ROOT’s new Cost-efficient and Feature Rich GitHub-based CI

A. Naumann, S. Hageböck, J. Hahnfeld, Ole Morud, S. Muzaffar, D. Piparo (CERN, EP-SFT)
P. Canal (FNAL), V. Vassilev (Princeton)
for the ROOT Project

21-10-2024

- The problem of Continuous Integration
- ROOT's adoption of Github Actions
- Optimisations and future plans
- Conclusions

Provide a unified software package for the storage, processing, visualisation and analysis of scientific data that is reliable, performant, supported and sustainable, that is easy to use and obtain, and that minimises computing resources and scientists' time needed to achieve results.

The success of experiments and all ROOT users at large is our priority

Provide a unified software package for the storage, processing, visualisation and analysis of scientific data that is reliable, performant, supported and sustainable, that is easy to use and obtain, and that minimises computing resources and scientists' time needed to achieve results.

The success of experiments and all ROOT users at large is our priority

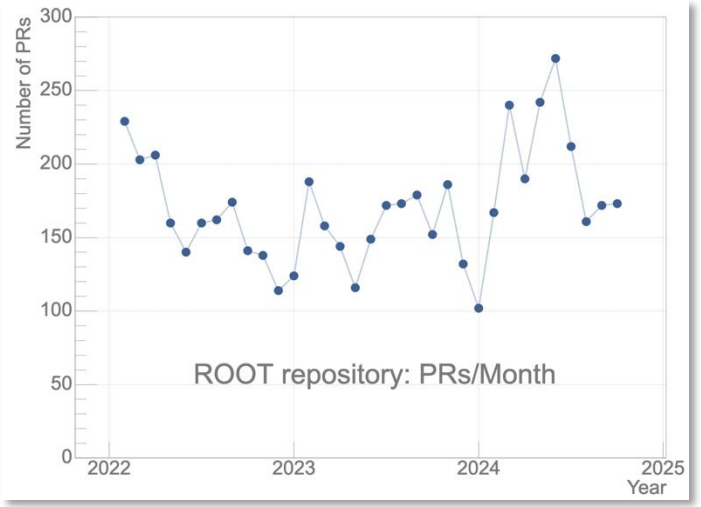
The Problem

The Problem We are Solving

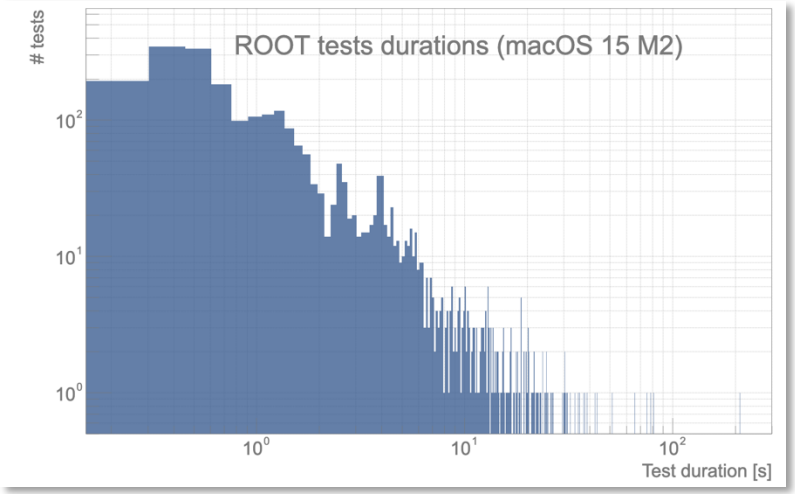
ROOT:

- **Open-source** on GitHub
- Core developers + many contributors
- **"Pull Request"-based open development model**
- Supports a rich set of platforms, on Linux, Mac and Win
- **~10k compilation targets, >3k tests**
  - ~45m-1h, depending on the computer
- Build+test: for every PR submitted/modified, every PR merged, and every night (presently for 4 branches)

  → 30-50 build+test jobs per day for all platforms!



