ePIC Software & Computing Review

October 12, 2023

SURA Headquarters, Washington DC

ePIC

SOFTWARE STACK

Status and Plans

Sylvester Joosten

Argonne National Laboratory

On behalf of the ePIC Collaboration

Introduction

Charge Items

1. At this stage, approximately ten years prior to data collection, is there a comprehensive and cost-effective long-term plan for the software and computing of the experiment?
2. Are the plans for integrating international partners' contributions adequate at this stage of the project?
3. Are the plans for software and computing integrated with the HEP/NP community developments, especially given data taking in ten years?
4. Are the resources for software and computing sufficient to deliver the detector conceptual and technical design reports?
5. Are the ECSJI plans to integrate into the software and computing plans of the experiment sufficient?

Introduction

2022 Software Infrastructure Review

Recommendations

1. Strengthen the connection between the EIC proto-collaboration software groups and the host lab computing and software support staff
2. Develop a long-term timeline for major software and computing infrastructure reviews/downselects/milestones
3. Clarify computing and software requirements on data handling, processing, and access
4. Ensure project needs associated with the CD process are being met by planned development and productions

Directly addressed through the ECSJI

We developed such a timeline, see this talk and the ePIC Computing Model talk

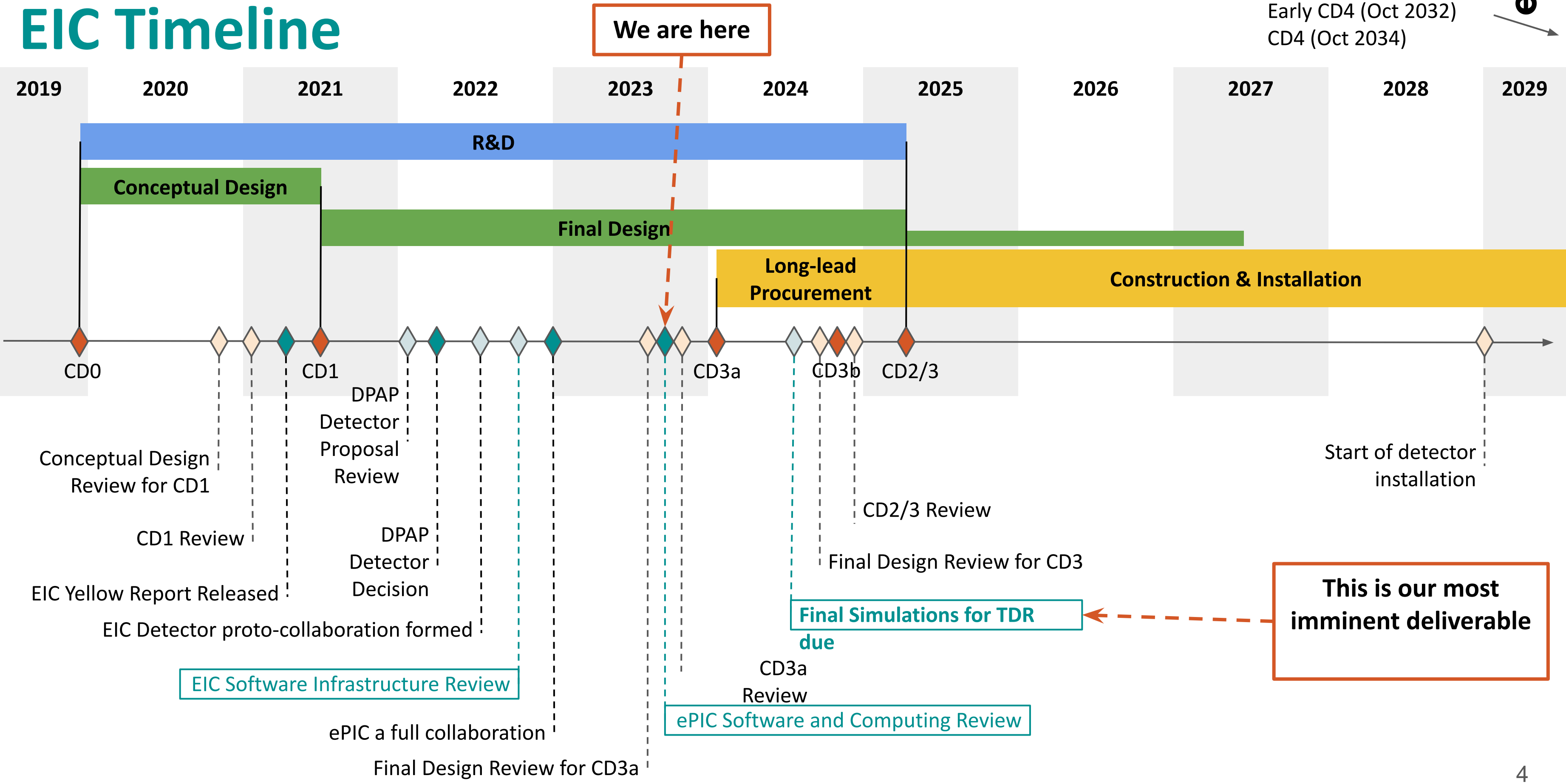
Being addressed through our working group structure, see also Markus' talk

Addressed through monthly production campaigns and dedicated topical task forces (see this talk)

Introduction

EIC Timeline

Not shown:
Early CD4 (Oct 2032)
CD4 (Oct 2034)

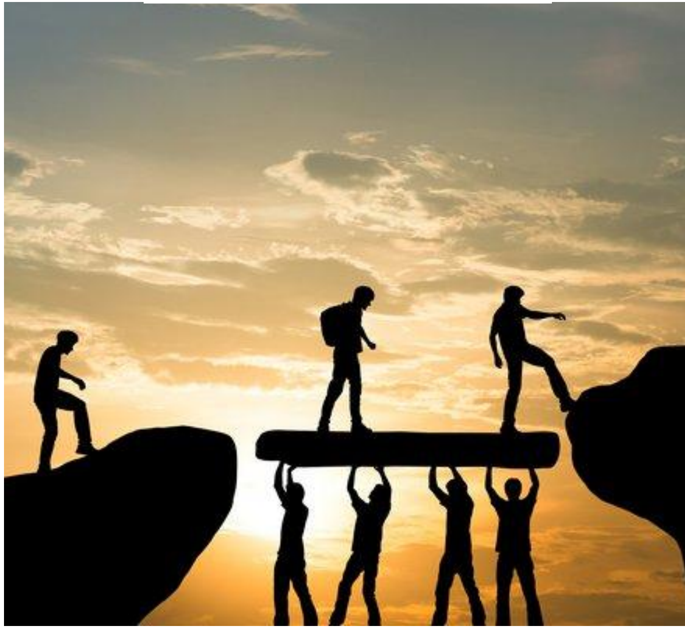


We are here
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
R&D
Conceptual Design
Final Design
Long-lead Procurement
Construction & Installation
CD0
CD1
CD3a
CD3b
CD2/3
DPAP Detector Proposal Review
Conceptual Design Review for CD1
CD1 Review
EIC Yellow Report Released
DPAP Detector Decision
EIC Detector proto-collaboration formed
EIC Software Infrastructure Review
ePIC a full collaboration
Final Design Review for CD3a
ePIC Software and Computing Review
CD3a Review
Final Simulations for TDR due
Final Design Review for CD3
CD2/3 Review
Start of detector installation
This is our most imminent deliverable

