Upgrading the software release workflow

Thomas Madlener, Federico Meloni
(DESY)

Physics and Detector simulation
and
MuCol WP2 meeting
10/12/2024

Co-funded by
the European Union

Software distribution

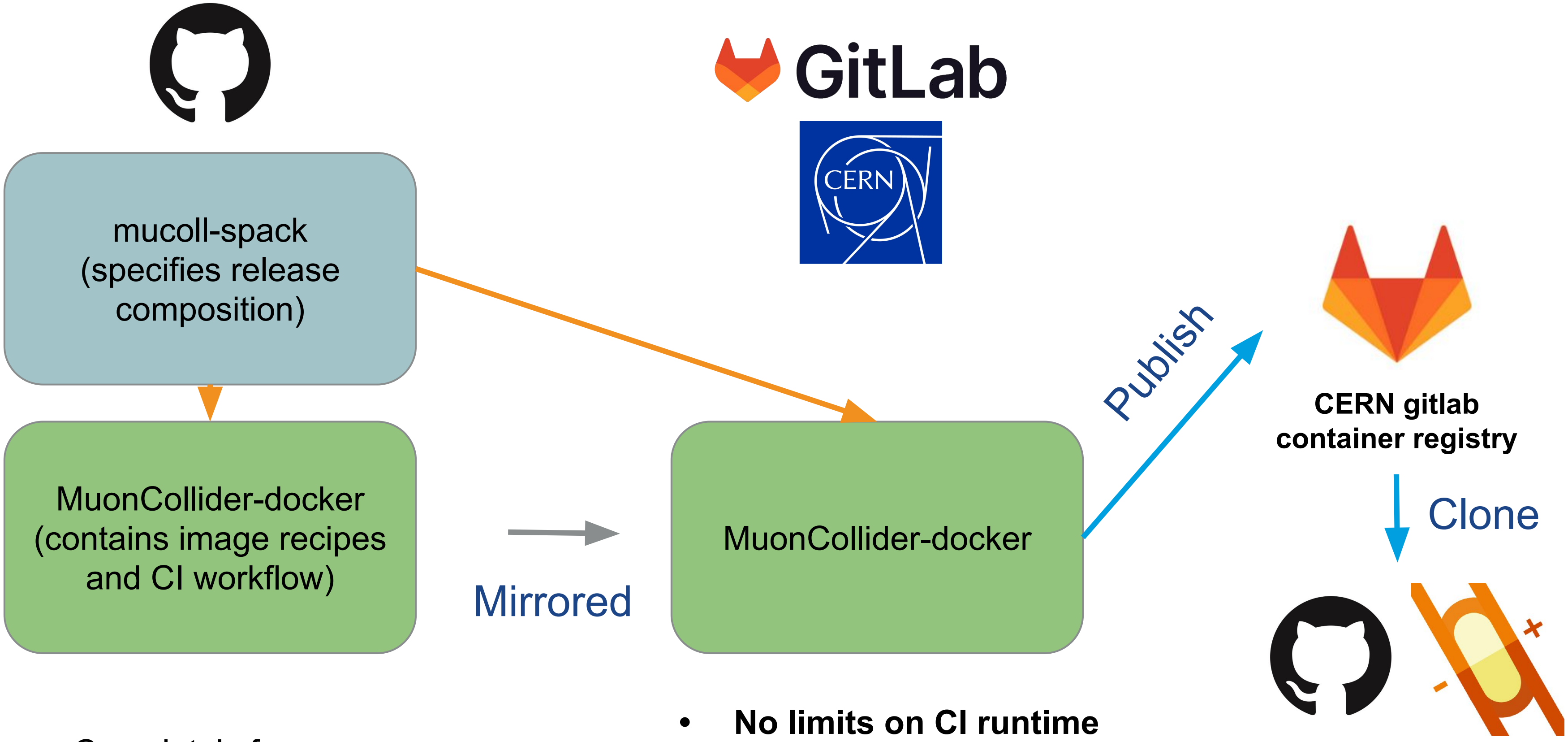
IMCC software task force recommendations