Prolific development and thoroughness must be supported by a powerful and automated CI solution



~3000 unit, functional and integration tests run in ~30 minutes

- Made of unit, integration and functional tests in the root and roottest repositories
- **An asset for the project, built during years** collecting test cases from users, experiments and developers
- Steered by ctest: one command to run them all
- Runs approximately in 30 minutes on a modern computer/laptop using 8-16 cores

Continuous Integration (CI) system: a pool of computing resources and a scheduling mechanism for submitting build+tests jobs

Some desirable properties of a CI:

- **Automatically build and test** on all supported platforms
- Jobs submitted on **different types of triggers** (commits, PRs, time...)
- Is **easily developed**, e.g. can change and test existing build/test configurations
- **Configuration sits the codebase** and is testable
- **Exposes failures and easy recipes to reproduce** them for debugging purposes
- Free, open-source, **supported by a community and well maintained**

The Solution

Jenkins: a good product, with many useful features, that served ROOT well until 2023.

Some limitations for ROOT's development model, divided in two main categories.

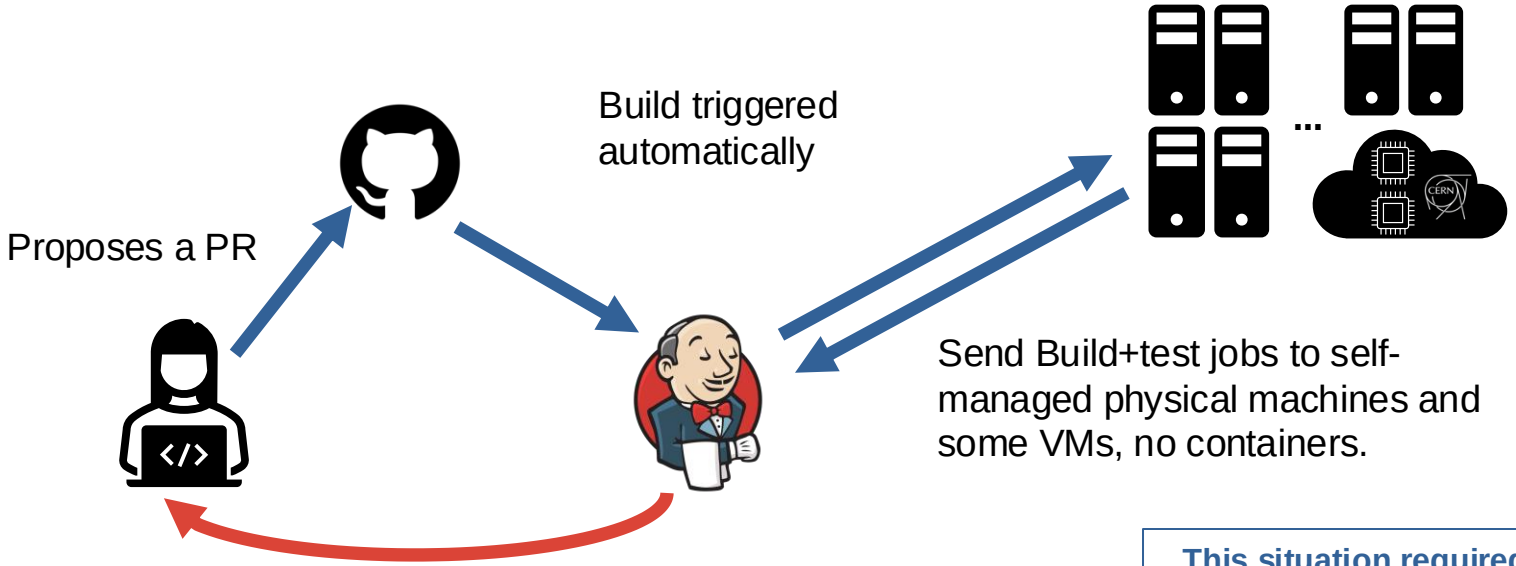
Intrinsic

- **Maintenance of the Jenkins server** (updates, fixes)
- **No native integration with GitHub** (e.g. GH roles – Jenkins roles of users/devs)
- **Job results not easily exposed to contributors** (CERN account needed)