Introduction

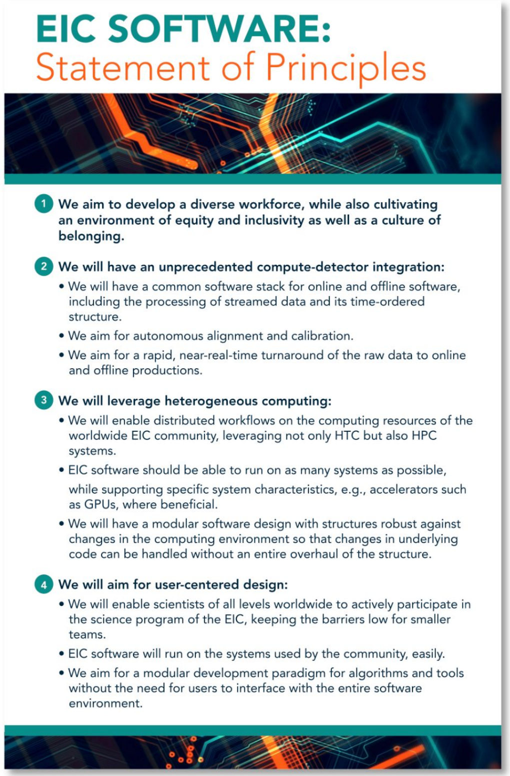
Our Philosophy

- We focus on modern scientific software & computing practices to ensure the long-term success of the EIC scientific program throughout all CD milestones
  - Strong emphasis on modular, orthogonal tools
  - Integration with HTC/HPC, CI workflows, and enable use of standard data science toolkits
- We leverage cutting edge sustainable community software where appropriate, avoiding the “not invented here” syndrome
  - Can build our software on top of a mature, well-supported, and actively developed software stack by using modern community tools, e.g. from CERN, the HPC community, and the data science community
  - Actively collaborate with external software projects, while externalizing some support burden to external projects
- We embrace these practices today to avoid starting our journey to EIC with technical debt.
- **We are writing software for the future, not the lowest common denominator of the past!**



Introduction

EIC Software: Statement of Principles



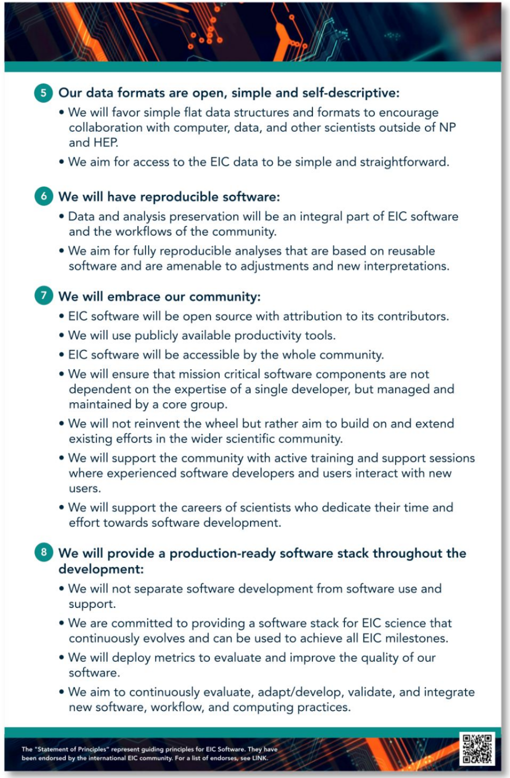
EIC SOFTWARE:
Statement of Principles

1. **We aim to develop a diverse workforce, while also cultivating an environment of equity and inclusivity as well as a culture of belonging.**

2. **We will have an unprecedented compute-detector integration:**
   - We will have a common software stack for online and offline software, including the processing of streamed data and its time-ordered structure.
   - We aim for autonomous alignment and calibration.
   - We aim for a rapid, near-real-time turnaround of the raw data to online and offline productions.

3. **We will leverage heterogeneous computing:**
   - We will enable distributed workflows on the computing resources of the worldwide EIC community, leveraging not only HTC but also HPC systems.
   - EIC software should be able to run on as many systems as possible, while supporting specific system characteristics, e.g., accelerators such as GPUs, where beneficial.
   - We will have a modular software design with structures robust against changes in the computing environment so that changes in underlying code can be handled without an entire overhaul of the structure.

4. **We will aim for user-centered design:**
   - We will enable scientists of all levels worldwide to actively participate in the science program of the EIC, keeping the barriers low for smaller teams.
   - EIC software will run on the systems used by the community, easily.
   - We aim for a modular development paradigm for algorithms and tools without the need for users to interface with the entire software environment.



5. **Our data formats are open, simple and self-descriptive:**
   - We will favor simple flat data structures and formats to encourage collaboration with computer, data, and other scientists outside of NP and HEP.
   - We aim for access to the EIC data to be simple and straightforward.

6. **We will have reproducible software:**
   - Data and analysis preservation will be an integral part of EIC software and the workflows of the community.
   - We aim for fully reproducible analyses that are based on reusable software and are amenable to adjustments and new interpretations.

7. **We will embrace our community:**
   - EIC software will be open source with attribution to its contributors.
   - We will use publicly available productivity tools.
   - EIC software will be accessible by the whole community.
   - We will ensure that mission critical software components are not dependent on the expertise of a single developer, but managed and maintained by a core group.
   - We will not reinvent the wheel but rather aim to build on and extend existing efforts in the wider scientific community.
   - We will support the community with active training and support sessions where experienced software developers and users interact with new users.
   - We will support the careers of scientists who dedicate their time and effort towards software development.