- IMCC should focus on a single avenue for software distribution
  - docker images were selected as the most modern/portable solution
  - optionally replicated to cvmfs via unpacked
- **Images should be automatically built using CI tools**
- **Images should be centrally published in a single container repository**
  - **ghcr.io feels like the most natural choice so that containers can be directly linked to release notes/pages**

Roadmap / prioritization

1. Setup CI pipeline to build releases ✅
2. Export images to ghcr.io [0.2 FTEmonth]

Workflow (as of release 2.9)



mucoll-spack (specifies release composition)

MuonCollider-docker (contains image recipes and CI workflow)

Mirrored

GitLab

CERN

MuonCollider-docker

Publish

CERN gitlab container registry

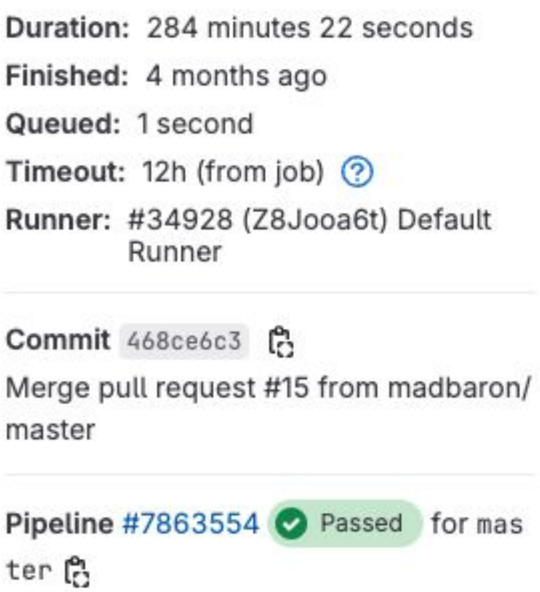
Clone

GHCR.io

- Completely free
- Highly discoverable
- **Runtime limited to 6h**
- **Limit to 100GB disk**

- **No limits on CI runtime**
- **Better CPUs**
- Mirroring a little unreliable
- Requires maintaining a number of tokens
- Publishes images to cern docker registry

Why is the build workflow so long?



Duration: 284 minutes 22 seconds
Finished: 4 months ago
Queued: 1 second
Timeout: 12h (from job)
Runner: #34928 (Z8Jooa6t) Default Runner