ROOT's Deployment

- **Manually configured individual hosts**
- **Effort-intensive to add new platforms: limit the phase space probed**
- **Effort-intensive to keep instances of the same flavour in sync**

Jenkins-based CI



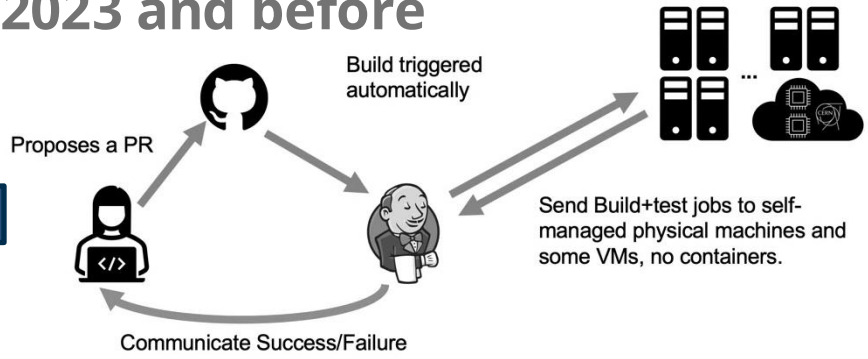
This situation required simplification and represented an opportunity for the streamlining of the building and testing workflow!

GitHub Actions (GHA)-Based CI: in Production since 2024

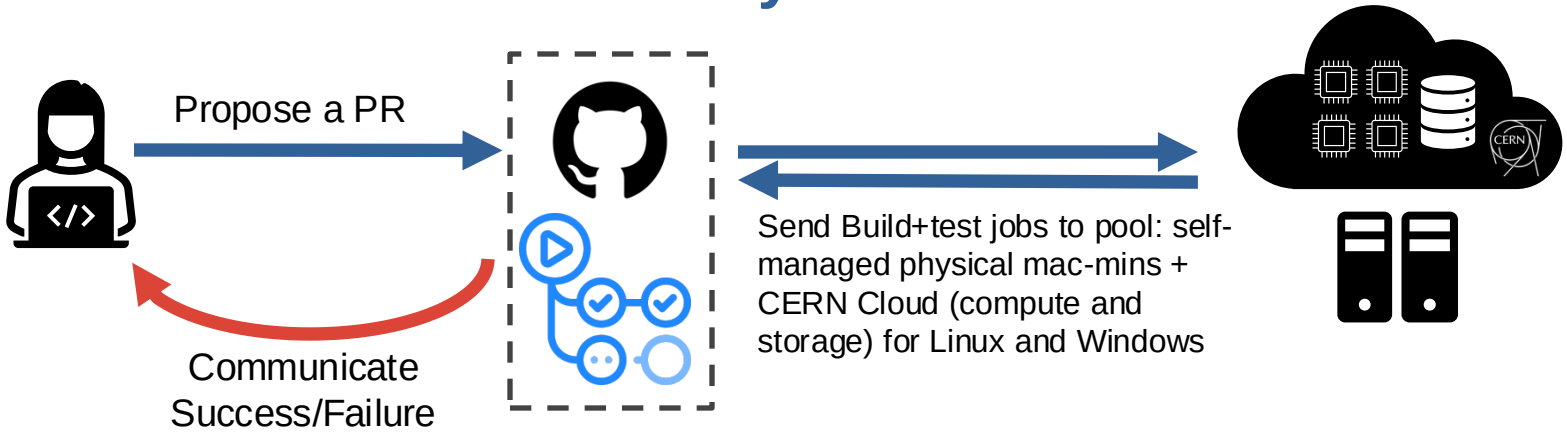
- API orchestrating workflows based on "events" (e.g. schedule, PRs...)
  - Self-hosted and GitHub cloud runners (free for public repos) possible
- Provided natively by GitHub: **no server to maintain**
- Configuration expressed in YAML files
  - **Steep learning curve,** but widely adopted, good documentation and *many* examples available
  - **Located in the project's repository (e.g. test CI changes with PRs)**
- **Easy to reproduce failures locally** with containers for debugging purposes
- **Easy to manage workers** profiting from CERN's Openstack cloud

