Continuous Software Quality in HPC

The FALL3D Experience in ChEESE-2P

IBERGRID 2026 | Quality of software, services and data

Heribert Pascual

GEO3BCN-CSIC | ChEESE-2P



GEO3BCN
Geosciences Barcelona - CSIC

ChEESE

PROJECT CONTEXT

ChEESE-2P and FALL3D: project context and application scope

ChEESE-2P

PROJECT FRAMEWORK

- 2023–2026 · GA No. 101093038
- 12 Exascale Computationale challenges
- Solid Earth applications and services
- Performance · portability · accelerators · CI CD

11 FLAGSHIP CODES

9 PILOT DEMONSTRATORS

15 SIMULATION CASES

FALL3D

APPLICATION IN FOCUS

- Volcanic tephra transport and deposition
- Research and operational forecasting
- Particles · aerosols · radionuclides
- Quality automation constrained by the Fortran tool ecosystem

20+ YEARS OF DEVELOPMENT

CPU + GPU MPI / OPENACC

FORTRAN HPC CODEBASE

This presentation examines how the ChEESE-2P quality objectives were extended and implemented in FALL3D

THE GAP

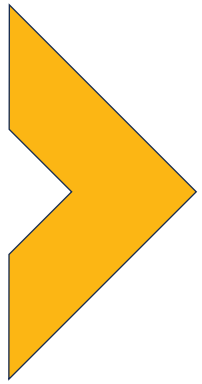
A successful CI job is not yet scientific validation

Software-level evidence

- Successful builds
- Unit tests pass
- Warnings + ASan
- Coverage evidence
- Reproducible checks

Scientific evidence

- Numerical validity
- CPU / GPU consistency
- Decomposition consistency
- Restart consistency
- Valid benchmark execution



Quality decision needs both

CODE EVIDENCE

+

SCIENTIFIC EVIDENCE

One green job is not enough

Scientific **HPC quality** requires both **software evidence and scientific evidence**

EVOLUTION

FALL3D turns minimum requirements into one evidence chain

Initial practice relied mainly on manual and visual comparison of simulation outputs.

ChEESE-2P targets

MINIMUM REQUIREMENTS

- Warning-free build
- >50% line coverage
- CI/CD adoption

FALL3D operational scope

INTEGRATED LEVELS

- Unit + component tests
- >70% coverage, warning-free builds and ASan
- Native HPC builds
- Scientific validation
- Correctness-gated benchmark

Quality evidence

TRACEABLE EVIDENCE PER CHANGE

- Distributed states
- Reports & artifacts
- Scientific comparisons
- Validated performance metrics

FALL3D evolved from **manual result checking to** continuous, **traceable quality evidence** for every software change

Each stage has one responsibility and one hand-off

1 - GitLab CI

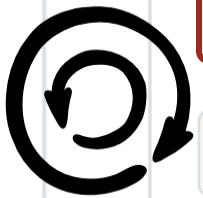
- Unit / component tests
- Coverage
- Trigger HPC gateway
- Strict builds
- ASan

Triggered after successful CI checks

2 - HPC gateway

- Validate HPC environment
- Submit Slurm workflow
- Record job identifiers
- Release CI runner

CI runners are free during HPC execution



3 - Slurm execution

- Native CPU / GPU builds and runs
- Scientific validation
- Correctness-gated benchmark
- Reference comparison + TALP metrics
- Report job statuses via GitLab API

Dependent Slurm jobs execute the workflow; the final job triggers result collection.

4 – Finalizer pipeline

- Collect job states
- Retrieve reports, logs and metrics
- Produce quality summary
- Store GitLab artifacts

Retrieves distributed evidence and stores it as GitLab artifacts

GitLab orchestrates, Slurm executes, and the finalizer turns distributed results into **traceable evidence**

SCIENTIFIC VALIDATION

Two cases, complementary evidence

Template case

ROUTINE VALIDATION

- 24-hour simulation
- Distributed with FALL3D
- Exercises the end-to-end workflow
- Stable regression scenario

Functional + numerical consistency

Scalability benchmark

REFERENCE + PERFORMANCE

- One-hour simulation
- High resolution, ~100M grid points
- Public reference files in Zenodo
- Instrumented with TALP