Commit `468ce6c3`
Merge pull request #15 from madbaron/master

Pipeline #7863554 Passed for `master`

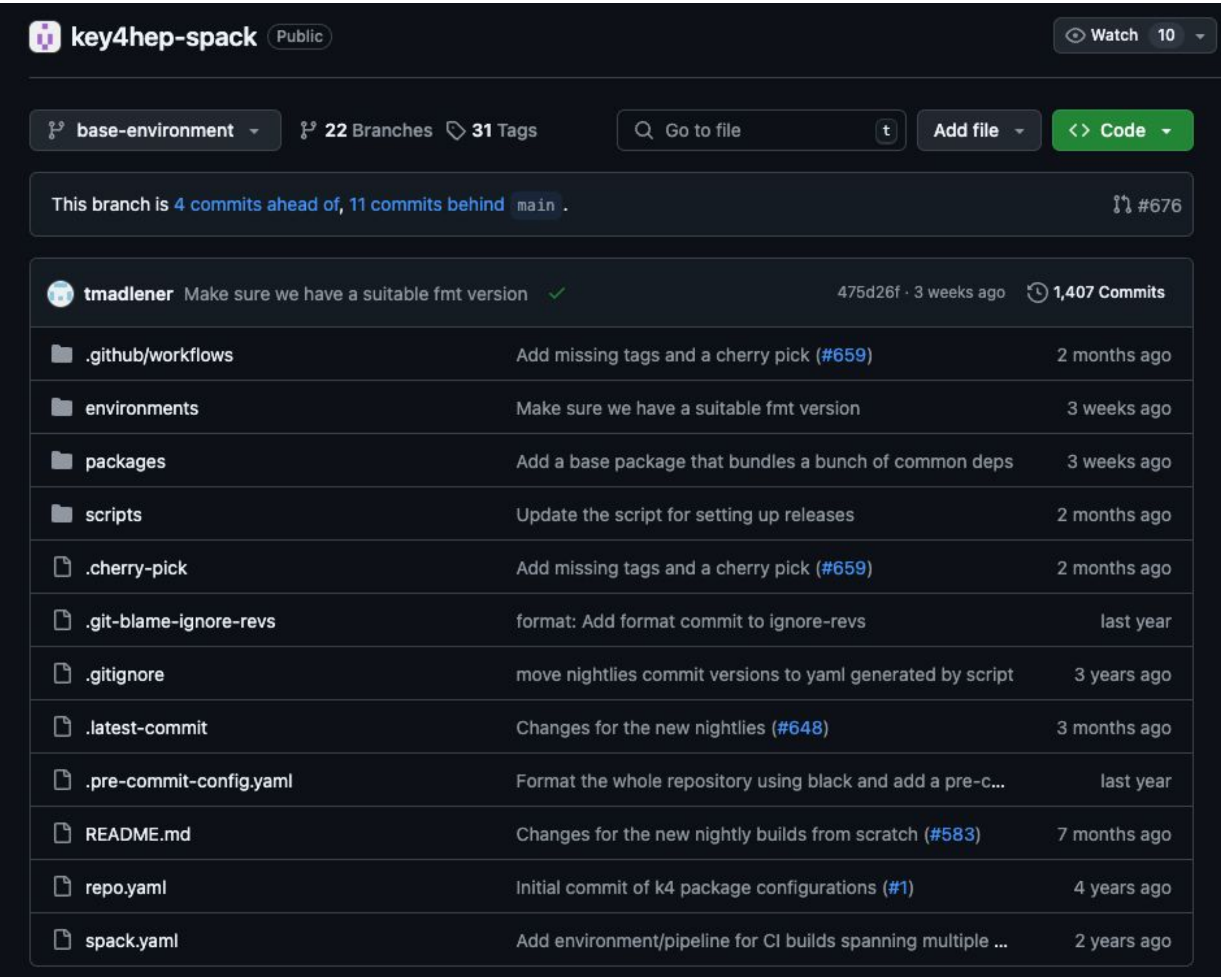
Release 2.9 build on gitlab

The workflow using spack requires by default to rebuild the whole spack recipe

- This means compiling a substantial amount of “external” packages that are not directly part of our software stack
- Compiling these takes 60-70% of the total time

These packages are updated relatively rarely

Key4hep-base



This branch is 4 commits ahead of, 11 commits behind `main`. #676

tmadlener Make sure we have a suitable fmt version ✓ 475d26f · 3 weeks ago 1,407 Commits

