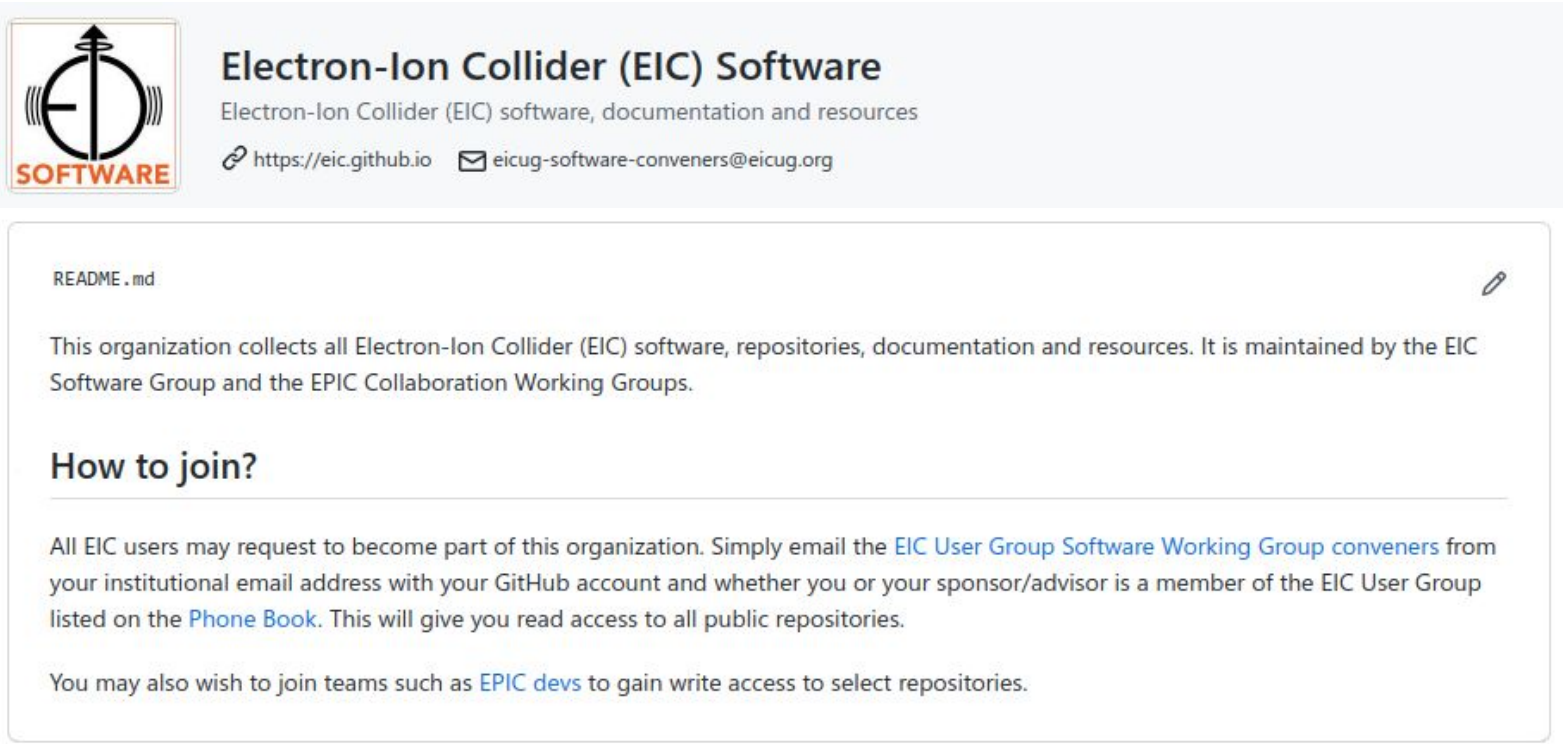
GitLab to GitHub Transition Status

Convener Summary (doc):

- We will implement a hybrid solution that uses GitHub as the primary repository, while using the eicweb GitLab instance for CI/CD.
- An ad hoc committee of eicweb experts will investigate the best option for leveraging CI/CD at ANL using GitHub (e.g. GitHub runners, mirrors, webhooks, etc.
- The existing “eic” organization at GitHub established by the EICUG Software WG will be used.
    - Some admin privileges will need to be shared with the Detector-1 WG conveners
- The best practices model for the repository will include:
    - Repositories will be open and public unless there is a specific reason to make them private
    - External packages will not be forked/cloned to the eic organization and modified unless under extremely exceptional circumstances.

