ACTS-Related Developments

Wouter Deconinck, University of Manitoba

Overview

ACTS versions:

- Upstream: 45 (2026-01-23) – 46 (2026-03-23) – 46.8.1 (2026-06-12)
- EICrecon: min 39.2.0 (since 25.08) – current 45.3.0 (since 26.06)
  - Newer versions not generally tested beyond ACTS continuous integration
- Developments for (requested by) EIC into ACTS are only usable after typically 6 months when we upgrade to them in nightly and releases

EICrecon support

Ongoing PRs

ACTS-EICrecon Compatibility Levels

- Forward compatibility: how well does our current EICrecon support exploring and testing with newer ACTS versions than those officially supported?
  - Can users easily install newer ACTS versions in the container?
  - Can current EICrecon compile/run/test against newer ACTS versions?
- Backward compatibility: how well does our current EICrecon support ACTS versions that were used for past campaigns (e.g. re-running in comparison)?
  - Can users easily install older ACTS versions in the container?
  - Can current EICrecon compile/run/test against older ACTS versions?
- Compatibility window: versions of ACTS that are tested in CI workflows
  - Current ACTS version in eic-shell is always compiled/run/tested against.
  - Up to (typically) 6 month old containers are required to still compile as well (but not actively tested for running/testing anymore).

Installing Different ACTS Versions in `eic-shell`

- Container is often read-only (and must be usable when it is read-only)
- Packages installed in `/opt/software`, combined in environments (e.g. `eic_xl`, `eic_cuda`), which have views exposed in `/opt/local`
- Package management with `spack`, targeted at HPC and HTC clusters
- Since modifying `/opt/local` not possible, customization inside `eic-shell` requires an extra layer in the user home directory.

```
$ spack env activate --create acts
$ spack install --add acts@46.8.1
```

This will (ideally) install `acts`, the environment, and the view under `~/spack`.

Installing Different ACTS Versions in `eic-shell`

Singularity (Linux, WSL)

After doing this a few times, since `~/spack` is always used, you may get a pretty large installation there.

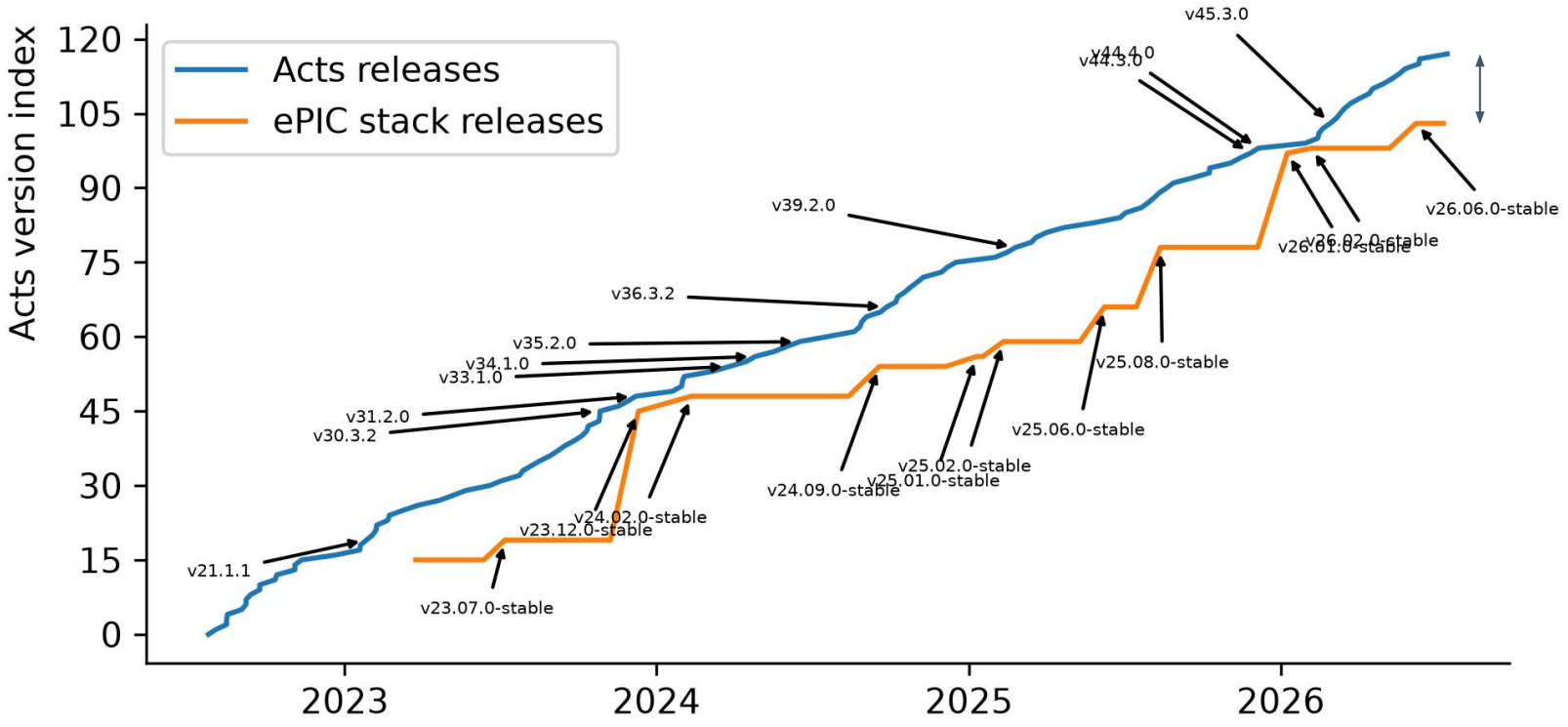
This is hard to test in continuous integration (which doesn't run singularity), so it's possibly a bit fragile, but `~Helpdesk` is your friend (or filled with them).