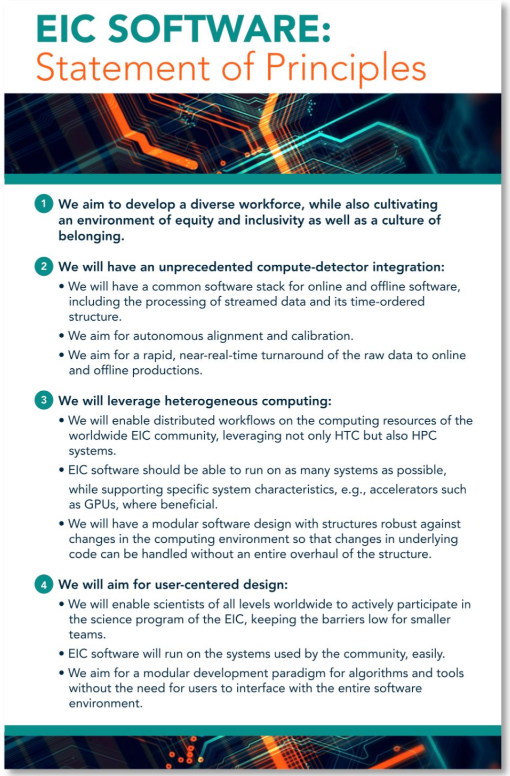
8. **We will provide a production-ready software stack throughout the development:**
   - We will not separate software development from software use and support.
   - We are committed to providing a software stack for EIC science that continuously evolves and can be used to achieve all EIC milestones.
   - We will deploy metrics to evaluate and improve the quality of our software.
   - We aim to continuously evaluate, adapt/develop, validate, and integrate new software, workflow, and computing practices.

The "Statement of Principles" represent guiding principles for EIC Software. They have been endorsed by the international EIC community. For a list of endorses, see LINK.

- Community document that encodes our aspirations (technical and cultural) for software and computing at the EIC
- The foundation of the ePIC Software Stack
- Co-written and endorsed by a large group representing the international EIC community

Introduction

EIC Software: Statement of Principles



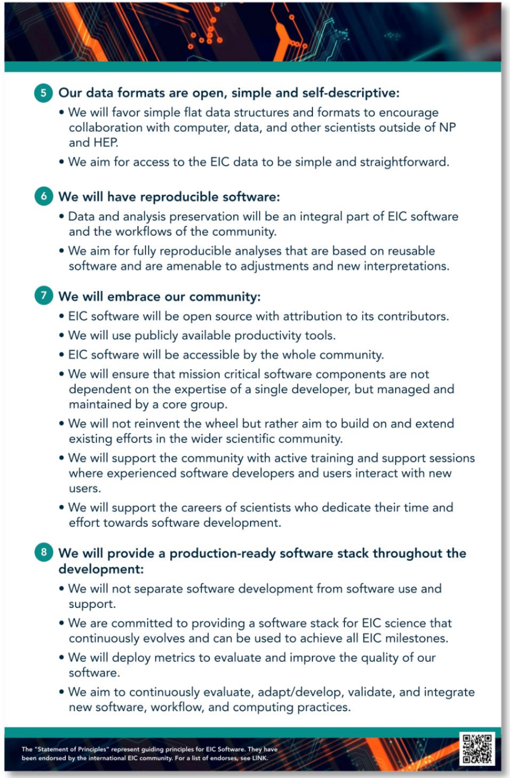
EIC SOFTWARE:
Statement of Principles

1 **We aim to develop a diverse workforce, while also cultivating an environment of equity and inclusivity as well as a culture of belonging.**

2 **We will have an unprecedented compute-detector integration:**
- We will have a common software stack for online and offline software, including the processing of streamed data and its time-ordered structure.
- We aim for autonomous alignment and calibration.
- We aim for a rapid, near-real-time turnaround of the raw data to online and offline productions.

3 **We will leverage heterogeneous computing:**
- We will enable distributed workflows on the computing resources of the worldwide EIC community, leveraging not only HTC but also HPC systems.
- EIC software should be able to run on as many systems as possible, while supporting specific system characteristics, e.g., accelerators such as GPUs, where beneficial.
- We will have a modular software design with structures robust against changes in the computing environment so that changes in underlying code can be handled without an entire overhaul of the structure.

4 **We will aim for user-centered design:**
- We will enable scientists of all levels worldwide to actively participate in the science program of the EIC, keeping the barriers low for smaller teams.
- EIC software will run on the systems used by the community, easily.
- We aim for a modular development paradigm for algorithms and tools without the need for users to interface with the entire software environment.



5 **Our data formats are open, simple and self-descriptive:**
- We will favor simple flat data structures and formats to encourage collaboration with computer, data, and other scientists outside of NP and HEP.
- We aim for access to the EIC data to be simple and straightforward.

6 **We will have reproducible software:**
- Data and analysis preservation will be an integral part of EIC software and the workflows of the community.
- We aim for fully reproducible analyses that are based on reusable software and are amenable to adjustments and new interpretations.

7 **We will embrace our community:**
- EIC software will be open source with attribution to its contributors.
- We will use publicly available productivity tools.
- EIC software will be accessible by the whole community.
- We will ensure that mission critical software components are not dependent on the expertise of a single developer, but managed and maintained by a core group.
- We will not reinvent the wheel but rather aim to build on and extend existing efforts in the wider scientific community.
- We will support the community with active training and support sessions where experienced software developers and users interact with new users.
- We will support the careers of scientists who dedicate their time and effort towards software development.

8 **We will provide a production-ready software stack throughout the development:**
- We will not separate software development from software use and support.
- We are committed to providing a software stack for EIC science that continuously evolves and can be used to achieve all EIC milestones.
- We will deploy metrics to evaluate and improve the quality of our software.
- We aim to continuously evaluate, adapt/develop, validate, and integrate new software, workflow, and computing practices.

The "Statement of Principles" represent guiding principles for EIC Software. They have been endorsed by the international EIC community. For a list of endorses, see LINK.

