Data Management in Nuclear Physics: Cultural Change For Larger Collaborations

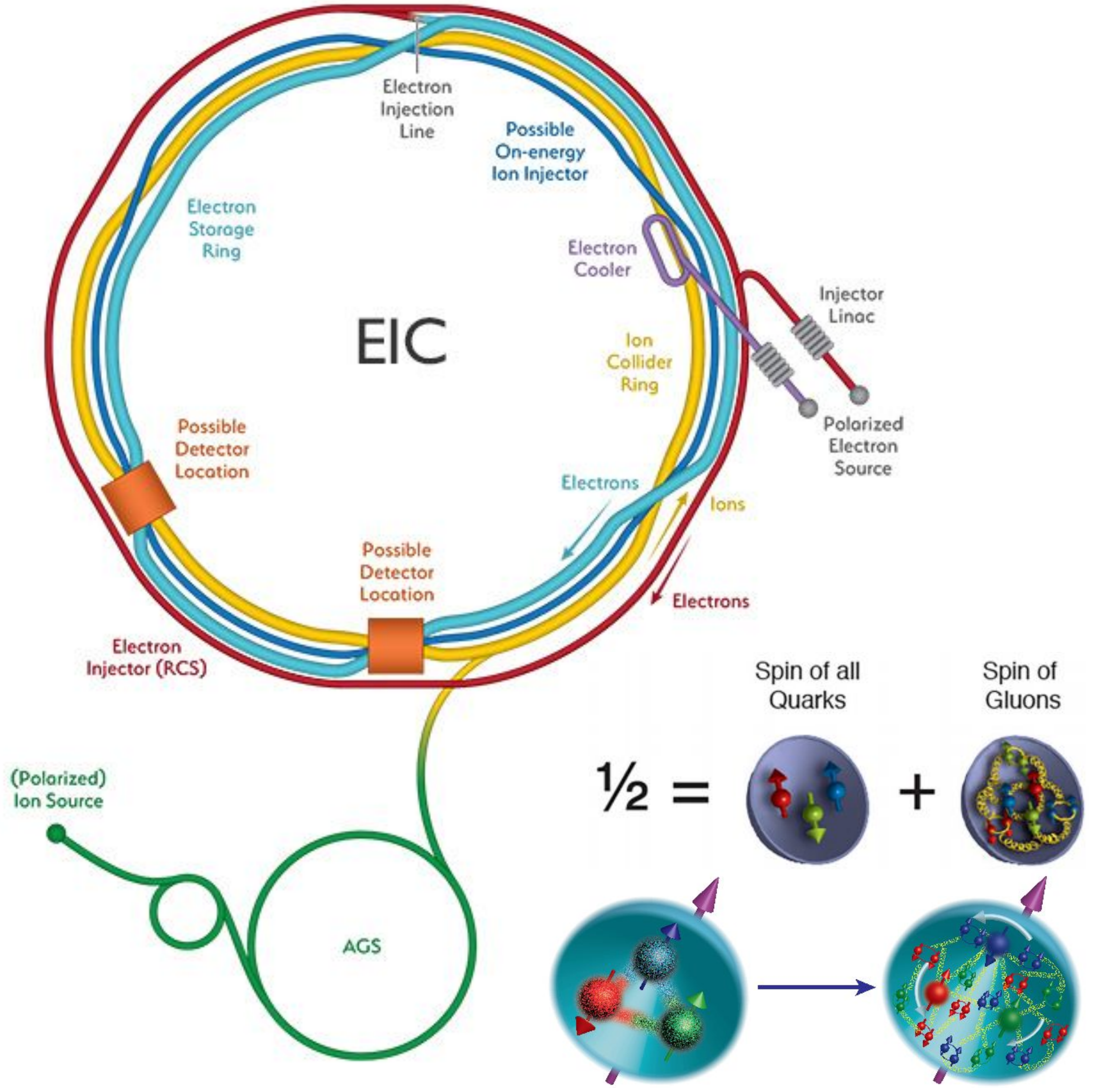
Wouter Deconinck, University of Manitoba

- Nuclear physics experiments increasing to O(1000) user community
  - Particularly Electron-Ion Collider, but following path by STAR/PHENIX/sPHENIX/GlueX/etc
  - Changing expectations: streaming readout through globally distributed computing centers; regional access to automatically replicated data; tight integration of metadata with original data; data analysis preservation; FAIR data access; alignment with science/industry standards
- User-centered design in practice at the Electron-Ion Collider:
  - What is user-centered design? Who are the users?
  - Gaining understanding → communicating understanding → implementing better solutions
  - Intentional selection process for software components, as well as for computing infrastructure

With input from Markus Diefenthaler, Sylvester Joosten, Kolja Kauder, and many others. Supported in part by NSERC SAPIN-2020-00049, SAPPJ-2021-00026.

Preparing for the Electron-Ion Collider Era (2030s)

- **First major collider** to be built in North America in the 21st century
  - **Polarized electrons**, 10-20 GeV
  - **Polarized light ions** (p, d, ^{3}He) and unpolarized nuclei → U, 50-250 GeV
  - Center of mass energy of 20-140 GeV
  - **High luminosity** $\mathcal{L}$ of 10^{34} cm^{-2} s^{-1}
- International facility, approx $2.6B
- Large community of 1200+ users at 220+ institutions in 30+ countries
- Mass of nucleon? Spin of nucleon? Emergent properties of dense gluons systems?