Adapting to GHA required rethinking, simplifying, and streamlining ROOT's approach to CI

2023 and before



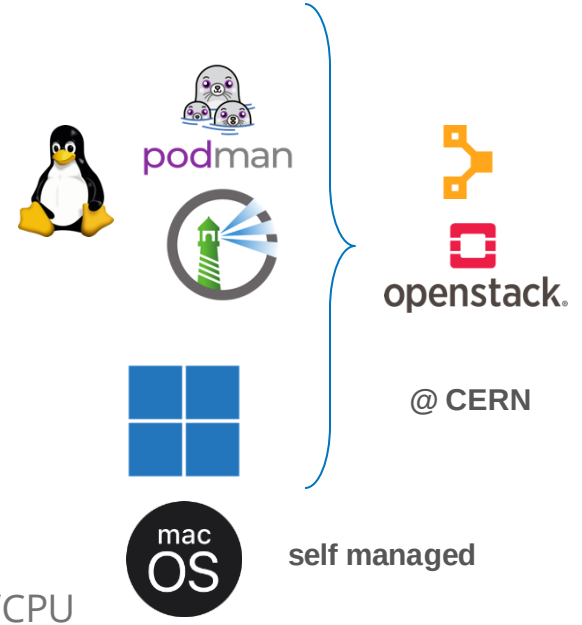
From January 2024



ROOT's Platforms in the Current CI

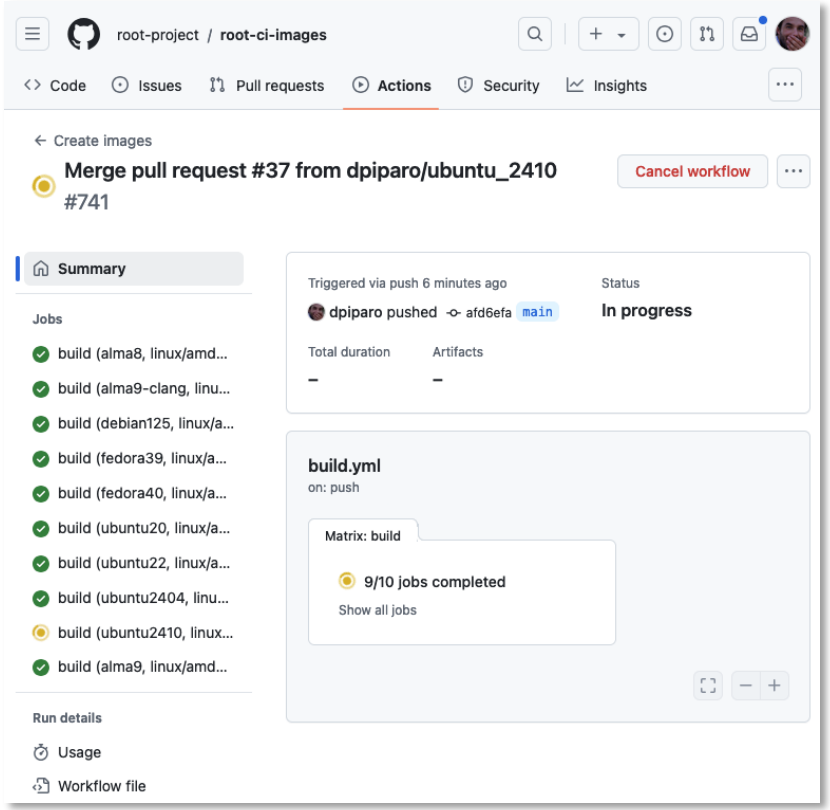
Three families of platforms: Linux, MacOS and Windows