Docker (i.e. on macOS)

Usually the `~/spack` directory will point to `/root/spack` and is not persisted.

This workflow is actively tested in continuous integration (which use Docker under linux), so it is supposed to be quite reliable.

Current Compatibility Window



Require Acts 44+ (EICrecon#2708)

Simple PR to remove the v39 backwards compatibility target since it is becoming untested.

New Features in ACTS

v39 – v41

- v39.2.0 / v39.2.1: TrackingGeometry visitor improvements; DD4hep fix to ignore non-TRACKER subdetectors (relevant for geometry conversion robustness; low migration cost)
- v40.0.0 / v40.1.0: Multiple breaking geometry/material refactors (ProtoBinning→ProtoAxis, accumulated material changes, GeometryIdentifier mutator deprecations); CKF recoverable error improvements (medium-high: geometry/material behavior changes affect track finding and material mapping quality)
- v41.0.0 / v41.1.0: TrackingGeometry generation exposure, geometry/EDM4hep refactors, seeding and CMake fixes (medium: useful for diagnostics/performance and integration stability)

New Features in ACTS

v42 – v44

- v42.0.0 / v42.1.x: Gen3 geometry/navigation advances; DD4hep contextual detector/alignment infra; seeding cleanup and unit refactors; breaking RiddersPropagator options; plugin churn (Legacy/CUDA removal, TGeo→Root rename) (high for geometry/nav/seeding behavior; moderate build impact if plugin naming assumptions exist)
- v43.0.x: CMake targets change; further Surface/Seeding breakages and DD4hep/SurfaceArray fixes (high build-system impact; already handled in EICrecon CMake)
- v44.0.x: `Seeding2` maturation (orthogonal seeder port), KalmanFitter feature extensions, breaking `ActsPlugins` namespace and several surface/material API changes (high: plugin namespace and tracking API drift; mostly guarded in EICrecon)

New Features in ACTS

v45 – v46

- v45.0.0+: Large deprecated-symbol removals; immutable GeometryIdentifier model enforced; Gen1 geometry pieces removed; major Kalman/material API changes (very high for downstreams with older code paths; EICrecon has partial guards but this is the key concern)
- v46.0.0 → v46.8.1: More breaking cleanup around material mapping/navigators; deprecation of templated seeding; ongoing seeding/material fixes and performance updates (medium-high: potential semantic shifts in tracking/material behavior even when compile succeeds)

Support free surfaces in Acts 45+ (EICrecon#2502)

In Acts v45 surfaces can exist without connection to a detector element (e.g. effective surfaces inside calorimeters).