Milestones in the Electron-Ion Collider Software Development

- **2016**: **EIC Software Consortium** as part of EIC Generic R&D program
  - Activities included: interoperability and common interfaces between simulation components
  - Produce consensus-based community documents setting out vision for EIC software
- **2019**: EIC User Group **Software Working Group**
  - Community endorsement of software as a valuable endeavor for the EIC user group
  - Focus on preparing the community for detector collaborations
    - Coordination during Yellow Report preparation
    - Coordination during detector proposal development
- **2022-2023**: ePIC Collaboration **Computing & Software WGs**
  - **Single software stack decision process**, together with EICUG SWG
  - Balancing rapid project progress with need to organize collaboration around future-oriented software: new nuclear physics tools needed for the scale of the Electron-Ion Collider
- **Overall goals**: development of a community-supported full-lifecycle software stack for nuclear physics; prevent fragmentation; encourage modularity

Preparing for the Electron-Ion Collider Era (2030s)

Requirements are different from many previous nuclear physics experiments:

- Single site (Long Island, NY), but **two host labs** (BNL, JLab), and developed as **highly international facility** with e.g. EIC RRB, significant international in-kind contributions, including for **distributed computing**
  - Strong desire to use distributed computing while previous nuclear physics experiments have primarily used host lab resources (GlueX and others: extensive use of Open Science Grid)
- From triggered to **streaming readout**; offline reconstruction to **rapid results**
  - Data streams sent to distributed computing sites during data taking for immediate processing
  - Reconstructed data and analyzed results available on distributed storage with rapid turnaround

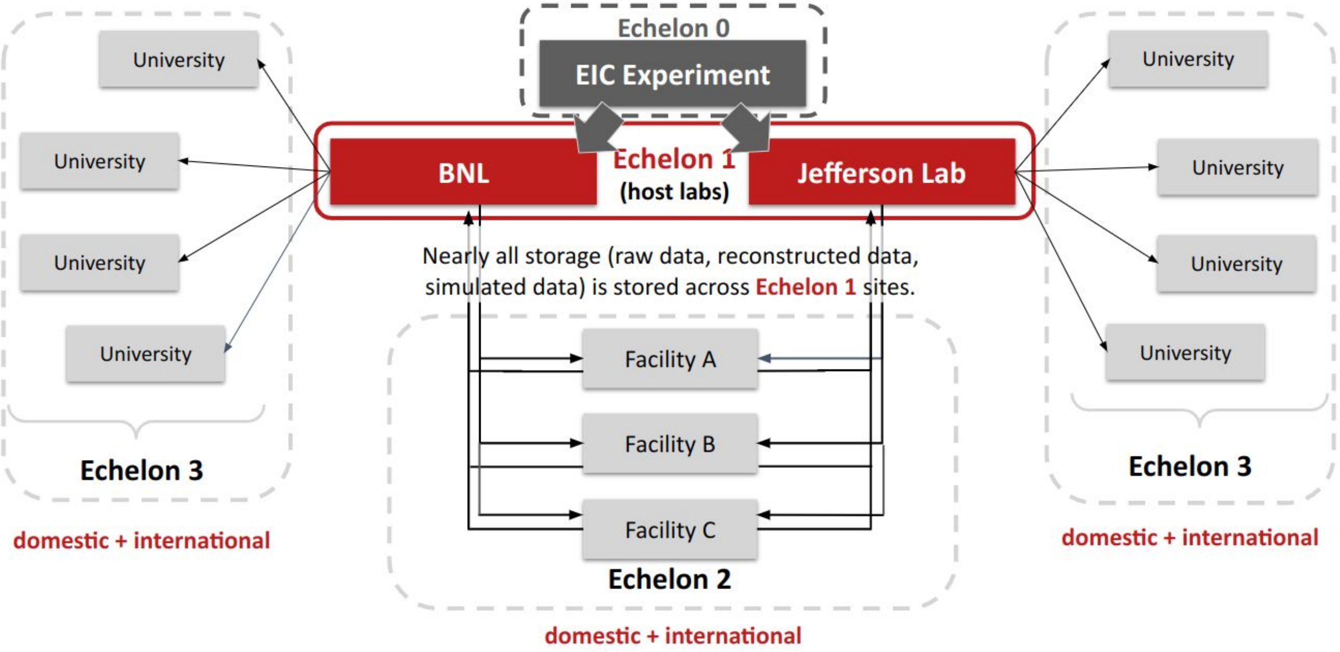
Unique opportunity to develop the data analysis environment in the AI/ML era; novel methodologies required (similar to or even beyond HEP in some areas)

Electron-Ion Collider’s Butterfly Model of Computing

The presence of **two host labs** with MOUs with various universities means that a model of one hub with many spokes will not be a natural evolution. **This computing model has wings.**

International agencies want to support their researchers in their science programs with in-kind computing resources, but only if those are inherent **part of the computing strategy.**

Instead of a primary focus on a single site for computing (as is still common in nuclear physics), the EIC community envisions a **distributed computing model from the start.**



Nearly all storage (raw data, reconstructed data, simulated data) is stored across Echelon 1 sites.

Flexibility to add new computing infrastructure; scaling with community size and computing requirements; inclusion of HPC for AI/ML

Electron-Ion Collider Requires a Cultural Change

Culture [ˈkəl-chər] *(noun)*

“The way we do things around here” (Deal and Kennedy, 1982)

The evolution in nuclear physics data management that has lead us up to the EIC is requiring nuclear physicists to “do things differently around here”

- computing/software systems are becoming **larger, more complex** (no more: nuclear physics is “like HEP but where you can still understand all the parts”)
- systems must be conquered with **modern organizational practices**: code reviews, social coding, well defined interfaces, intentional decision making,...
- different classes of users must **adapt** in different ways (professor != student)

To guide this culture change, we must understand our community.

What is User-Centered Design?