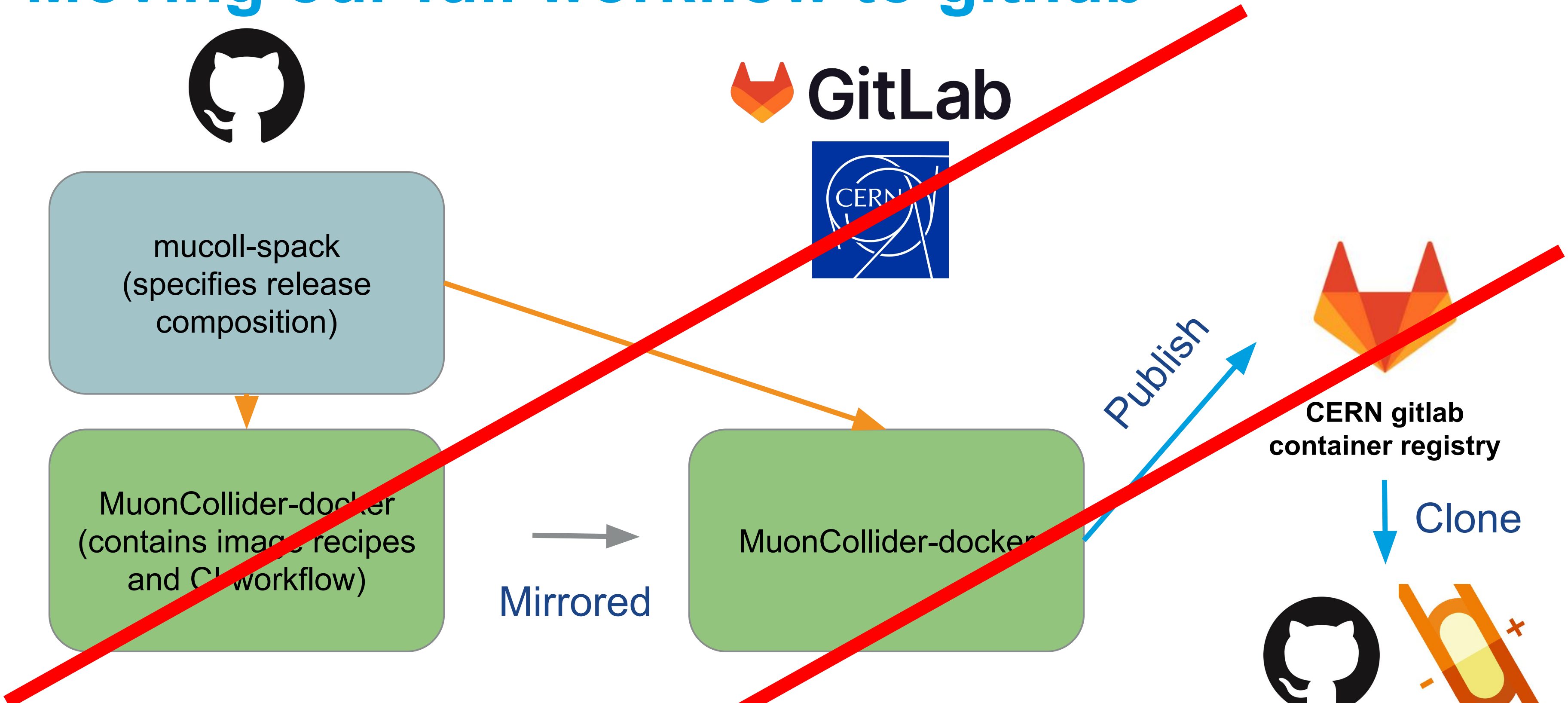
.github/workflows	Add missing tags and a cherry pick (#659)	2 months ago
environments	Make sure we have a suitable fmt version	3 weeks ago
packages	Add a base package that bundles a bunch of common deps	3 weeks ago
scripts	Update the script for setting up releases	2 months ago
.cherry-pick	Add missing tags and a cherry pick (#659)	2 months ago
.git-blame-ignore-revs	format: Add format commit to ignore-revs	last year
.gitignore	move nightlies commit versions to yaml generated by script	3 years ago
.latest-commit	Changes for the new nightlies (#648)	3 months ago
.pre-commit-config.yaml	Format the whole repository using black and add a pre-c...	last year
README.md	Changes for the new nightly builds from scratch (#583)	7 months ago
repo.yaml	Initial commit of k4 package configurations (#1)	4 years ago
spack.yaml	Add environment/pipeline for CI builds spanning multiple ...	2 years ago

Thomas implemented a “minimal” base environment (and bundle package) within the key4hep release that contains all those external dependencies