- May allows us better to define these effective surfaces in DD4hep geometry, instead of in the configuration of the track projection algorithm
- Associating the effective surfaces is essentially a geometry construction, so it should be stored there.

Adjust Tracking Time Unit (ElCrecon#2551)

Problem:

- `edm4hep::units::ns` = `dd4hep::ns` ≠ `Acts::UnitConstants::ns`
- Reason: ACTS aims to avoid unnecessary conversions and works in natural units where *c* = 1, and so time is 1 ns = 299.792458 mm = 299.792458 (since mm = 1). DD4hep does not use natural units.
- Issues in:
  - `timeError = 0.1 * Acts::UnitConstants::mm;` // assumes ns == mm
  - `cov(5, 5) = m_cfg.timeError / Acts::UnitConstants::ns;` // cov square
  - `time{static_cast<float>(parameter(Acts::eBoundTime))};` // missing conversion

This PR addresses that but also (unintentionally) introduces numerical changes that we want to avoid, so needs a few more steps before ready:

- `timeError = 1 * Acts::UnitConstants::ns;`

Support ACTS Gen1/Gen3 Geo (ElCrecon#2227)

ACTS geometry models generations:

1. What we are currently using: volumes and layer
2. (the lost generation; it shall not be mentioned or used)
3. Where the future lies: defined by blueprints
    - “Portal:” prescription for how a track goes from one volume to another at a boundary.
    - “Blueprint:” connection of sets of volumes through portals.
    - Removes O(n) lookup when moving to another volume; replaces it with O(1) operation.

Implementation:

- Aiming for backward-compatibility during validation phase: support for both Gen1 and Gen3 with user-settable flag.
- Support for automatic building of blueprints from DD4hep was added in February 2026 and is only as of recently supported in our ACTS version.

Upgrade to Acts::Seeding2 (EICrecon#2524)

Transition in ACTS track seeding interfaces:

- `Seeding2` introduced in v39; `Seeding` deprecated in v45, removed in v46
- We have to update to `Seeding2` for our next ACTS upgrade, and to be able to use our integration with ACTS continuous integration (broken since a few weeks now)

Implementation:

- Bimodal `TrackSeeding` algorithm that can pick either `Seeding` or `Seeding2` as the underlying algorithm to use.
- Agrees bit-for-bit for the backwards compatible path.
- Differences for forwards compatible path (as somewhat expected).

Ready for expert evaluation to determine if impact is acceptable.

Alignment Workflow Support (EICrecon#2603)

ACTS with Millepede-II

Goal: support alignment workflows to explore anticipated commissioning procedures and use as a test case for automated calibration workflows

Two components:

- `EDM4eic` datamodel (EDM4eic#161): adds `AlignmentDerivativeSet` with per-surface alignment derivative data for one track (we already have support for storing full surface information in the PODIO output data)
- `EICrecon` algorithm (EICrecon#2603): export measurements into Mille format for offline alignment solving by Millepede-II.

Development is aligned with work within ACTS to increase support for alignment workflows.

ActsPodioEdm for Intermediates (EICrecon#2338)