Correctness-gated performance evidence

VALIDATION DESIGN

TEMPLATE CASE — Six cross-configuration runs



SCALABILITY BENCHMARK — Single fixed configuration

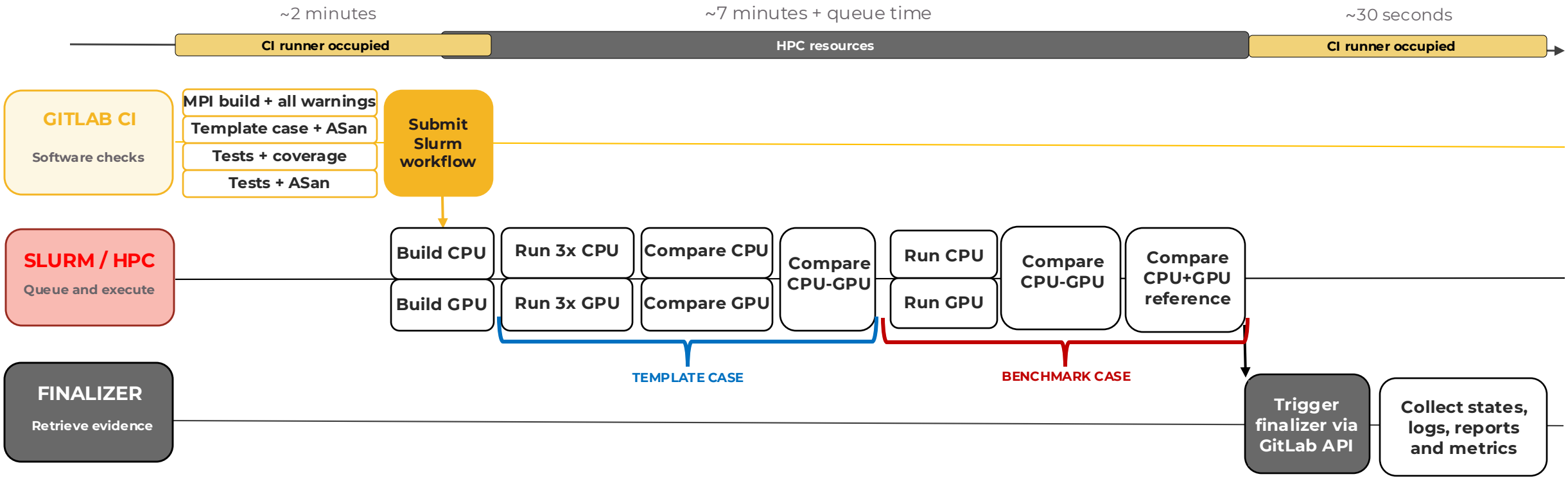
2 x 2 x 2 domain decomposition

Zenodo reference + TALP metrics

COMMON ACCEPTANCE — outputs · restart predefined tolerances

The **template validates workflow** and the **benchmark** adds reference-based **correctness and performance** evidence

The CI runner submits the work; Slurm owns queueing and execution



After submission, the runner is free: **Slurm owns queueing and execution**, while the finalizer returns the evidence

GENERALIZABLE LESSONS

Separate workflow mechanics and code-owned validation

WORKFLOW MECHANICS

- CI-to-HPC hand-off
- Asynchronous Slurm execution
- Selective CPU/GPU allocation
- Artifact and metric aggregation
- Traceability per software change

SCIENTIFIC VALIDATION

- Representative scenarios
- Scientific outputs
- Numerical tolerances
- Restart acceptance criteria
- Benchmark validity criteria

SEPARATION OF CONCERNS

Workflow mechanics

Code-owned validation

Evidence orchestration
Scientific acceptance

Workflow mechanics and scientific validation remain **separate responsibilities**

ROADMAP

Validation is operational; monitoring and delivery continue to evolve

Current capabilities

OPERATIONAL

- Layered CI + HPC execution
- Six-run template validation
- Fixed 2 x 2 x 2 benchmark
- Zenodo reference + TALP evidence
- Distributed finalization

Monitoring and regression

WORK IN PROGRESS

- Publish coverage evolution
- Track TALP metrics across versions
- Define performance regression criteria
- Extend scientific validation scenarios

Next project phase

PLANNED

- Energy as regression evidence
- SQAaaS assessment + badges
- Further delivery automation
- ChEESE-Factory WP4 input

OPEN CHALLENGE: Security restrictions and changing platform environments still **limit reusable delivery automation**

TAKEAWAYS

Quality evidence must follow the science and the machine

01

Use the right level

Keep reproducible software checks in CI. Run representative builds and validation on HPC

RIGHT CHECK · RIGHT ENVIRONMENT

02

Gate performance with correctness

Validate scientific outputs and restart files first. Interpret performance only after numerical validation passes

CORRECTNESS FIRST

03

Aggregate distributed evidence

Collect job states, reports, logs and metrics. Return one traceable quality result per software change

ONE RESULT PER CHANGE

Continuous HPC software quality connects software checks, scientific validation and representative execution

Thank you for your attention

Open discussion

Continuous Software Quality in HPC

The FALL3D Experience in ChEESE-2P

Heribert Pascual | GEO3BCN-CSIC

IBERGRID 2026 Quality of software, services and data

GEO3BCN
Geosciences Barcelona - CSIC

RIGHT LEVEL

CORRECTNESS FIRST

TRACEABLE EVIDENCE

CI → HPC → QUALITY



ChEESE