

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



Electron-Ion Collider (EIC) Software

Electron-Ion Collider (EIC) software, documentation and resources

<https://eic.github.io> [✉ eicug-software-conveners@eicug.org](mailto:eicug-software-conveners@eicug.org)

README.md 

This organization collects all Electron-Ion Collider (EIC) software, repositories, documentation and resources. It is maintained by the EIC Software Group and the EPIC Collaboration Working Groups.

How to join?

All EIC users may request to become part of this organization. Simply email the [EIC User Group Software Working Group conveners](#) from your institutional email address with your GitHub account and whether you or your sponsor/advisor is a member of the EIC User Group listed on the [Phone Book](#). This will give you read access to all public repositories.

You may also wish to join teams such as [EPIC devs](#) to gain write access to select repositories.

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.

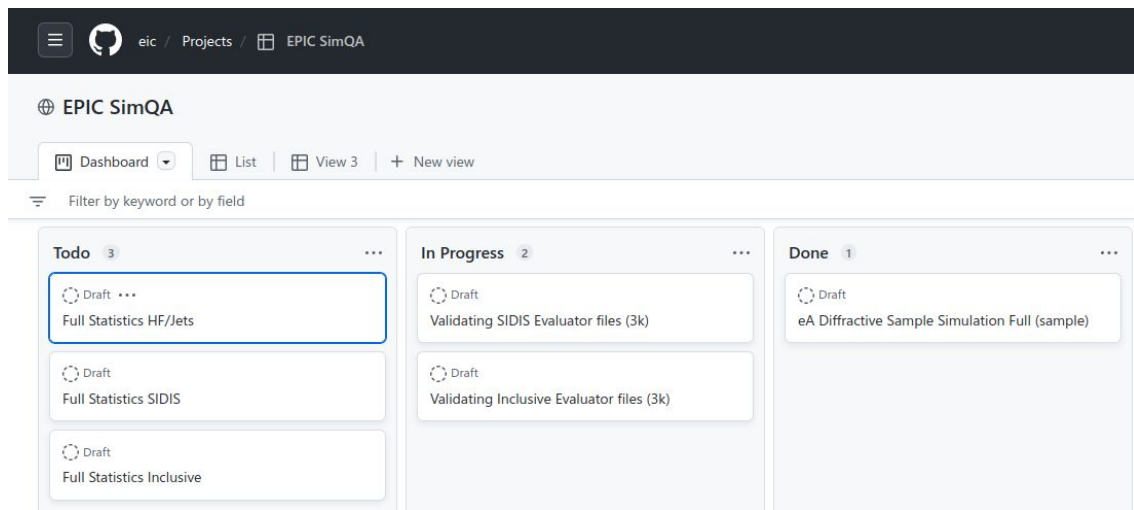
The screenshot displays the GitHub Discussions interface for the 'Electron-Ion Collider (EIC) Software' repository. The navigation bar includes links for Overview, Repositories, Discussions (active), Projects, Packages, Teams, People, and Settings. A search bar and filters (New, Top: All, Label, Filter) are present. The main content area shows a list of discussions:

- G4SiPM Integration** (1 upvote) by c-dilks, started 5 days ago in Ideas. 3 comments.
- Where can I find what the fields in MCParticles mean?** (1 upvote) by wdconinc, asked 5 days ago in Q&A. **Answered**. 1 response.
- What are loc.first and loc.second in outputTrackParameters?** (2 upvotes) by wdconinc, asked 5 days ago in Q&A. **Unanswered**. 1 response.
- What is the difference between e.g. EcalEndcapNClusters and EcalEndcapPClusters?** (2 upvotes) by wdconinc, asked 5 days ago in Q&A. **Answered**. 1 response.
- What is the difference between GeneratedParticles and ReconstructedParticles?** (2 upvotes) by wdconinc, asked 5 days ago in Q&A. **Answered**. 1 response.

The right sidebar shows the 'Most helpful' section for the user 'wdconinc', with 3 helpful discussions, including 'Community guidelines' and 'github.com/orgs/eic/discussions'.

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/eeee](#) [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)



The screenshot shows the GitHub interface for the repository 'ecce'. At the top, there is a header with the repository name 'ecce' and 'Project ID: 521'. To the right of the header are icons for notifications, stars (0), and forks (0). Below the header, there are statistics: '244 Commits', '6 Branches', '0 Tags', and '77.4 GB Project Storage'. A note indicates 'Primary development at github.com/eic/ecce !!'. A progress bar is visible below the statistics. The main content area shows a merge request for the 'main' branch, titled 'Merge branch 'github-workflow-disable-lcg99' into 'main'', authored by 'Wouter Deconinck' 2 weeks ago. The merge request ID is '1bf6192e'.

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...)

Triggered via push 3 days ago	Status	Total duration	Artifacts
veprbl pushed 2325756 main	Success	54m 19s	18

Triggered via push 16 days ago	Status	Total duration	Artifacts
wdconinc pushed 4ed9578 master	Success	1h 28m 40s	5

linux-eic-shell.yml
on: push

```
graph LR; A[xmlint 70s] --> B[build-test 3m 30s]; B --> C[generate-grim-file 8m 10s]; C --> D[down-view (view30) 1h 13m]; C --> E[convert-to-tgeo 2m 46s]; E --> F[dump-constants 1m 50s]; F --> G[check-overlap-tgeo 4m 6s]; G --> H[check-overlap-geant4 14m 14s];
```

The diagram shows a workflow with a main sequence of jobs: xmlint (70s), build-test (3m 30s), generate-grim-file (8m 10s), and down-view (view30) (1h 13m). A sub-sequence of jobs branches off from generate-grim-file: convert-to-tgeo (2m 46s), dump-constants (1m 50s), check-overlap-tgeo (4m 6s), and check-overlap-geant4 (14m 14s). All jobs are marked as successful with green checkmarks.

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

jug_xl

Install from the command line: [Learn more](#)

```
$ docker pull ghcr.io/eic/jug_xl:4.0-deathvalley-stable
```

Recent tagged image versions

4.0-deathvalley-stable	↓ 0
Published 40 minutes ago · Digest ...	
4.0-deathvalley-1.5T-stable	↓ 0
Published 41 minutes ago · Digest ...	
4.0-canyonlands-stable	↓ 0
Published 42 minutes ago · Digest ...	
testing	↓ 0
Published 43 minutes ago · Digest ...	
nightly-2022-08-09 nightly	↓ 0
Published about 1 hour ago · Digest ...	

[View and manage all versions](#)

Last published

40 minutes ago

Total downloads

0

[Package settings](#)

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.