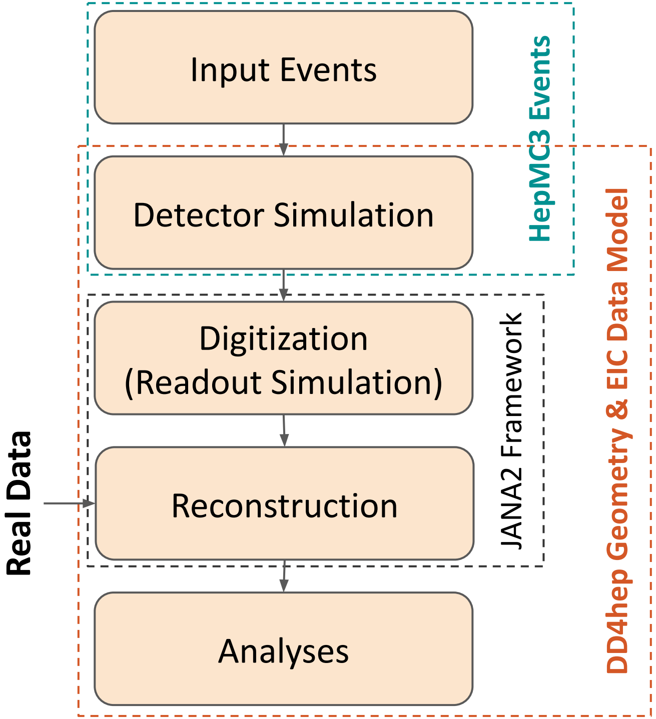
EIC Software is:

1. Diverse
2. Integrative
3. Heterogeneous
4. User-centered
5. Accessible
6. Reproducible
7. Collaborative
8. Agile

Software Overview

A modular simulation, reconstruction, and analysis toolkit

The ePIC Software Stack



Input events from MC event generators or particle guns, with optional physics background merging

GEANT4 simulations driven by DD4hep, output data in the EIC Data Model (EDM4hep + EDM4eic, described in Podio)

Algorithms to transform the GEANT4 hits to mimic real detector readout, including background stacking, “pileup”, DAQ frames

Realistic reconstruction algorithms starting from raw detector output (from digitization or real data)

User analyses in plain C++/ROOT or Python/uproot, facilitated by using a flat data model

Continuous integration for detector and physics benchmarks and monthly production campaigns ensure a production-ready software stack at any time

Our data model is the glue for the software stack

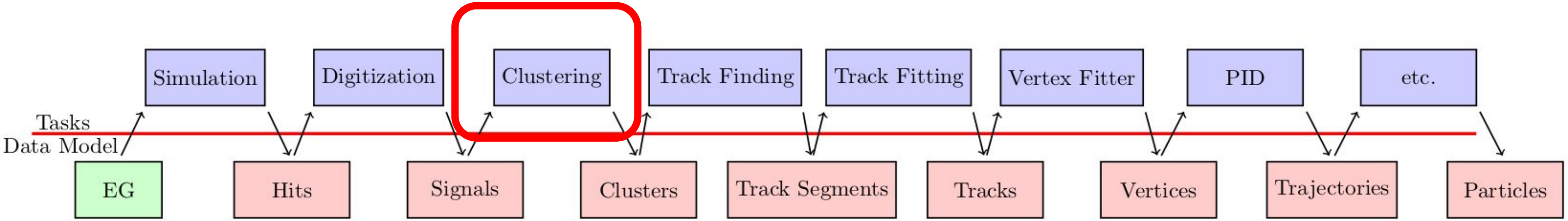
What is a Data Model?

- **Standardized data structures that we collectively agree to use to pass information between simulation, reconstruction, and analysis algorithms**
  - Example: The information we talk about when we say ‘a hit in a tracking detector,’ such as channel number, energy deposition, time, position, etc…
  - The data model is the “protocol” that the components in our software stack use to talk to each other
- This does **not** include: Decisions about the input/output file format, memory layout, or the physical data storage medium
  - Example: Our choice of data model does not require storage in ROOT files (but can be written to ROOT files, HDF5 files, and many others), does not require C++ (or Python), does not require row-oriented memory layouts (but allows for GPU processing), etc…
- **We aim for flexibility through our choice of data model.**

Standardized interfaces between components ensure modularity

Data-Driven API Design and the EIC Data Model

- Use of **standard interfaces** between individual simulation, reconstruction, and analysis tasks **creates modularity** that enables **easy exchange of components**
- Example: multiple clustering algorithms can be swapped out, as long as they adhere to the data model interfaces
- We standardized on EDM4eic (an extended version of EDM4hep from the Key4HEP project) for our entire software stack, and HepMC3 for Monte Carlo input.
- This modularity extends beyond the EIC community, since many data structures are common across NP and HEP experiments worldwide

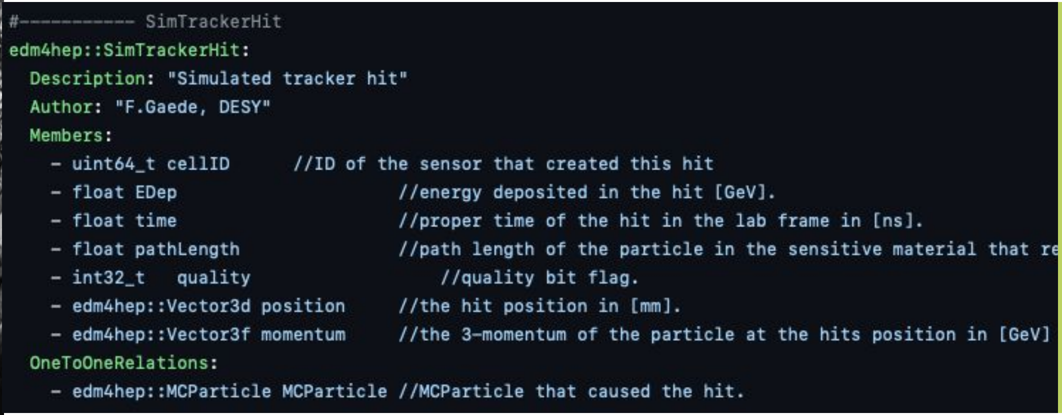


Open source community data model toolkit for NP & HEP

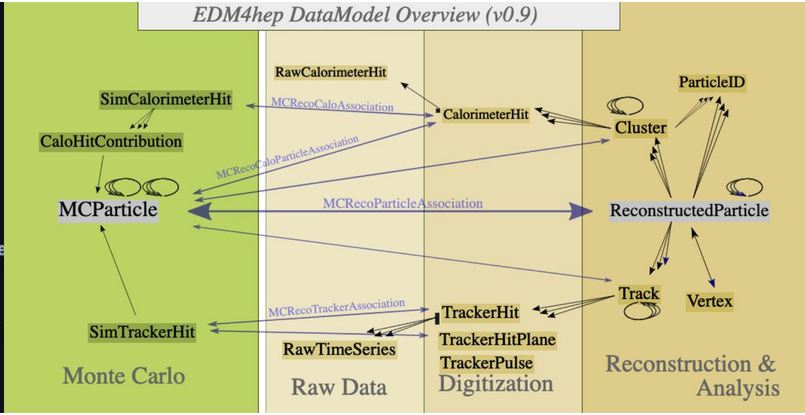
Podio, EDM4hep, and EDM4eic

- Podio is a community tool to define data models in a human-readable format
- EDM4hep implements a standard data model for HEP using Podio
- EDM4eic is a set of EIC-specific extensions to EDM4hep
- All components of the data model are **open source** and supported by **multiple institutions and collaborations** with goals aligned with ours
- **ePIC is closely aligned and actively involved with Key4HEP**