- Enough to build gaudi and podio on top of it

Takes about 5h30 to build on the github CI…

Moving our full workflow to github

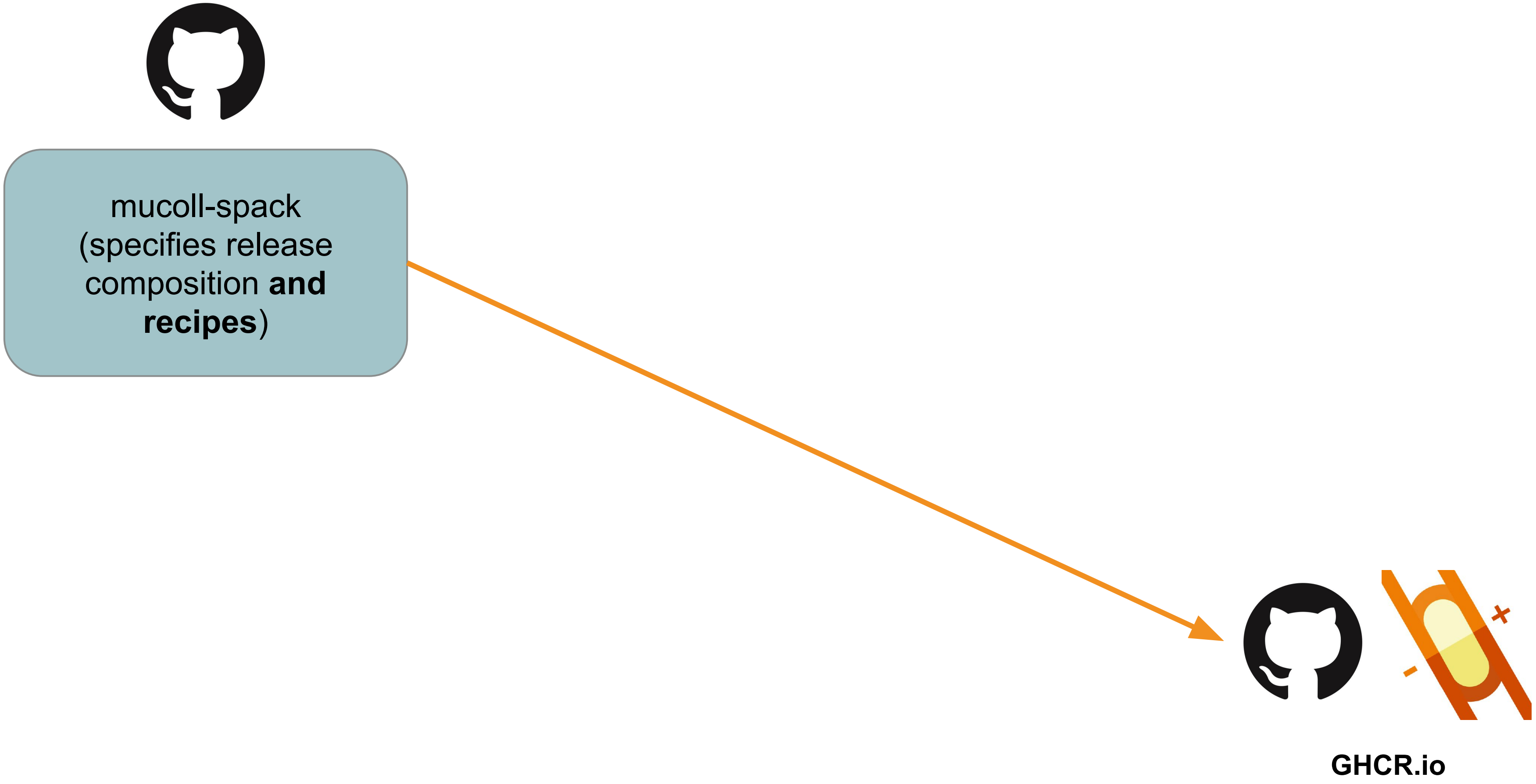


GHCR.io

- Completely free
- Highly discoverable
- **Runtime limited to 6h**
- **Limit to 100GB disk**

- **No limits on CI runtime**
- **Better CPUs**
- Mirroring a little unreliable
- Requires maintaining a number of tokens
- Publishes images to cern docker registry