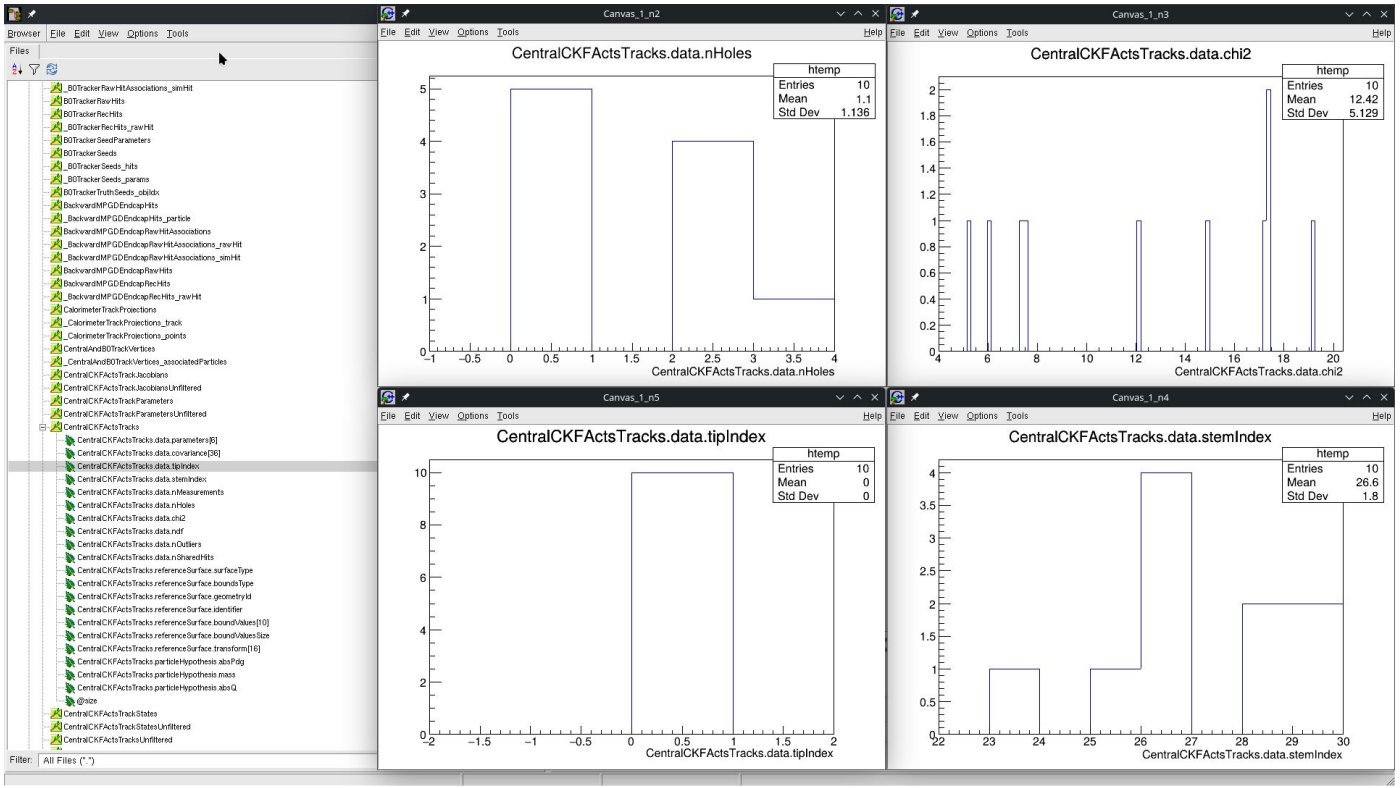
Current model:

- `edm4hep::TrackerHit` – `edm4hep::Measurement2D`
- `CKFTracking` to ACTS trajectories and tracks stored in `Acts::VectorMultiTrajectory`/`VectorTrackContainer`
- `ActsToTracks` converts this back into `edm4hep::Track` etc.

