-state	9	0	6
ascic154	stmille/TL-retain-local-fields-into-master-gcc-release-target-branch-state	9	0	89
ascic0198	stmille/TL-retain-local-fields-into-master-intel-release-target-branch-state	9	0	81
ascic154	stmille/TL-retain-local-fields-into-master-gcc-release	9	89	746
ascicgpu061	stmille/TL-retain-local-fields-into-master-nvidia-release	9	6	14
ascic0198	stmille/TL-retain-local-fields-into-master-intel-release	9	81	754
ascic151	stmille/TL-retain-local-fields-into-master-gcc-debug	9	287	315

Connecting the Dots: GitLab/Jenkins Integration

Challenges with community support
Webhook integration

Challenges with community support





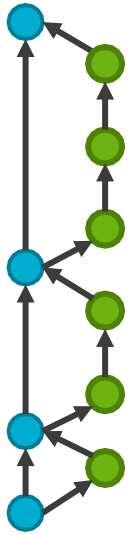
Jenkins uses a plugin architecture with community development of feature plugins

What do you do when a critical feature is no longer supported?

Desired workflow:

- Use semi-linear merge strategy
- Auto-build on MR creation only
- Manual re-build for subsequent changes

Not possible via plugin due to bugs



Broken plugin forced rebuild whenever MR updated

GitLab on record as not wanting to support semi-linear CI workflows

Limited or non-existent support of tools forced less-desirable workflows

Webhook integration

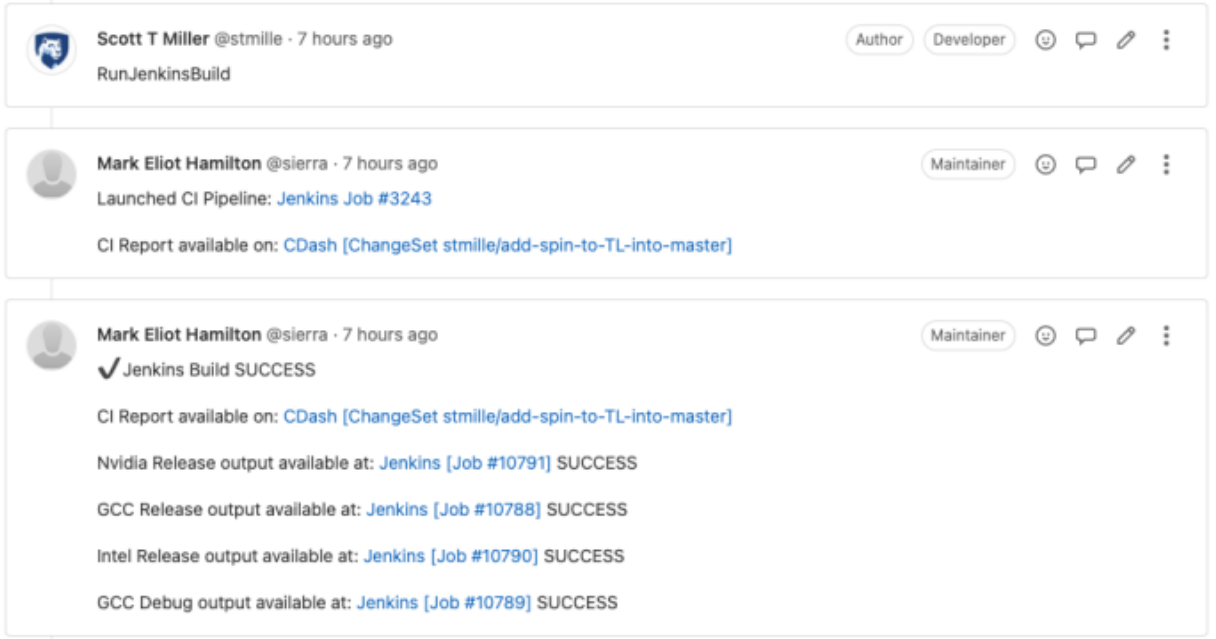
Challenges with GitLab/Jenkins plugins drove manual webhook integration

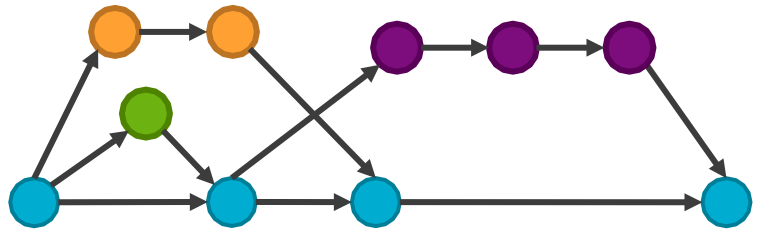
Forced to use manual comment-based trigger to achieve desired testing behavior

Moving community towards non-linear merge strategy for better automation support

Building community familiarity and trust

- Identifying issues, e.g., merge conflicts
- Trusting the results rather than re-triggering





Slowly moving towards fully automated CI testing given system constraints

Conclusions and future work

CI process has been in production since ~March 2022

- GitLab analytics show ~20 merges/per day with mean time to merge of 2 days
- CI process performs hundreds of builds per day (up to 339 on 5/2, ~85 changes) with an average turnaround time of ~2 hours (depending on # tests)
- Successfully maintained/improved low rate of defect introduction

Moving to the non-linear merge strategy will be better supported by automation

- All changes logically should trigger testing

Test suite reduction for faster turnaround with similar defect rate

Investigate using GitLab/Jacamar runners

Questions?

mdmosby@sandia.gov