GitHub Organization Management



GitHub Organization Management

Requirements for joining (44 members as of 2022-08-09):

- You or your sponsor/advisor is listed on the EIC phonebook.
- Administered through EICUG SWG conveners email.

Admin team @eic/admins (Andrea, Markus, Maxim, Torre, Wouter):

- Public team chat for reporting issues/concerns/requests.
- Private team chat for discussing admin settings changes, etc.

Additional teams (10 teams as of 2022-08-09):

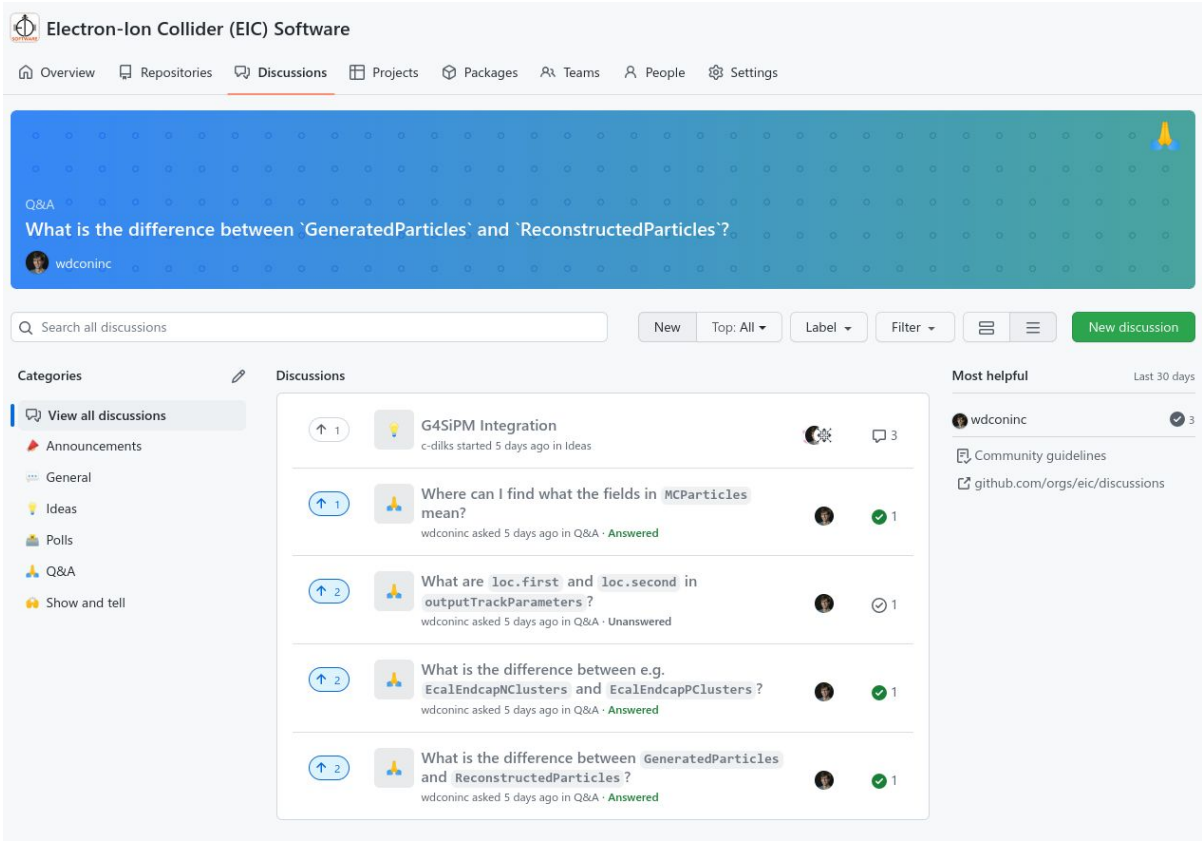
- @eic/epic-devs: write permission to EPIC-related repositories (branches)
- @eic/epic-admins: admin permission to EPIC-related repositories (settings)
- … (requests at @eic/admins)