Multi-stage problem-solving process that requires software developers

- to **analyze** and envision the way users are likely to use a product,
- to **validate** their assumptions in real world tests.

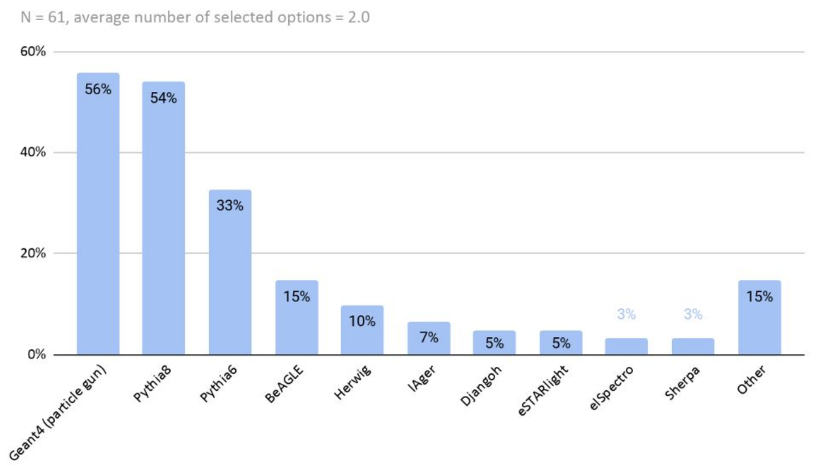
In nuclear physics computing, in the context of the Electron-Ion Collider:

- **Long-term** researchers (pre-existing mental models and expectations)
  - Faculty, staff scientists
  - Experienced in the traditional ways, too busy to learn a new language or tool (unless we make it easy to change existing mental models: it is our task to know the users' attitudes)
- **Short-term** researchers (frequent training need, short-term career goals)
  - Undergraduate, graduate researchers, post-doctoral researchers
  - Trained in more modern languages (or not at all), motivated to learn and apply new skills if empowered and transferable to future opportunities

User-Centered Design at the EIC

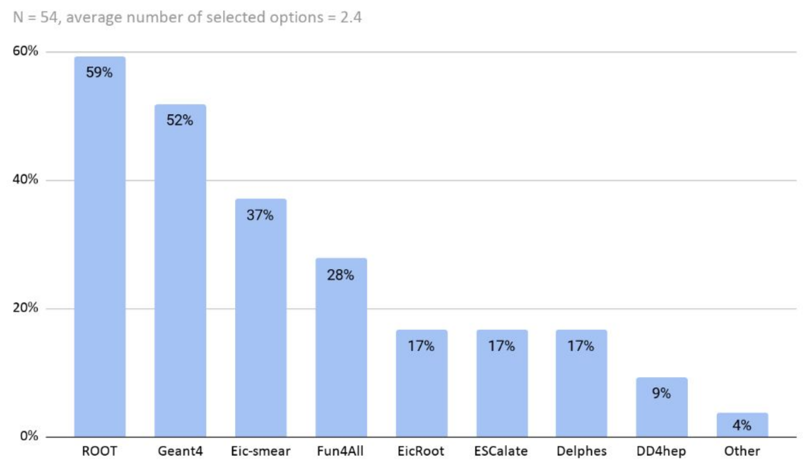
- Annual “State of EIC Software” exercise (2021, 2022)
    - Quantitative survey with consistent questions from year to year (observe longitudinal impact)
    - Qualitative focus groups (~5 users each) to drill into recurring themes for classes of user
- Development of user personas to highlight diversity of experiences

Over the past year, which physics event generation tools did you use for EIC simulations?



Other (N = 9): personal computer codes (N = 2), ACT, CLASDIS, ComptonRad, GRAPE-DILEPTON, MADX, MILOU, OPERA, RAYTRACE, Sartre, Topeg, ZGOUBI

Over the past year, which detector simulation tools did you use for EIC simulations?



Other (N = 2): GEMC, RAYTRACE

EIC User Profile: Vaibhavi Gawas

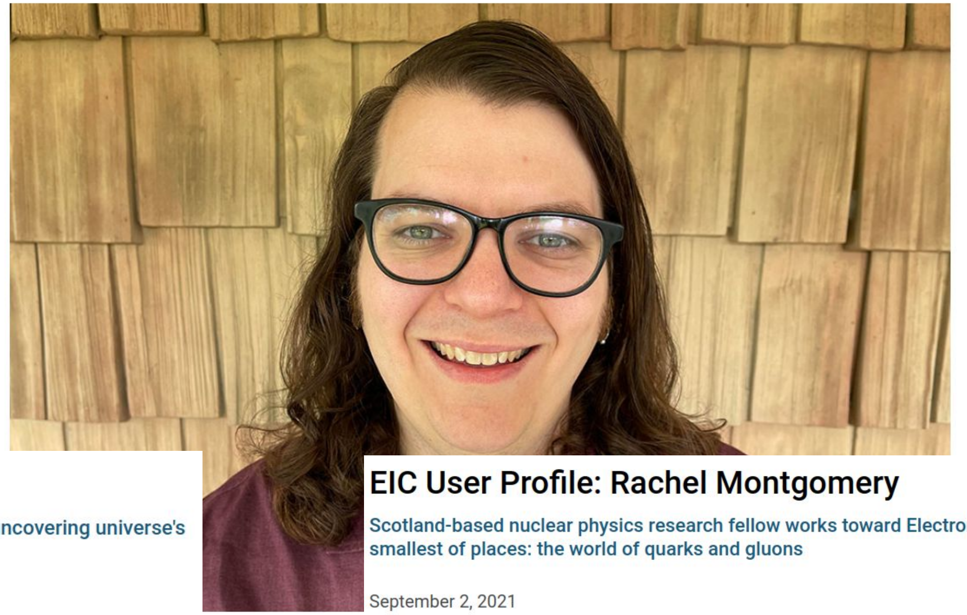
Undergraduate student studying physics at the Indian Institute of Technology (IIT) Madras, India, conducts simulations of Electron-Ion Collider interactions

