May 18th, 2026

Ethan Lang

Modernization: Continuous Integration

Battelle Energy Alliance manages INL for the
U.S. Department of Energy's Office of Nuclear Energy

Idaho National Laboratory

Outline

- Correctness:
  - Requirements,
  - Traceability,
  - Test Types
- What is Continuous Integration (CI)?
- Historical Integration Method
- Modern CI Method
- Questions

Correctness

- What does ”correct” mean for RELAP5-3D?
  - Physics are captured by the math.
  - Algorithms implemented as intended.
  - Ultimately: the numbers are good.
- How do we know that a number is “good”?
  - Robust validation against data.
  - Historical precedence.
  - Ultimately: results that are defensible to regulators.

- RELAP5-3D has been tried and tested for decades.
- We are confident in its results.
- To support our users, we must continue to improve in this area.

Requirements

- Historically, RELAP5-3D has not maintained a formal, consistently tracked requirements specification.
- Matching validation data was broadly accepted as sufficient proof of correctness by the development team, industry, and regulators alike.

- RELAP5-3D modernization efforts now include developing formal requirements specifications.
- Formal requirements will safeguard the code's validated core while providing a structured framework for future improvements.

Requirements

- **Some modern requirement specification examples:**

1. *”Physically invalid input decks will be detected, and a descriptive error message will be printed before terminating.”*
2. *"The two-fluid model shall demonstrate global mass conservation across all supported transient scenarios.”*
3. *"RELAP5-3D shall reproduce all historically validated results within accepted tolerances upon each new code release."*

Traceability

- Every test shall be mapped to a requirement, and every requirement shall be covered by a test.
- Ensures all requirements are tested and no tests are without a purpose.
- Recordkeeping across code versions.
- RELAP5-3D's traceability system:
  - Tests are tagged with one or more requirements.
  - An autogenerated traceability matrix maps test status to their requirement, showing immediate coverage.
- For legacy code: fix or identify behavior, formalize requirement, write test
- For new developments: formalize requirement, write test, implement code

Test Types

- All test types are regression tests protecting against unintended changes

- Unit Tests:
  - Target narrow, specific behavior in isolation
  - Assert what a value must equal
  - Used on individual functions/subroutines, models
  - Lacking in RELAP5-3D; an active area of growth
  - New developments and legacy changes require unit tests as part of the deliverable

Test Types

- Verification Tests:
  - Compare code output against gold values within a tolerance to a high degree of accuracy.
  - Any deviation indicates either
    - the underlying algorithm has changed, or
    - the environment has changed.
- Validation Tests:
  - Compare code output against experimental data with a tolerance using L2 norms
  - L2 norm measures the overall magnitude difference between two datasets.
  - Data is noisy, so visual inspection of plots can also be used.

Test Types

- Looking ahead:
  - More unit tests.
  - More validation cases, especially for more working fluids.
  - Track down uncertainties for legacy data and include uncertainty quantification in validation cases; formalize acceptance more.
  - Explore the use of deterministic simulation testing to systematically find edge cases and stress the code base.
    - Mock inputs for a given subroutine are generated using a random seed.
    - Run many cases and collect seeds that result in notable outcomes.
    - Investigate the subset.
    - Primary value would be reducing unanticipated segmentation faults.

What is Continuous Integration (CI)?

- Developers continually merge code changes into a central repository.
- Executables are automatically built on all target platforms with versioned, consistent, and reproduceable environments.
- Test suites are run with each executable.
- Test documentation (traceability matrix, VnV pass/fail, etc.) is automatically generated.

- Challenges CI addresses:
  - Manually building, testing, and documenting on each platform was tedious, time consuming, and inconsistent.
  - Environments were not guaranteed to be standardized.
  - Slow feedback from other developers because changes were siloed.

Historical Integration Method

- The Process:
  - Developers worked on local copies of the code base.
  - When ready, changes were submitted to the Code Custodian (CC).
  - The CC manually resolved conflicts between simultaneous submissions, working with the submitters.
  - Building, testing, and documenting results were manually done for each platform by the CC.
  - Integrated changes were committed to a central SVN repository.
- Issues:
  - Manual conflict resolution was slow and error-prone.
  - Long periods without submission made merges complex.
  - Testing was not automated, and occasionally breaking errors were merged into the internal version of the code.
  - Finding such bugs was very challenging.

Modern CI Method

- The Process:
  - Developers submit changes via pull request (PR) to a ‘devel’ branch using a structured PR template that requires:
    - A summary and detailed description of changes,
    - Links to a tracked issue and requirements,
    - A checklist confirming coding standards, documentation, traceability requirements and associated tests, and performance review have been completed.
  - Every PR automatically triggers ~600 short tests across all build configurations
  - A documentation step creates the versioned documentation.
  - All tests and checks must pass before merge is permitted.
  - At least one member of the Change Control Board must review and approve.

Modern CI Method

- The Process, continued:
  - Each week, the ’devel’ branch is automatically promoted to ‘weekly’, where the full test suite (1000+ tests) runs across all builds.
  - If all tests pass, ‘weekly’ is automatically promoted to ‘main’.
  - New releases of the code are created from ‘main’.
- Advantages:
  - Integration is continuous, changes are frequent and smaller.
  - No single bottleneck (Code Custodian).
  - Bugs are caught immediately.
  - Every change is traceable, reviewed, and tested across all builds.
  - PR template enforces requirements and testing as a matter of course, not an afterthought.

Modern CI Method

- Automatic Code Releases:
  - The new CI pipeline automates and frontloads the building, testing, and documenting of new code releases.
  - Code Custodian simply runs a single script to create tar/zip packages for each build configuration with all associated documentation for full traceability.

- What if tests fail?
  - Manual review of failures must occur.
  - Sometimes failures are expected (new physics model, algorithm bug fix, etc.) in which case a justification is written and reviewed. Then a change of gold values (regolding) can be done.

Summary

- RELAP5-3D has earned its reputation through decades of validated results. The modernization effort ensures that reputation is backed by a formal, auditable development process.
- Formal requirements, automated testing across all platforms, and enforced traceability mean that every change to the code is justified, tested, and documented.
- The CI pipeline automates these best practices so that we can consistently demonstrate that the code base is demonstrably correct without ambiguity.

Discussion and Questions

Idaho National Laboratory

Battelle Energy Alliance manages INL for the U.S. Department of Energy's Office of Nuclear Energy. INL is the nation's center for nuclear energy research and development, and also performs research in each of DOE's strategic goal areas: energy, national security, science and the environment.