GitHub Discussions

Think: StackOverflow

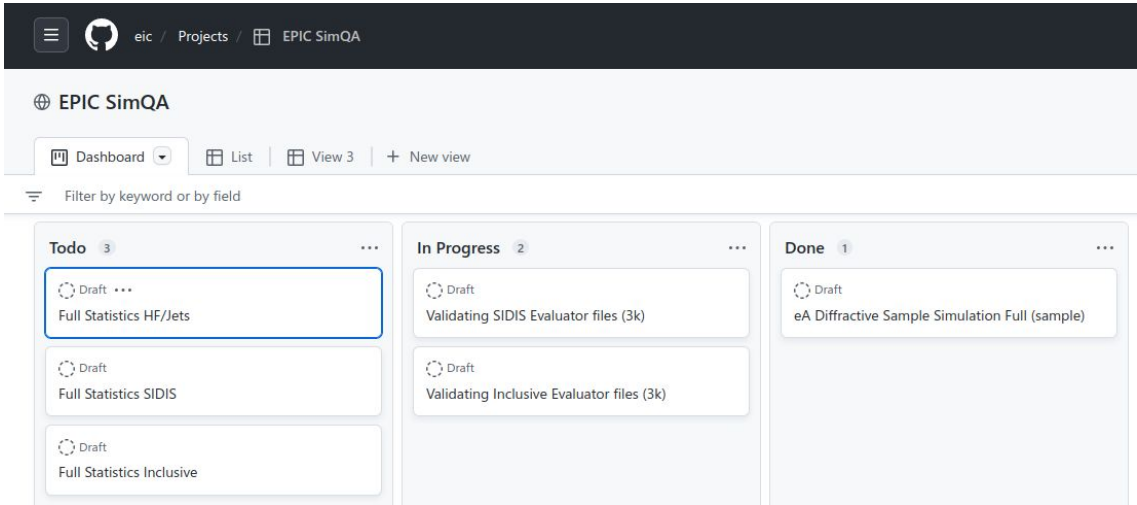
Used for Q&A/FAQ, but also discussion of features etc.

Individual repositories can open their own discussion pages, but this is an area where that may result in fragmentation.



GitHub Projects

- Project boards (kanban), e.g. EPIC SimQA
  - Default is private view, ask @eic/admins to set public if so desired
- Not all tasks must be issues, but can assign (or turn into) issues later
- Individual repositories can open their own project pages, but this is an area where that may result in fragmentation.



GitHub Migration: Pull Mirroring Not Possible

Issue realized early on after decision:

- Push mirroring from eicweb to github.com has always worked.
- Pull mirroring from github.com to eicweb became paid-only in ~August 2021.
- Push mirroring while accepting PRs on github.com leads to push conflicts.

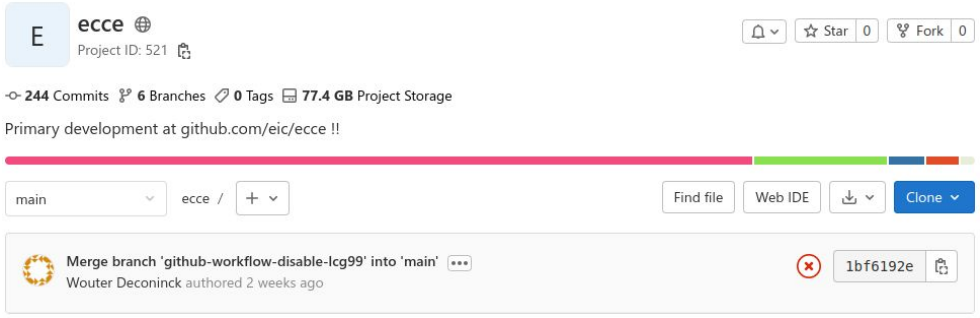
Therefore, all migrations require a **full migration and clean break** with eicweb:

- eicweb repository archived, read-only, and marked protected on all branches.
- Benchmark pipelines should not pull from old (stale) eicweb repositories.
- Scope of migration forced to include shift of some CI pipelines:
  - Cannot rely anymore on eicweb pipeline after pull mirror (since eicweb is now stale)
  - Cannot rely on eic/trigger-gitlab-ci (since eicweb is now stale)