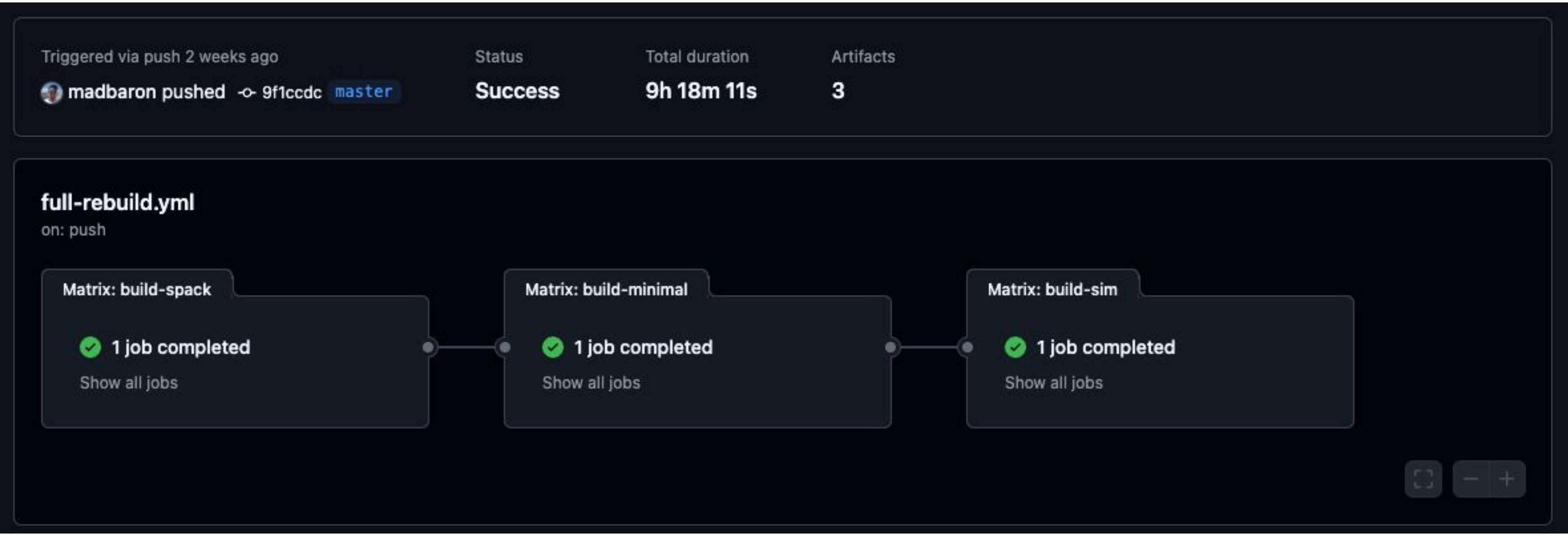
Moving our full workflow to github



mucoll-spack (specifies release composition **and recipes**)

GHCR.io

Exercising the workflow



Triggered via push 2 weeks ago

madbaron pushed -o- 9f1ccdc master

Status

Success

Total duration

9h 18m 11s

Artifacts

3

full-rebuild.yml

on: push

Matrix: build-spack

✓ 1 job completed

Show all jobs

Matrix: build-minimal

✓ 1 job completed

Show all jobs

Matrix: build-sim

✓ 1 job completed

Show all jobs

Workflow split in 3 steps:

- Build-spack (checkout repositories and setup environment)
- Build-minimal (build the key4hep-base environment)
- Build-sim (build mucoll-stack on top of key4hep-base)

Workflow triggered on changes to Dockerfiles and Spack recipes

- **Re-run only the steps that you need**

Highlighting some details

All changes included in PR (https://github.com/MuonColliderSoft/mucoll-spack/pull/17)

Most visible change is that the **Docker workflows are moved over from MuonCollider-Docker to mucoll-spack**.

- This makes MuonCollider-Docker redundant. Deprecate and archive?

The previous concretize workflows inherited from the key4hep projects will be run only on PRs (the image publication on the registry can't be run on PRs to public repositories, per github policy).

Key4hep-base not yet merged in upstream key4hep-spack.

- Could move it to mucoll-spack if for some unlikely reason it is rejected

Future steps

The preparation of the mucoll-minimal image (which is really a key4hep-minimal) would be best moved to key4hep-spack as it is completely generic

- Key4hep currently only releases software via cvmfs

Publish images also to dockerhub and CERN gitlab registry?