August 3, 2021

EIC User Profile: Alexander Jentsch

Postdoctoral fellow works to develop detectors for particles that emerge very close to colliding beams at the Electron-Ion Collider

August 5, 2021



EIC User Profile: Prabhakar Palni

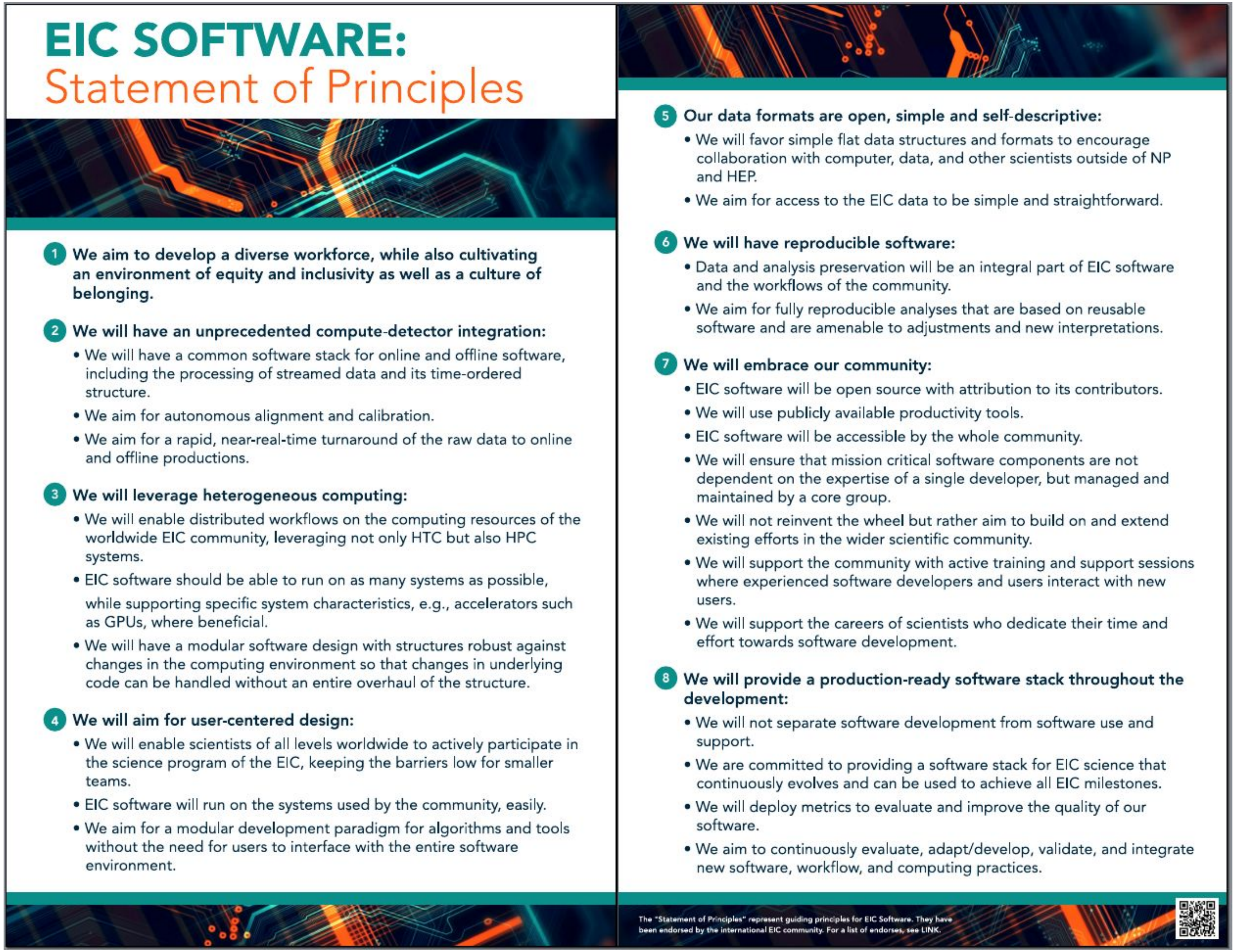
Goa University assistant professor joins Electron-Ion Collider collaboration in hopes of uncovering universe's building blocks

August 27, 2021

EIC User Profile: Rachel Montgomery

Scotland-based nuclear physics research fellow works toward Electron-Ion Collider in hopes of journey to the smallest of places: the world of quarks and gluons

September 2, 2021



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.