Key4HEP stack



```
#----------- SimTrackerHit
edm4hep::SimTrackerHit:
  Description: "Simulated tracker hit"
  Author: "F.Gaede, DESY"
  Members:
    - uint64_t cellID      //ID of the sensor that created this hit
    - float EDep                     //energy deposited in the hit [GeV].
    - float time                     //proper time of the hit in the lab frame in [ns].
    - float pathLength               //path length of the particle in the sensitive material that re
    - int32_t   quality                  //quality bit flag.
    - edm4hep::Vector3d position     //the hit position in [mm].
    - edm4hep::Vector3f momentum     //the 3-momentum of the particle at the hits position in [GeV]
  OneToOneRelations:
    - edm4hep::MCParticle MCParticle //MCParticle that caused the hit.
```



EDM4hep DataModel Overview (v0.9)

Single source of geometry for simulation and reconstruction

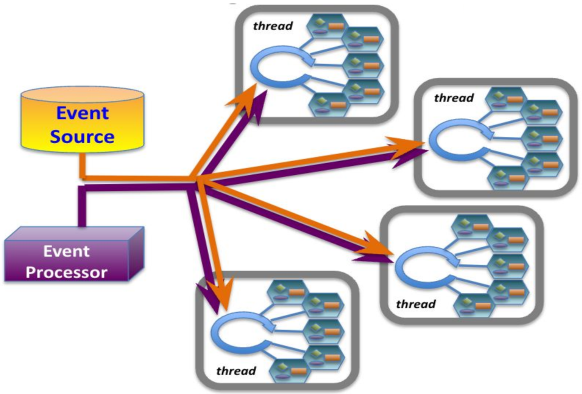
Geometry Description and Detector Implementation

- Simulation and reconstruction both need a single source of geometry
- To ensure modularity this should be provided by an orthogonal toolkit (e.g. not directly connected to the event processing framework)
- DD4hep provides a complete solution for a full detector description (geometry, materials, visualization, readout, alignment, …)
  - Parametrized geometries are a powerful tool for detector optimization
  - A full implementation of the ePIC detector in DD4hep has been used in production for well over a year
  - Our experience has been very positive, in terms of new user onboarding, using DD4hep to drive GEANT4 simulations , accessing DD4hep geometry information in reconstruction algorithms, and collaborating with the DD4hep developers
- EIC-specific npsim library configures EIC physics, optical photon settings, …
- **ePIC is now a major contributor to the DD4hep project** (bugfixes and new features)

Event processing framework for reconstruction (and digitization)

Reconstruction Framework

- We selected JANA2 as the reconstruction framework based on a carefully formed set of requirements reviewed by the EIC software community
  - Natively multithreaded from the start
  - Supports streaming DAQ and heterogeneous hardware
  - Active development and support (1 dedicated FTE from Jefferson Lab to support ePIC)
  - Developed by Jefferson Lab, one of the EIC host labs
  - Over 20 years of production experience between JANA and JANA2
- We implemented EICrecon with the tooling and algorithms needed for ePIC
  - Podio frames support
  - DD4hep geometry support
  - Quasi-framework independent algorithms
  - Algorithm configuration and wiring
- Current mid-term framework developments:
  - Fully framework independent (modular) algorithms
  - Flexible external algorithm wiring and configurability
  - Metadata and conditions handling
  - Better integration of ML in the reconstruction



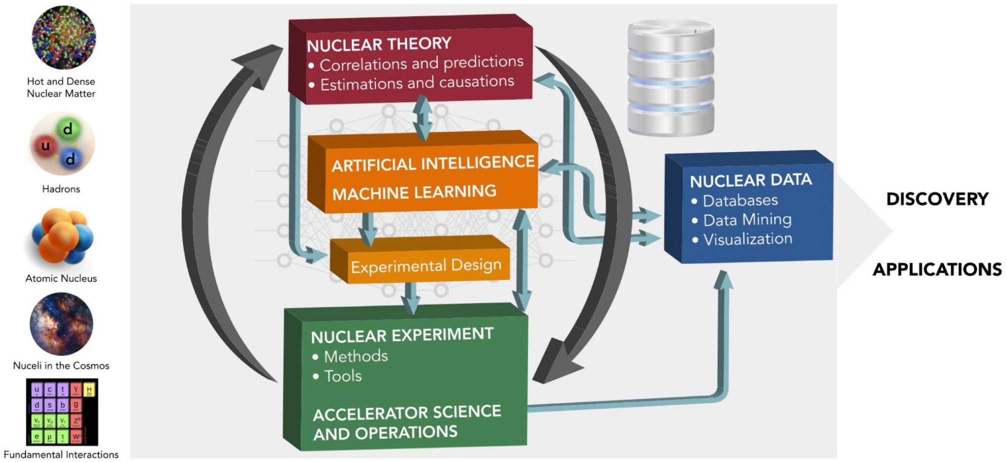
Example: ACTS for Tracking Algorithms

- ACTS is the main toolkit to express our tracking algorithms, we have used ACTS for all our ePIC simulation campaigns
- ACTS integrates almost seamlessly with our DD4hep geometry
- ACTS developers attend out weekly Tracking Reconstruction meetings, which is invaluable
- Solving EIC-unique problems together with the ACTS team
- Many EIC-related commits are now part of the main ACTS codebase
- **Highly positive experience:** the ACTS team has treated us as first-rate “clients” of their project, and we are contributing back code

Towards AI-accelerated simulation productions

Artificial Intelligence at ePIC

- AI already has an important presence in ePIC, with many usages in the prototyping stage, e.g., for detector surrogates and reconstruction methods
- To explore and develop the full potential of AI for ePIC, we will move from prototyping to production in the next year, adding powerful AI approaches into our workflows



- First step: integrate existing AI reconstruction routines, e.g. for calorimeter reconstruction, into the reconstruction framework
    - Currently using TensorFlow and Torch for learning and standalone inference
    - Determine and evaluate framework for inference, e.g. ONNX
    - Aim to have first production-ready AI reconstruction algorithms by December 2023