- **Linux: ~10 flavours/configurations**
  - **Jobs run in containers,** stored in CERN's registry
  - Images configuration and building in the root-ci-images repo
  - **Puppet managed host VMs on CERN's Openstack instance***
  - **Fleet: ~40 16-Cores servers**
- **8 Windows 10 VMs** in CERN's cloud
- **macOS: 3 versions (13, 14, 15 as of today) + Beta**
  - **8 Mac mini** – the cheapest `24 configuration has enough RAM/disk/CPU



** Thanks to the CERN IT-CD group for this scalable, powerful and reliable service!*

Interlude: Creation of Containers



- root-project/root-ci-images: configuration of the CI containers
- Also relies on CERN's containers' registry
- Images built daily and at every change using GH cloud runners

A powerful and automated way to build CI container images based on GitHub Actions

Latest News: ARM and GPU Runners!

Note: ROOT supports NVidia GPUs today, in release, e.g. to boost RooFit and TMVA.

Very fresh: in commissioning since 3 weeks

- **ARM64 and x86+GPU: setup identical to x86-64 one**
  - Same Puppet configuration of x86 hosts
  - Containers in the CERN registry
  - Nvidia T4
  - 2 generic builders for each flavour

Test ROOT on ARM and GPUs continuously, to guarantee even more stability to experiments and users

Leveraging the Power of Actions

Expertise acquired to craft workflows usable for a variety of procedures:

- Building and publishing the ROOT website with Jekyll 

- Harvesting the ROOT issues addressed for a certain release 

- Building and publishing ROOT Doxygen Documentation (in progress)

Strategy: automate as many operations as possible through GH actions

GitHub actions: helpful not only for building and testing

Improvements and Future Plans

- **CMake dependencies were optimised** to reduce build time and maximise parallelism
- **Tests duration was reviewed periodically and reduced**:
  - Parameters values refined (e.g. # of iterations/events/threads) w/o reducing test coverage
- macOS (and ARM): **ccache active on the nodes**
  - Considerable speedup, works flawlessly since many months
- **Incremental PR builds**, w/o keeping build artifacts on the nodes
  - From CERN S3 storage download build artifacts (libraries, .o files...)
  - For every PR merged, a build+test job is executed. If successful, build artifacts are uploaded to S3

Future Plans

- There is room for further optimisation in the **targets dependency tree**
- Use **VMs for mac** (on mac nodes, e.g. via Parallels)
  - Reduce the partitioning of the pools of CPU resources as much as possible
- Explore publication of results via my.cdash.org (xRootD is there!)
  - Easily browsable summaries, follow the history of rare, hardly reproducible failures
- Use the **new CI also to generate and publish ROOT Python Wheels**
  - `pip install root` (a feature currently in alpha-mode)
- Use the runners provided by GitHub even more
  - Currently only used to build container images and the website

pip install ROOT

Conclusions

The Migration to the GitHub CI was a Success for ROOT

- In production since 2024: ~10 Linux, 4 mac, 4 Win flavours - 30/50 build+test jobs daily
  - Self-hosted runners, resources from CERN Openstack + self-managed mac-minis
- **ROOT quality improved , costs were reduced**  ROOT's Plenary Talk
  - More builds, running more tests on more platforms and no Jenkins server to maintain
- **Streamline: behave like many other open-source products hosted on GH**
- **Matches well ROOT's open-source approach & open-development process**

Do you have experience with GHAs? Are you interested to work with us?
Feedback? Suggestions? Talk to us during CHEP or drop an email!