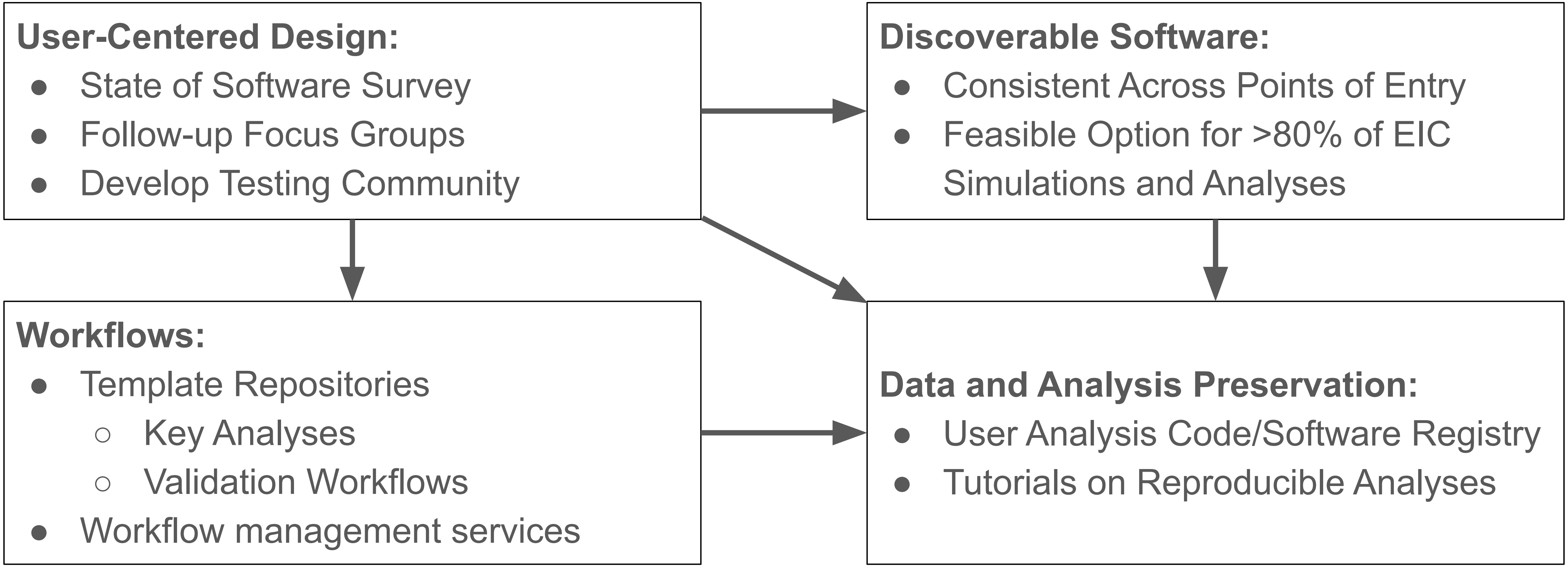
8. **We will provide a production-ready software stack throughout the development:**
   - We will not separate software development from software use and support.
   - We are committed to providing a software stack for EIC science that continuously evolves and can be used to achieve all EIC milestones.
   - 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.

Computing is

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

User-Centered Design at the EIC: Model for Software



User-Centered Design:

- State of Software Survey
- Follow-up Focus Groups
- Develop Testing Community

Discoverable Software:

- Consistent Across Points of Entry
- Feasible Option for >80% of EIC Simulations and Analyses

Workflows:

- Template Repositories
  - Key Analyses
  - Validation Workflows
- Workflow management services

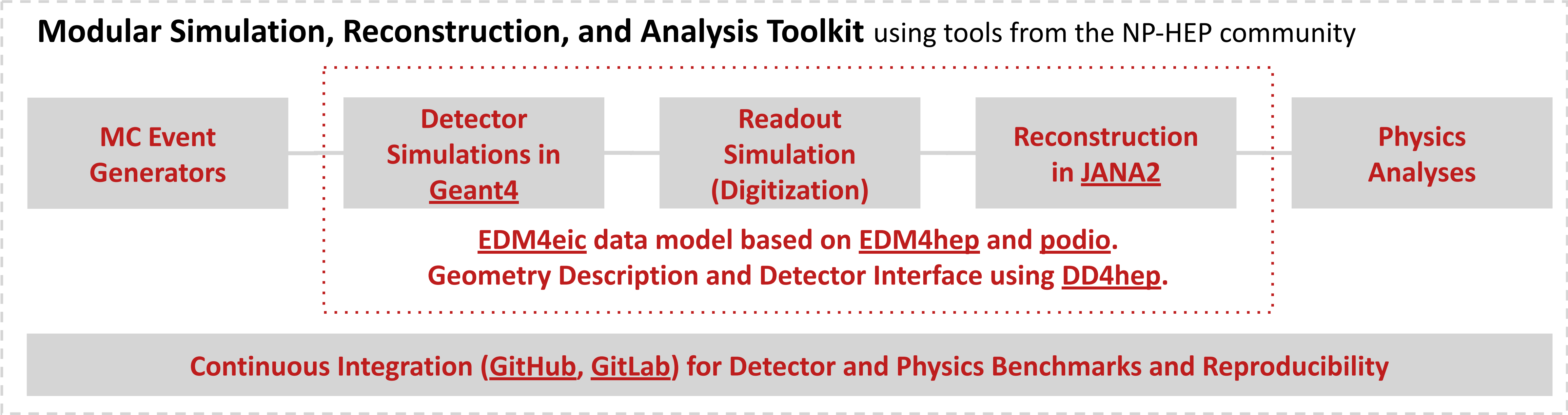
Data and Analysis Preservation:

- User Analysis Code/Software Registry
- Tutorials on Reproducible Analyses

Most of these approaches are new to nuclear physicists!