Our next priorities towards to TDR

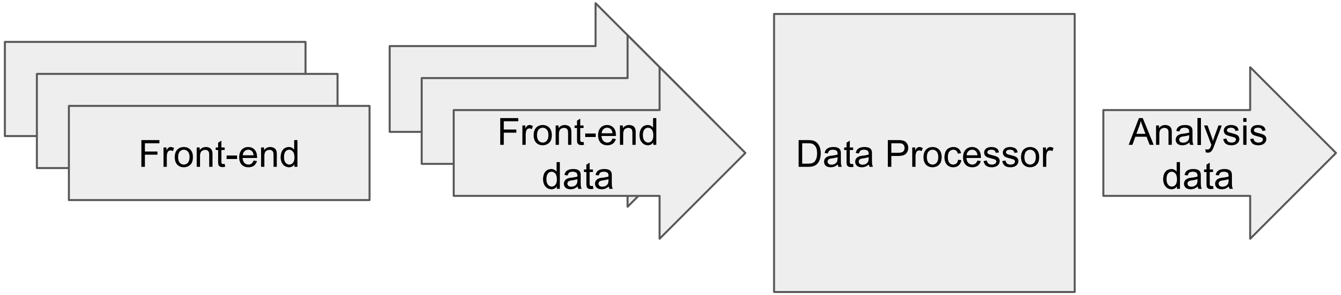
Developing the Reconstruction

- Integrate streaming readout frames and the transition of frames → events in the reconstruction flow
  - Necessary to evaluate tracking detectors with large integration time
  - Strictly separate digitization from reconstruction, and implement frame-builder after digitization
  - Develop time-based strategy for implementing frame-based reconstruction (a must for the TDR)
- Prioritize direct AI/ML integration by the end of this year
- Address the need for metadata and a conditions database
- Better algorithms:
  - Deal with realistic DAQ readout for *all* detectors
  - Vertexing and seeding, particle identification, event-level reconstruction, jet reconstruction, …
- Collaboration and future development: promote the opportunity for team members to take ownership and responsibility for different parts of the software ecosystem

See Markus’ talk

Why we care about streaming readout for the EIC

Streaming readout for the EIC



Front-end

Front-end data

Data Processor

Analysis data

Theory

- Integration of DAQ, analysis, and theory will optimize the physics reach for the EIC
- Aim for a research model with seamless processing from sensor through DAQ to analysis and theory.
- Need to consider this from the start to ensure we build the best detector that supports streaming readout and fast algorithms for alignment, calibration, and reconstruction in real time or near real time
- Streaming readout and AI work hand-in-hand to enable a rapid turnaround from data taking to physics analysis and publication.

The critical path towards a unified software approach for the EIC

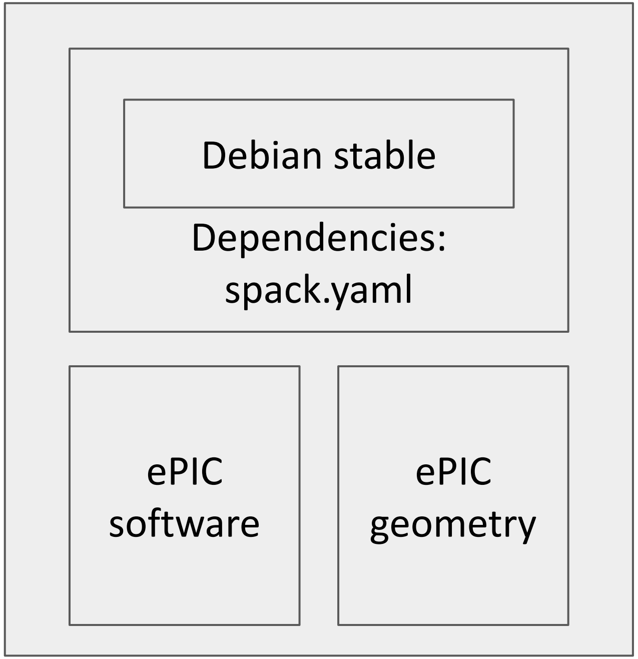
How did we get here?

1. Assessment on the software solutions (pro & con list), guided by the EIC Software Statement of Principles
2. Proposed conclusion and recommendation to Collaboration and Project at the Summer 2022 EICUG meeting
3. Software choice treated as any other technology choice: Independent Software Review in September 2022
4. All new development went into the official framework
5. **Rolled out new ePIC software stack in October 2022 (1 year ago!)**

Deployment and Workflows

Enabling a consistent working environment

Deployment with Containers



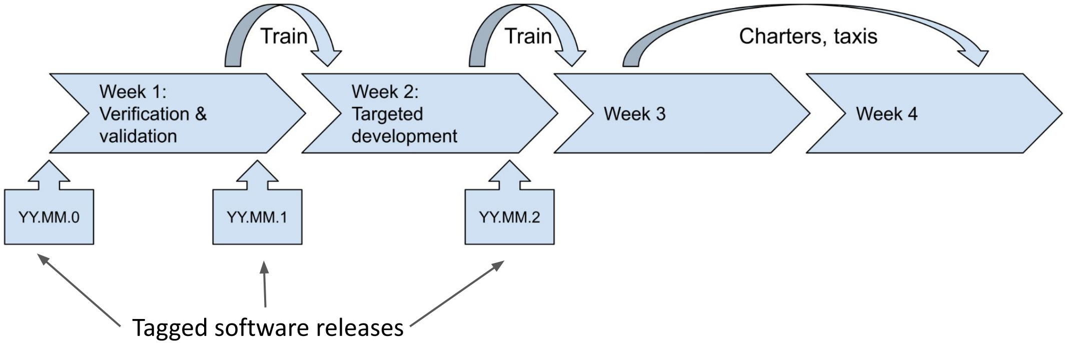
- Provide a single curated software build "eic-shell" for local development, CI, and production campaigns
  - Multiple architecture-specific versions of images where needed (e.g. amd64 and aarch64)
  - Build docker image and converted singularity image
- Different flavors:
  - **nightly:** all master branches, built every night
  - **stable/tagged:** release versions
  - **unstable**: temporary containers for Pull Requests
- Distribution:
  - DockerHub & Github Registry: all docker images
  - eicweb: Internal docker images, all singularity images
  - CVMFS: OSG ~6 hour synchronizations, to `/cvmfs/singularity.opensciencegrid.org`

Monthly simulation productions since May 2022

Simulation Campaign Strategy

Objectives

1. Achieve **continuous deployment** of the software used for detector and physics simulations
2. Ensure **regular updates** of simulation productions for detector and physics studies, and for geometry and algorithm development
3. Implement **timely validation and quality control** for simulation productions on datasets that require substantial time and resources



Train: Major central campaign at a fixed (monthly) schedule

Charter: Special interest runs for working groups

Taxi: Bespoke runs for individual users

Monthly resource needs ~ 200 core-years

Jobs run on the Open Science Grid, with output stored on BNL S3 and JLab XrootD

See "Simulation Production Strategy" prebrief material

Easy to get started locally… in only 1 line!

Local Software Deployment with eic-shell

Step 1: `curl -L get.epic-eic.org | bash`

Step 2: ???