Plan to set up a cron-triggered workflow to run as nightly build with the latest versions of all packages

We can review the list of packages to be compiled (i.e. all event generators) to make the resulting image a little slimmer

- Can also envision more dedicated images (i.e. “analysis-only”, as noted in the IMCC software task force recommendations

Thank you!

Starting the wrap-up of the task force

The software task force has met 8 times

- Discussed all items in mandate
- Most activities seeded / exercised in getting here
  - Notable outstanding item is setting up the central documentation repository (more on this in a moment)

Move towards wrap-up of activities

- Reporting findings in this meeting
- This slide deck is meant as documentation of effort (but another format can be considered, if useful)

Software development

Recommendations

- The software repositories should be publicly hosted on github, under **MuonColliderSoft**
- A central developer team in charge of repo creation/management
  - Team is open and people can join at any time

- The development approach should stick to “micro-services” (i.e. current approach)

- Cross-examination of MR should be enforced
  - Use tools to fix / point out the trivial stuff: clang-format, black (for formatting), clang-tidy, sanitizer builds

- Schedule dedicated discussions to discuss major software changes at the end of tuesday meetings
  - Advertise discussions on sw mailing list / any github issue participant can initiate

- Use the **GitHub Projects** interface for task management, with a structured list of **Labels** and **Milestones** to be used in **Issues** and **PRs** across repositories
  - Roadmap set up by this TF and repeatedly advertised in Tuesday’s meetings
  - Follow-up efforts will need to pick it up and keep it up to date

Integration into key4hep

Recommendations

- IMCC should move away from maintaining forks of general key4hep packages and rather push the changes to the upstream repositories
- Only muon collider-specific packages should be hosted under MuonColliderSoft
- IMCC should maintain independence when building releases
  - Mucoll-stack configurations should be moved to key4hep-spack package
  - Mucoll-spack to be maintained as fork of key4hep-spack to tag and build muon collider software releases
- To minimise any compatibility issues, IMCC should move to key4hep standard tagging schema

Software development

Roadmap

Super short-term ✅

1. Initial release for ESPPU
   - Based on key4hep-release + mucoll-spack and muon collider packages (essentially v02-08-MC + updates)
   - Switched to ACTS v32 + migrated ACTSTracking
   - “Current” detector geometries for MUSIC and MAIA not included
   - Update tutorial/example Marlin configuration files to python
   - Based on tagged CI

Short-term (fall)

1. Complete the upstreaming of the muon collider changes to key4hep packages [~1 FTEweek]
   https://github.com/MuonColliderSoft/ConformalTracking
   https://github.com/MuonColliderSoft/MarlinTrkProcessors
   - Set up mucoll-release folders within key4hep-spack
   - Get developer access to manage our own changes
2. Merge bug-fixes from initial ESPPU release (2.9) [~1 FTEweek]
   - Adjust ACTS seeding [2 FTEweeks]
3. Include MUSIC and MAIA detector geometries
4. Build release image and distribute

Software development

Roadmap

Medium term (~2025)

1. Migration to gaudi/edm4hep-based workflow
   - Migrate digitisers to gaudi [1 FTEmonth]
   - Migrate/re-design Overlay processor for gaudi [1 FTEmonth]
   - Output edm4hep files by default

2. Configure Issue templates in each repository that is part of the release

3. Organise a systematic review of code to identify any ee assumptions (e.g. occupancy) are baked into the framework. [6 FTEmonths]
   - Output a list
   - Based on the list, prioritise ILCsoft packages updates (Gaudi framework should be good / LHC-tested)

4. Muon PID for 10 TeV detector concepts [3 FTEmonths]

5. Setup workflows to build specialised software images (i.e. analysis only) [1 FTEmonth]

Software development

Roadmap

Long term (2026+)

1. Tracking [4 FTEmonth]
   - Load tracking geometry from dd4hep plugin - being tackled community-wide, requires changes to ACTS
   - Update again to latest and greatest ACTS (currently v35.2.0)