User-Centered Design at the EIC: Software Stack

- Using open-source, community-oriented software components from NP-HEP, with focus on software sustainability in selection



Modular Simulation, Reconstruction, and Analysis Toolkit using tools from the NP-HEP community

MC Event Generators

Detector Simulations in Geant4

Readout Simulation (Digitization)

Reconstruction in JANA2

Physics Analyses

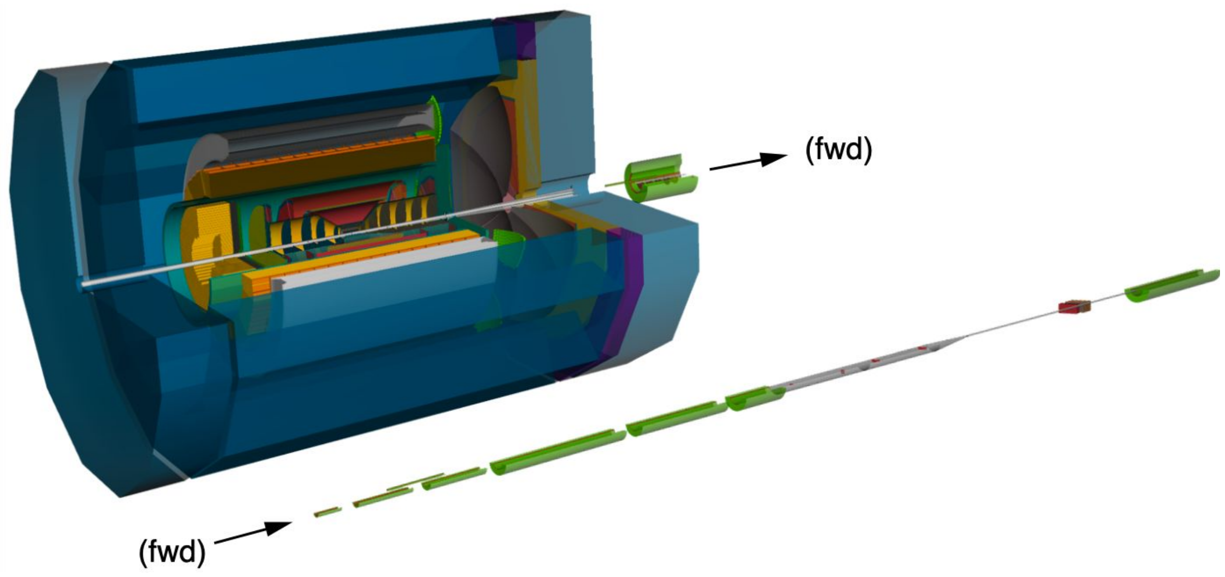
EDM4eic data model based on EDM4hep and podio.
Geometry Description and Detector Interface using DD4hep.

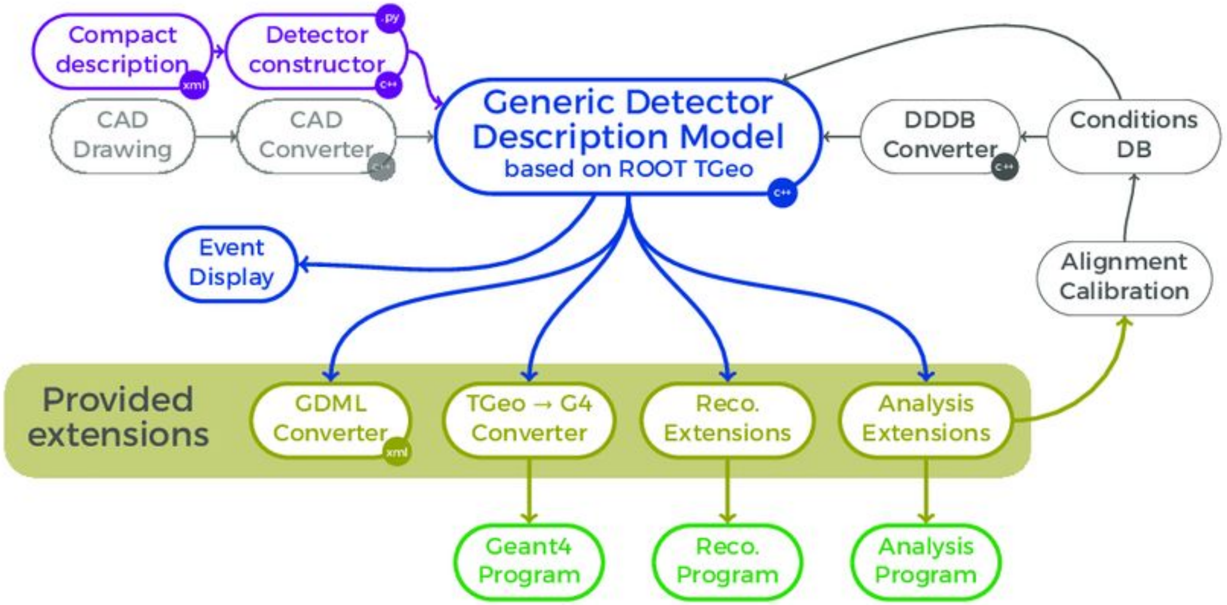
Continuous Integration (GitHub, GitLab) for Detector and Physics Benchmarks and Reproducibility

Geometry Description and Exchange: DD4hep

Requirement: **consistent geometry** for simulations, reconstruction, data taking

- DD4hep: Abstraction layer for Geant4, TGeo, and other geometry consumers
- Geometry service from simulation through reconstruction and analysis
- Community-managed external project with large experimental user base