Step 3: Profit

- Uses deployed images on /cvmfs when available, downloads singularity sifs otherwise
- Rolling out seamless container updates to end users
- At the same time basis of scalable computing on OSG: same containers are used everywhere.
- Note: In principle not even needed to look at data (flat format!)

Approach has worked robustly for multiple years now. Biggest challenge was making people believe it can really be this simple!

Main software repository for development

GitHub: EIC organization and managed runners

- Recommended standard interface for all source code projects in ePIC
- Modest computational resources:
  - 20 dual-core job slots for all projects under github.com/eic
- User management and workflow:
  - Everyone can get a GitHub account and every EIC user can get EIC organization membership
  - All new contributions happen through a pull request (PR)
  - Code can only be merged if it passess all CI checks and passes expert review
  - All PRs are squash-merged into the main branch to maintain a clear history
  - We strongly encourage users to make small incremental changes to most effectively develop software in a collaborative context
- Additional features:
  - GitHub actions model of development: easily shared across all of GitHub
  - GitHub pages for presentation (e.g. https://eic.github.io/epic/craterlake_views)

Ancillary software repostitory for CI

EICweb: ANL-located self-hosted GitLab instance

- eicweb is used for specific CI-intensive repositories (expert usage only)
  - Benchmarks (detector/reconstruction/physics)
  - Container building
- Larger computational resources:
  - Dedicated GitLab runner backend (AMD EPYC 7H12 - 256 threads, 512 GB, 2 GB/thread)
  - Access to a Kubernetes cluster
  - Jobs defined through gitlab’s yaml-based job specification language
  - Triggered by push, schedule, or webhook+token
- User management:
  - Requires account with access to ANL eicweb server
  - Aim to keep user base limited for easier management, focus on expert users who develop services that can be used through external calls
- Integration with GitHub:
  - CI pipelines integrated between GitHub and eicweb
  - Mirroring from eicweb to GitHub

Open, collaborative software development

Workflow Philosophy

Encourage Upstream Contributions

- Requirements of well-formed HepMC as input has resulted in real improvements to multiple MCEGs used by EIC community.
- Various upstream contributions to DD4hep, ACTS, Spack, uproot,...

Encourage Social Coding

- CI platform provides the incentive for developers to commit code frequently: achieving data management and analysis preservation goals.
- Pull request reviews to ensure higher quality code and build developer skills.

Enable Access Without Restrictions

- ePIC collaboration members include over 170 institutions worldwide
- Data ‘publicly’ available through BNL S3 and publicly available through JLab xrootd.
- Flat data structures (i.e. could be a csv), stored as ubiquitous ROOT trees without need for data structure libraries.
- Support for uproot using numpy library (awkward not needed).

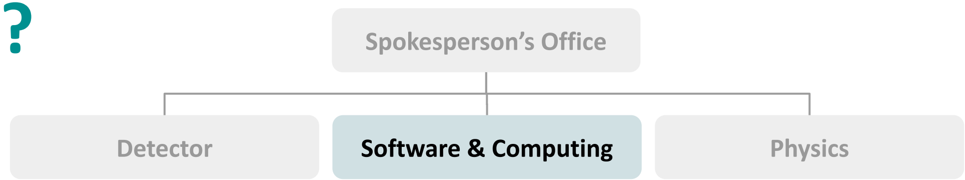
Approaches Under Evaluation

- Rucio for data management
- Reproducible analysis workflow tools

Collaboration and Planning

ePIC Software and Computing Organization

Who are we?



Guiding Principles:

- *Diversity, Equity, and Inclusion*
- *Statement of Software Principles*
- *Sustainability.*

Software and Computing Coordinator
Markus Diefenthaler (Jefferson Lab)

Cross-cutting Working Group:

- *Data and Analysis Preservation*

Deputy Coordinator (Operations)
Wouter Deconinck (U. Manitoba)

Operation Working Groups:

- Production
- User Learning
- Validation

Deputy Coordinator (Development)
Sylvester Joosten (ANL)

Development Working Groups:

- Physics and Detector Simulation
- Reconstruction Framework and Algorithms
- *Analysis Tools*

Deputy Coordinator (Infrastructure)
Torre Wenous (BNL)

Infrastructure Working Groups:

- Streaming Computing Model
- *Multi-Architecture Computing*
- *Distributed Computing*

+ cross-cutting task force efforts (across *all* ePIC Working Groups)

Software & Computing: Operations

Operations | Development | Infrastructure

Production Working Group

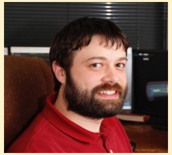
Charge:

- Responsible for the coordination and production of simulation campaigns based on priorities form the Technical and Analysis Coordinators.
- Develop automated production workflows that scale with the needs of the collaboration

Priorities for FY23-24

- Implement and document our Simulation Production Strategy, together with the Validation WG
- Survey current production resources and identify potential future resources
- Help determine the need to start the Distributed Computing (Infrastructure) WG

Production WG Conveners



Thomas Britton (Jefferson Lab)

Sakib Rahman (U. Manitoba)

Validation Working Group

Charge:

- Responsible for the validation of the simulations via a suite of detector and physics performance plots
- Develop autonomous checks and verification (CI)

Priorities for FY23-24

- Implement and document our Simulation Production Strategy, together with the Production WG
- Develop and maintain a collection of plots that showcase the performance of the ePIC detector, its physics reach, and enable comparison to a baseline or previous simulation campaigns.
- Drive the development of unit tests for the ePIC software, together with the Development WGs
- Design and implement a Software and Analysis Validation Policy containing enforceable testing requirements in line with our Statement of Principles

Validation WG Conveners

Torri Jeske (Jefferson Lab)

Dmitry Kalinkin (U. Kentucky)

User Learning Working Group

Charge:

- Responsible for support via documentation, help desk, and training
- Ensure software is discoverable and simulated data and metadata is findable

Priorities for FY23-24

- Develop a centralized documentation landing page for ePIC Software and Computing
- Establish a regular, predictable schedule of training that includes introductory and intermediate materials, incorporating relevant materials from the HSF Training WG
- Organize a “help desk” office hour

User Learning WG Conveners

Kolja Kauder (BNL)

Holly Szumila-Vance (Jefferson Lab)

Physics and Detector Simulation Working Group

Charge:

- Development of accurate MC simulations using a suite of physics and background generators, using a detector simulation based on GEANT4 and DD4hep

Priorities for FY23-24

- Support the detector design and integration with services.
- Support the needs for eA simulations
- Support the development of background modeling and its implementation in physics and detector simulations, together with the Background Task Force
- Embrace modularity and separate the simulation of the detector readout (digitization) from the reconstruction.
- Embrace streaming readout in simulation, using it as the default data format and enabling the capability to stream the continuous readout of the detecto

Physics and Detector Simulation WG Conveners

Kolja Kauder (BNL)

Chao Peng (ANL)

Reconstruction Framework and Algorithms Working Group

Charge:

- Development of a holistic and modular reconstruction for the integrated ePIC detector

Priorities for FY23-24

- Enforcing modularity for clear separation between the development of reconstruction algorithms and the development of the framework and its services
- Embrace algorithmic development that utilizes the holistic information from detector components or the entire detector
- Integrate far-forward and far-backward detectors in reconstruction
- Implement a web-based event display.