Geometry Repositories: EPIC and IP6

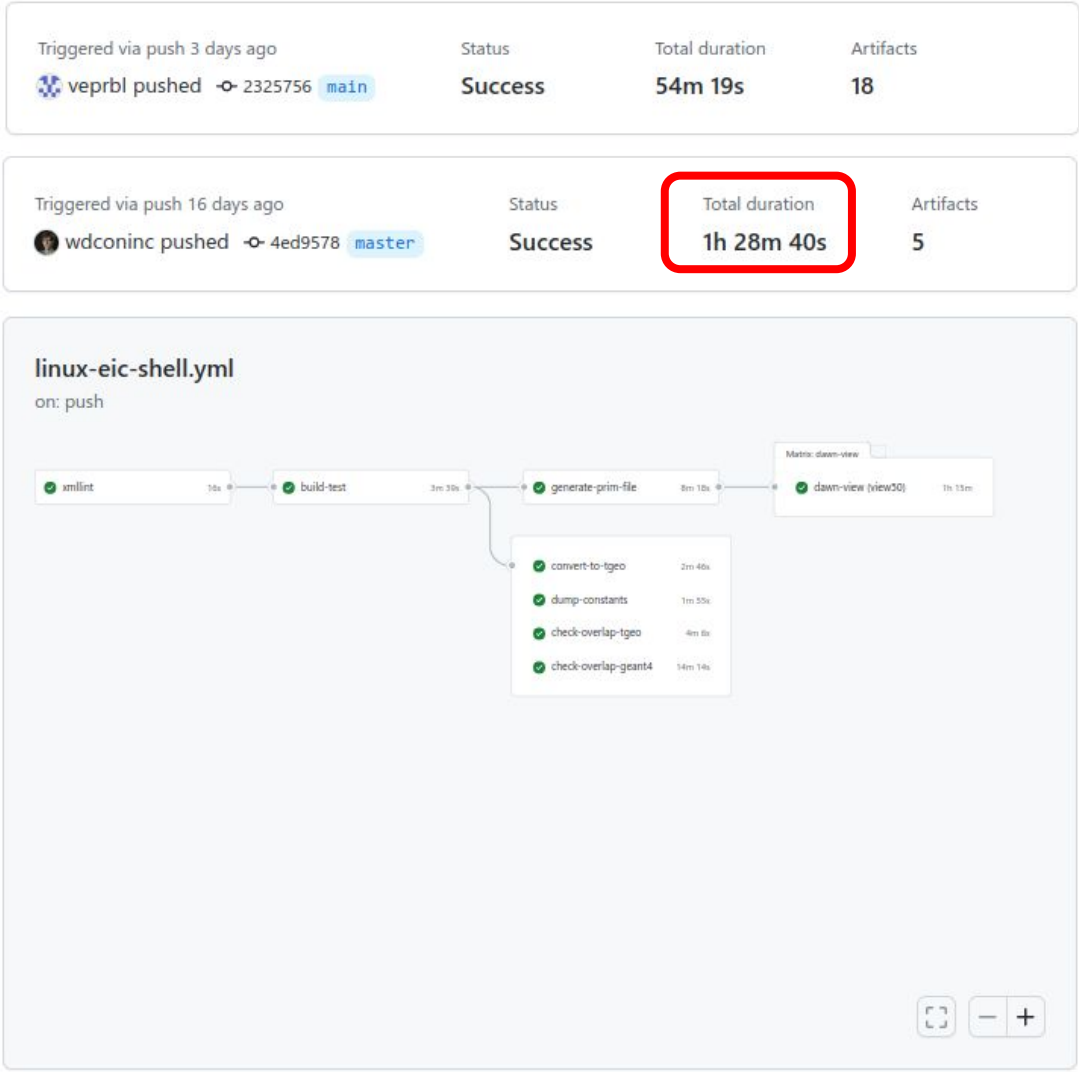
Move completed from eicweb to ~~eic/ecce~~ eic/epic and eic/ip6

- Separate repositories since downstream infrastructure still assumes this.
  - No immediate plan for merging since it would simply add more work right now.
- Primary geometry development now happening completely on GitHub
- Plan to start regular weekly meeting review of issues, PRs, and WIPs
- All merge requests require passing CI pipelines, one review, resolving all open discussions
- Still dealing with callback issue on EPIC pipeline triggered on eicweb, which never turn into completed
- IP6 does not currently trigger EPIC pipeline (overlap checks)