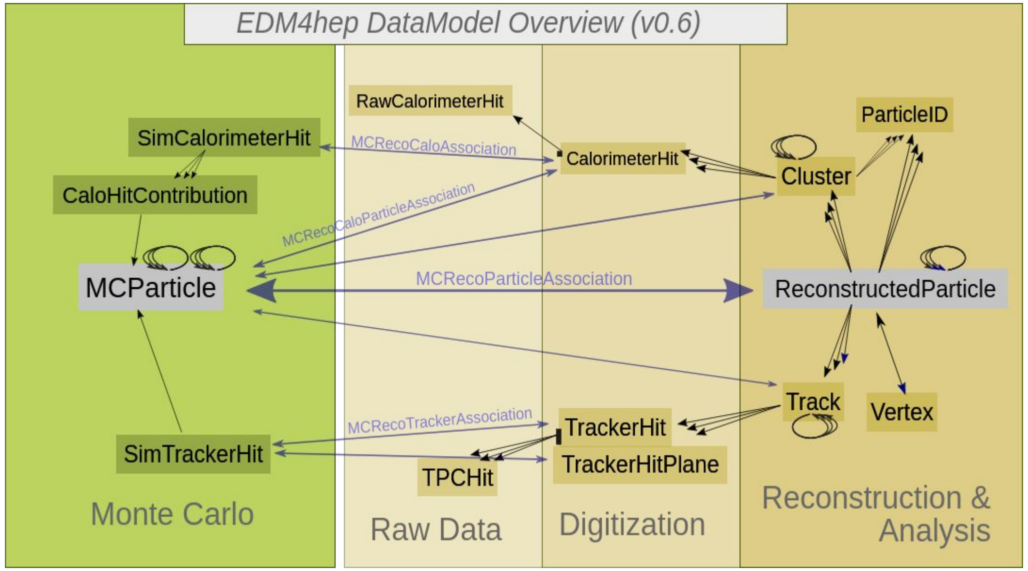


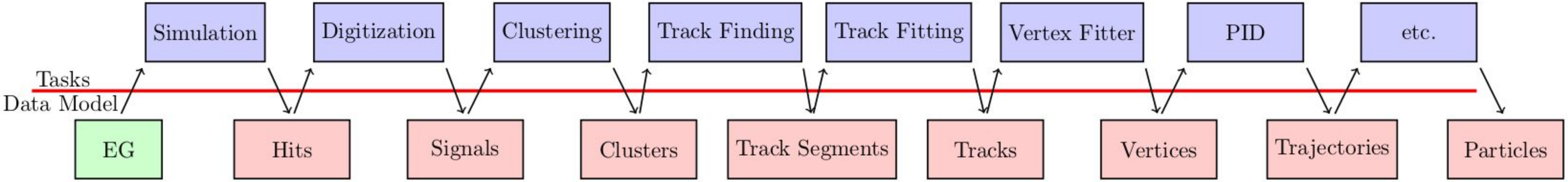


Data Model: `podio`, `EDM4hep`, `EDM4eic`

Use of **standard interfaces** between individual simulation, reconstruction, and analysis tasks **creates modularity** that allows **easy exchange of components**.

- `podio` (github.com/AIDASoft/podio)
  - ○ Text-based definition of flat data models
  - ○ Automatic C++ and Python interfaces
  - ○ Stored inside ROOT files or other formats
- `EDM4hep` (github.com/key4hep/EDM4hep)
  - ○ Designed as a standard for current/future HEP
  - ○ `EDM4eic`: few EIC-specific extension data types
  - ○ Struggled to define in EIC for several years





Code Repositories and Continuous Integration

Code Repositories:

Centralized collaborative development of all software components, for preservation of a full record of the development activity.

Several widely used options based on `git`:

- GitHub (github.com or enterprise)
- GitLab (gitlab.com or self-hosted)

We use a GitHub organization, github.com/eic

Milestones and versioning, reproducibility, preservation, collaboration, code review

Continuous Integration/Deployment (CI/CD):

A strategy of **automatic evaluation** of software components, and of **automatic deployment** into testing and production environments.

Tightly integrated with repositories:

- GitHub
- GitLab

We use GitLab instances on national lab resources triggered from GitHub.

Automation, quality control, workflows, deployment into production environments

Community-Supported Components

Experiment-agnostic community software projects are **key to our strategy.**

- Key4HEP turnkey software stack, Phoenix event display, HSF conditions database reference implementation, Acts track reconstruction
- Spack package management, Rucio scientific data management

Thanks to all HEP Software Foundation, AIDAInnova, IRIS-HEP,... contributors!

Challenges to this model we are navigating in the nuclear physics community:

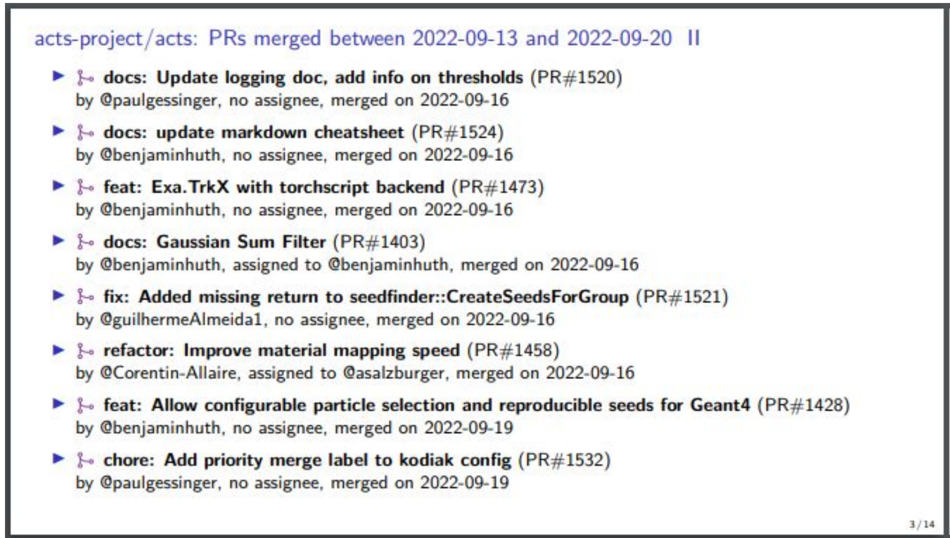
- Users' **lack of understanding** that external projects do not work "for" the EIC (because that's all they have ever known in entirely home-grown software)
- **"Externality" risk** attributed by project management to community-supported software components