Reconstruction Framework & Algorithms WG Conveners

Derek Anderson (Iowa State)

Shujie Li (LBL)

Streaming Computing Model Working Group

Charge:

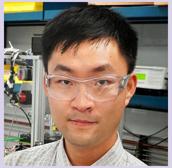
- Development of the computing model for the compute-detector integration using streaming readout, AI/ML, and multi-architecture computing (CPU, GPU, …) with a specific focus on the data flows after the FEE layer.

Priorities for FY23-24

- Establish a collaborative dialogue with the Electronics and DAQ WG
- Define the requirements and high-level design for a computing model that enables rapid processing for the data for physics analyses, while leveraging external compute resources, including those provided by international partners
- Coordinate activities on prototyping streaming computing systems, together with the Physics and Detector Simulation and Reconstruction Framework and Algorithms WGs
- Document a streaming computing model that can be redefined further with international partners

Streaming Computing Model WG Conveners

Marco Battaglieri (INFN Genova)



Jin Huang (BNL)

Jeff Landgraf (BNL)

Task Force Efforts

- Task Forces deal with high-priority tasks involving one or more ePIC Working Group
- Task Forces are temporary in nature, different from (Standing) Working Groups
- Current Task Force efforts:

Reconstruction WG + Physics WGs

- ○ **Electron Finder**: Develop an efficient and accurate algorithm for identifying electrons and identifying the scattered electron of the DIS process
- ○ **Vertexing and PID**: Enhance the vertexing capabilities and particle identification techniques to study heavy flavor physics
- ○ **Particle Flow**: Improve the jet reconstruction using particle flow information
- ○ **Low-Q^2**: Integrate the low-Q^2 tagger into the reconstruction framework for precise measurements of photo production and vector mesons

EIC Project + Simulations WG

- ○ **Backgrounds**: Introduce and evaluate realistic backgrounds in the ePIC GEANT4 simulations

Electron Finder
Lead: Daniel Brandenburg
(Ohio State)

Vertexing and PID
Lead: Shujie Li
(LBL)

Particle Flow
Lead: Derek Anderson
(Iowa State)

Low-Q^2
Lead: Simon Garnder
(University of Glasgow)

Backgrounds
Lead: Elke-Caroline Aschenauer (BNL)

Missing Working Groups?

Cross-cutting Working Group:
- *Data and Analysis Preservation*

Operation Working Groups:
- Production
- User Learning
- Validation

Development Working Groups:
- Physics and Detector Simulation
- Reconstruction Framework and Algorithms
- *Analysis Tools*

Infrastructure Working Groups:
- Streaming Computing Model
- *Multi-Architecture Computing*
- *Distributed Computing*

- Organizational structure setup to serve the collaboration in the long run
- EIC schedule is ambitious and ePIC is a young collaboration:
  - Our immediate priority is to prepare for the TDR and CD3
  - Prioritized Working Groups needed for our current priorities to leverage our limited workforce
- The following Working Groups and Task Forces will be phased in as needed:
  - **Data and Analysis Preservation (cross):** Post-TDR; currently directly managed by coordinators
  - **Analysis Tools (dev):** Already have one convener, likely to be started in the next year
  - **Multi-Architecture Computing (infra):** Post-TDR; build on initial work from the Reconstruction WG
  - **Distributed Computing (infra):** Post-TDR; build on initial work from the Production and Streaming Computing Model WGs
  - **Meta-data (TF):** Early effort managed by coordinators, will identify Task Force leads soon

Regular Workshops

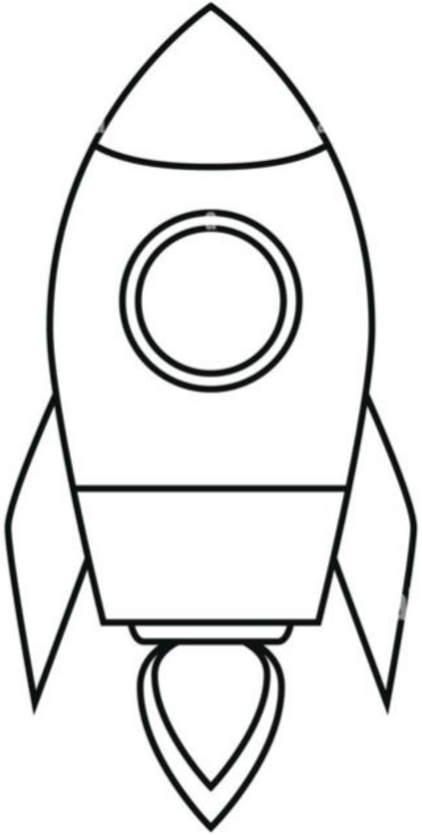
- We organize regular workshops/workfests to drive forward priority targets and provide an avenue for new collaboration members to actively engage.
- First workshop was in September at UIC, on Reconstruction Workflows, the Streaming Computing Model, User Learning, and Validation; up to 43 participants
- Next occasions will be a short pre-workshop to the collaboration meeting (January 2024, on TDR readiness), and a full workshop at CERN in Spring 2024 focussing on community software and collaboration

Wrapping it all up…

On our way to success!

Conclusions

- ePIC has organized a large-scale software and computing effort with clear short-term, mid-term, and long-term goals to ensure long term success of the EIC scientific program throughout all the milestones
- The ePIC software stack is a modern and modular toolkit built from HEP/NP community tools and components from HPC and Data Science; ePIC is an active member of the HEP/NP software and computing community
- ePIC employs modern software development practices and has a strong focus on community building and training to ensure a healthy developer and user community for the EIC experiment
- ePIC leverages monthly production campaigns, CI-driven benchmarks, timeline-based prioritization, and topical Task Force efforts to ensure timely completion of the simulation studies for the Technical Design Report.



This work is supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics, under contract DE-AC02-06CH11357.

U.S. DEPARTMENT OF ENERGY Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC. Argonne NATIONAL LABORATORY

ePIC Questions?