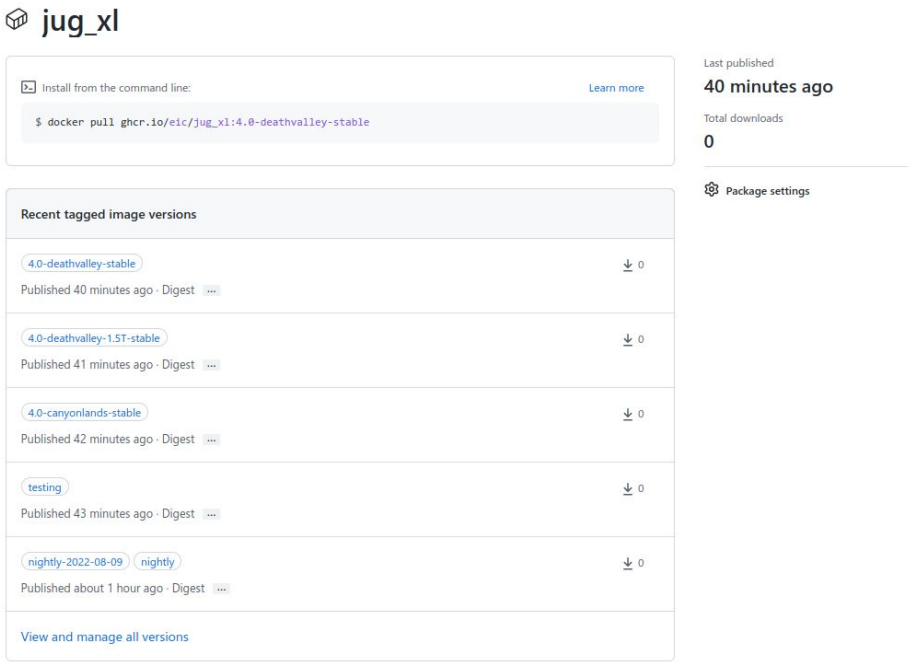
GitHub Pipelines

- EPIC: ~50+ jobs on GitHub
- IP6: ~10+ jobs on GitHub
- Triggering of downstream eicweb benchmark pipelines
- GitHub Actions infrastructure:
  - eic/trigger-gitlab-ci
  - eic/run-cvmfs-osg-eic-shell
  - AIDAsoft/run-lcg-view
  - cvmfs-contrib/github-action-cvmfs
- Persistence on self-hosted runners is major advantage that we have to work around on GitHub (1GB artifacts…)



GitHub Pipelines

- eic/run-cvmfs-osg-eic-shell appears preferred: lowest download overhead, fastest time to payload (30s with cache hits)
- Alternatives available:
  - CERN LCG views (older DD4hep/podio)
  - jug_xl containers (now pushed to ghcr.io)
    - essentially the same as eic-shell
- Important concerns for pipelines:
  - should allow rapid feedback on commits, more detailed feedback on pull requests
  - stay well under 250 jobs to avoid backlogs
- Next possibilities:
  - posting benchmark results as comments
  - clang-tidy suggestions on PRs



EIC Data Model

Currently still using eicd on eicweb, preparing for migration to eic/eicd.

New issues with this migration:

- eicd changes require a pipeline to evaluate whether consistency is maintained in algorithms, triggering a pipeline in juggler
- juggler pipeline builds new local container, then triggers benchmarks in the local container

This is a trickier set of pipeline interactions than EPIC and IP6 repositories.

Juggler Algorithms

Separation of juggler algorithms from Gaudi parts, into a new `algorithms` library

- `algorithms` depends only on DD4hep and EDM4hep (and e.g. EDM4eic)
- `algorithms` used by juggler to provide algorithms (juggler provides services for DD4hep, IO, random numbers; Gaudi provides orchestration)
- `algorithms` used by EICrecon to provide algorithms (EICrecon provides services for DD4hep, IO, random numbers; JANA2 provides orchestration)
- During transition, continued development of `algorithms` algorithms possible
- After transition, `algorithms` as starting point of joint key4HEP efforts

Advantages: algorithm provenance and full commit history; continuous production environment; faster implementation in EICrecon; ability to cross-check EICrecon

Juggler continues to exist as thin Gaudi layer around these `algorithms`.

Juggler Algorithms

Implementation details to provide functionality independent of Gaudi/JANA2):

- Units: internal use of DD4hep units only
- Logger facilities (e.g. `error() << “...” << endmsg;`)
- Properties (e.g. `Property m_tRes{“timeRes", 0.0d * dd4hep::ns};`)
- Algorithms with `initialize()`, `execute()`, `finalize()`, where `execute() const` takes EDM4hep collections

Proof of concept with `CalorimeterHitDigi` (took about 30 minutes to convert, which involved writing some supporting classes)

Container Infrastructure

- Based on spack for scalability and not reinventing wheel (also: key4hep):
  - gcc-11.3.0 (c++17 requirement), ROOT 6.26.02, geant4 10.7.3, more with `eic-info`
- Unlikely to migrate actual container building to GitHub (time, size, etc…)

EIC spack repository with custom packages now fully migrated to eic/eic-spack

- Includes overlays, e.g. local geant4 patches to modify Birks' constant, etc…
- New package versions, bug reports should be submitted there
- Branch `develop` tracks `spack:develop`, `v0.18` tracks `spack:v0.18`, tags on versioned branched simply sequential
- Some more CI pipelines need to be developed for automated checking
- eic/eic-spack-cvmfs-environments provides pathway to propagate environments used in eic-shell to cvmfs as well (in progress)

JANA2 and EICrecon

- JANA2 not migrated to github.com/eic yet
- EICrecon actively being developed on GitHub

More in David’s talk.