Proposed model:

- `CKFTracking` to ACTS trajectories and tracks stored in `ActsPodioEdm` which can also be written to output ROOT file, inspected as regular PODIO collection, and used in relations or links (associations).
- Access to more basic ACTS internal performance information (essentially full record)

ActsPodioEdm for Intermediates (EICrecon#2338)



Integration of BIC 1st Layer in Tracking

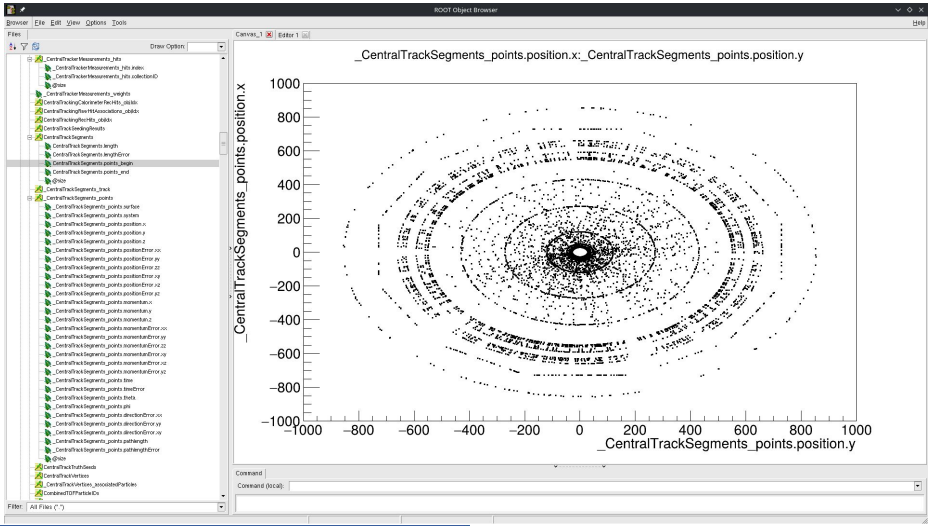
ePIC geometry support:

- epic#710 ✓ EcalBarrelImagingTracker support (support as tracking layer)
- epic#912 ✓ BIC 1st layer in separate envelope (performance)

EICrecon reconstruction support:

- EICrecon#1570: inject BIC hits into track reconstruction
  - Some bit rot, but ready for test/review
- EICrecon#1921 ✓propagate from last hit to calorimeters either forward or backward, as appropriate

Status: EICrecon#1570 is only open item



Possible Other Developments of Interest

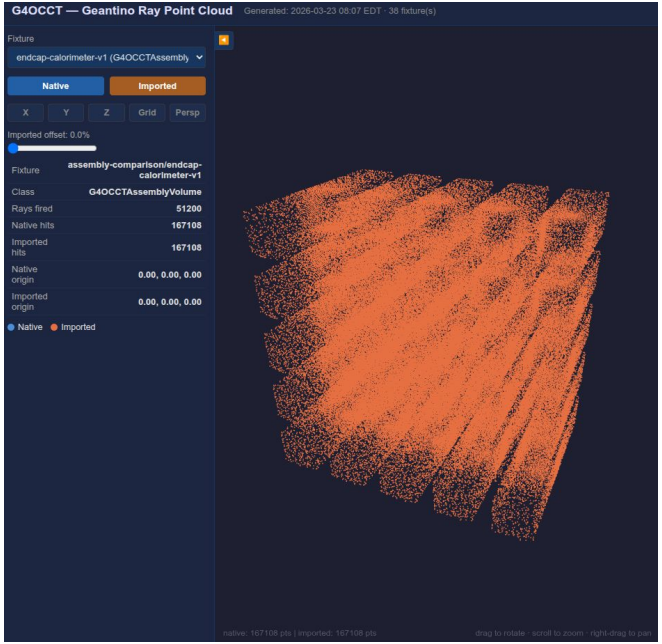
G4OCCT as library for importing CAD into DD4hep

Imports CAD geometry (without tessellations) into Geant4: `G4OCCTSolid`,...
Built on top of OpenCascade (OCCT).

- Navigation performance on CSG solids is worse than native Geant4 by factor ~15; on par with non-geometry processing in standard Geant4 calorimeter example B4

Possible uses in some of the more complicated geometry, like carbon fiber cooling structures.

But would only be ok if performance is still fast.



Closing

ACTS has formed an Experiment Liaison Group, with one Experiment Liaison requested for ePIC; S&C has request for me to be that person.

ACTS developer meetings open to all on Tuesday 5pm Zürich, https://indico.cern.ch/category/7968/