2. Calo clustering [6 FTEmonths]
   - Test/Use of more modern clustering tools, e.g. k4Clue

3. PFO reconstruction [30 FTEmonths]
   - Move away from PandoraPFO (worldwide expertise problematic)
   - Switch to simpler / non-monolithic algorithms
   - Break down workflow small configurable algorithms
   - Get rid of hardcoded PFO assumptions (e.g. taking always track p_T over calo measurement even at high energy)
   - Need to monitor wider community and possibly move in synch

Fast Simulation

Recommendations

- IMCC should centrally provide up-to-date Delphes cards for phenomenology studies
- The centralised cards should reflect the “Baseline” and “Aspirational” scenarios described in the “Interfaces” chapter of the Interim report (or the analogous location in future documents)
  - Each detector concept is welcome to provide a “state-of-the-art” card reflecting the currently achieved performances in full simulation (but this is not considered mandatory)
  - Should be labelled with detector concept name to avoid confusion with central IMCC cards
- IMCC should also provide tools to produce Delphes cards given a geometry description + simulated datasets
  - These could be distributed as a part of the mucoll-benchmarks repository
- A note with “Terms and conditions” should be distributed with the cards to avoid misuse where fake objects become relevant
  - Develop “BIB object libraries” for overlay (for tracks and calo clusters) to address this in the long-term
- Cadence of maintenance: at least once per major report
- Maintenance responsibilities: synthetic “baseline” and “aspirational” cards to be provided by IMCC, detector-concept-specific by concept teams

Fast Simulation

Roadmap / prioritization

1. Write “Terms and conditions” - embed them in Delphes card and publish them on https://mcd-wiki.web.cern.ch/
2. Generate “synthetic” Delphes cards for “Baseline” and “Aspirational” scenarios [0.5 FTEmonth]
3. Develop tools to extract delphes parameterisations from object performance plots [1 FTEmonth]
4. Detector concepts provide cards for “state-of-the-art”
5. Work on automation of state-of-the-art cards via github CI/CD [1 FTEmonth]
6. Develop “BIB object libraries” for overlay (for tracks and calo clusters) to enrich fast-simulation [3 FTEmonth]

Documentation

Recommendations

- Public, no need for private documentation
- Minimise links to external documentation, even if we have to repeat some things, to avoid out-of-synch material
    - Links should include the exact version instead of the latest one
- Tools / platform
    - github preferred for openness
    - Mimic Key4hep approach (one central repository with general info and CI, pulls in from several other repositories on demand to build complete key4hep documentation)
    - Use auto-documentation where sensible (for API/package level)
        - Requires work to set up EOS folder to be auto-published to the web
        - Doxygen for generating API documentation
- A responsible for keeping the documentation building + auto-publishing should be found
- Pace of updates / versioning
    - Maintain “current” version and tag specific version for events
- IMCC should publish a recommended documentation format for all READMEs in packages.
    - Consistent naming scheme and structure for packages
    - Use of doxygen syntax for all languages.

Tutorials

Recommendations

- The tutorials should cover the full computing chain (sim / reco / analysis)
  - They should have entry points per chapter, with no reliance on previous steps
- Tutorials should include runnable minimal examples with useful code snippets
  - Auto-test in documentation as in Key4hep
- Technical developer-oriented instructions should be given in README.md of the relevant GitHub repositories
- Provide wiki instructions on how to update the wiki itself and a dedicated tutorial section
- Tutorials should cover the use of github project management tools (e.g. issues…)
- Integration of training in other ongoing efforts
  - The analysis step is generic enough that could be merged with key4hep tutorials
  - “General” training can be (partially) deferred/integrated into HSF training https://hepsoftwarefoundation.org/workinggroups/training.html

Tutorials and documentation

Recommendations

QA responsibilities:

- Package-level - rely on documentation schema check at MR level
- Top-level - rely on group of developers approving merge request to protected branch
- Tutorials - rely on team of tutorial organisers

Roadmap / prioritization

1. Remove obsolete documentation pages ✅
2. Setup central documentation repository
3. Setup auto-publishing on the web
4. Provide templates for package-level documentation
5. Set up auto-parsing of documentation (CI fails if README is out of date)
6. Set up auto-parsing of tutorials (CI fails is snippets don't succeed)