Community-Supported Components: Acts

Weekly dev meeting with involvement of users at multiple experiments

Status of work visibility through presentations

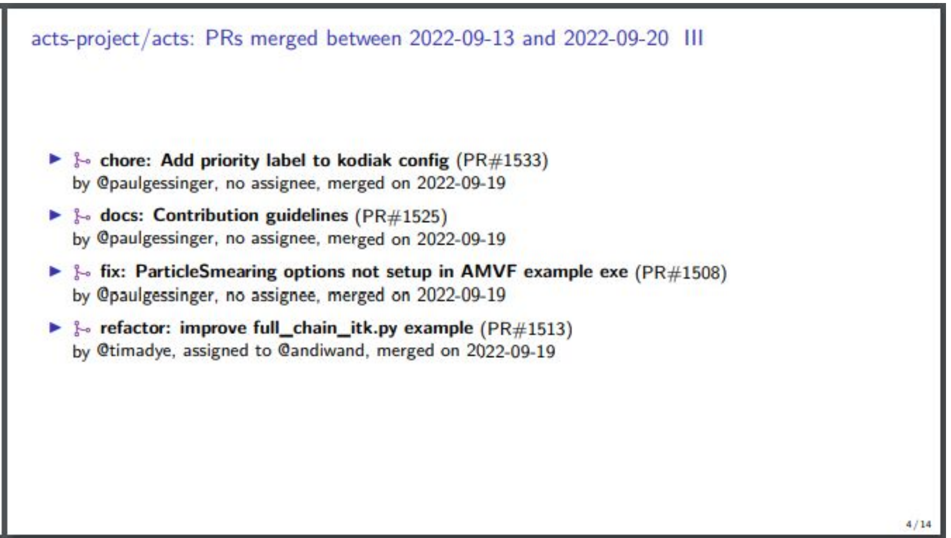
Example of agile in community software



acts-project/acts: PRs merged between 2022-09-13 and 2022-09-20 II

- **docs: Update logging doc, add info on thresholds** (PR#1520)
  by @paulgessinger, no assignee, merged on 2022-09-16
- **docs: update markdown cheatsheet** (PR#1524)
  by @benjaminhuth, no assignee, merged on 2022-09-16
- **feat: Exa.TrkX with torchscript backend** (PR#1473)
  by @benjaminhuth, no assignee, merged on 2022-09-16
- **docs: Gaussian Sum Filter** (PR#1403)
  by @benjaminhuth, assigned to @benjaminhuth, merged on 2022-09-16
- **fix: Added missing return to seedfinder::CreateSeedsForGroup** (PR#1521)
  by @guilhermeAlmeida1, no assignee, merged on 2022-09-16
- **refactor: Improve material mapping speed** (PR#1458)
  by @Corentin-Allaire, assigned to @asalzburger, merged on 2022-09-16
- **feat: Allow configurable particle selection and reproducible seeds for Geant4** (PR#1428)
  by @benjaminhuth, no assignee, merged on 2022-09-19
- **chore: Add priority merge label to kodiak config** (PR#1532)
  by @paulgessinger, no assignee, merged on 2022-09-19

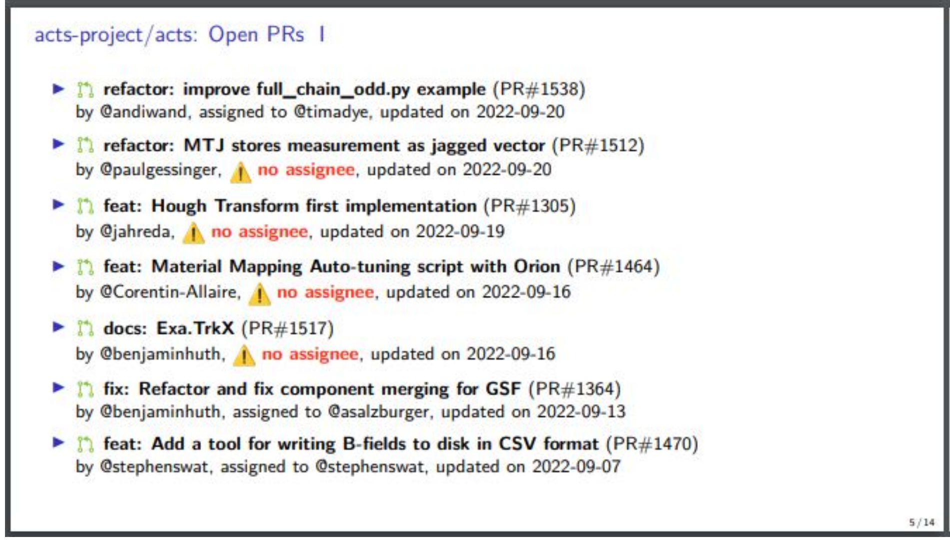
3 / 14



acts-project/acts: PRs merged between 2022-09-13 and 2022-09-20 III

- **chore: Add priority label to kodiak config** (PR#1533)
  by @paulgessinger, no assignee, merged on 2022-09-19
- **docs: Contribution guidelines** (PR#1525)
  by @paulgessinger, no assignee, merged on 2022-09-19
- **fix: ParticleSmearing options not setup in AMVF example exe** (PR#1508)
  by @paulgessinger, no assignee, merged on 2022-09-19
- **refactor: improve full_chain_itk.py example** (PR#1513)
  by @timadye, assigned to @andiwand, merged on 2022-09-19

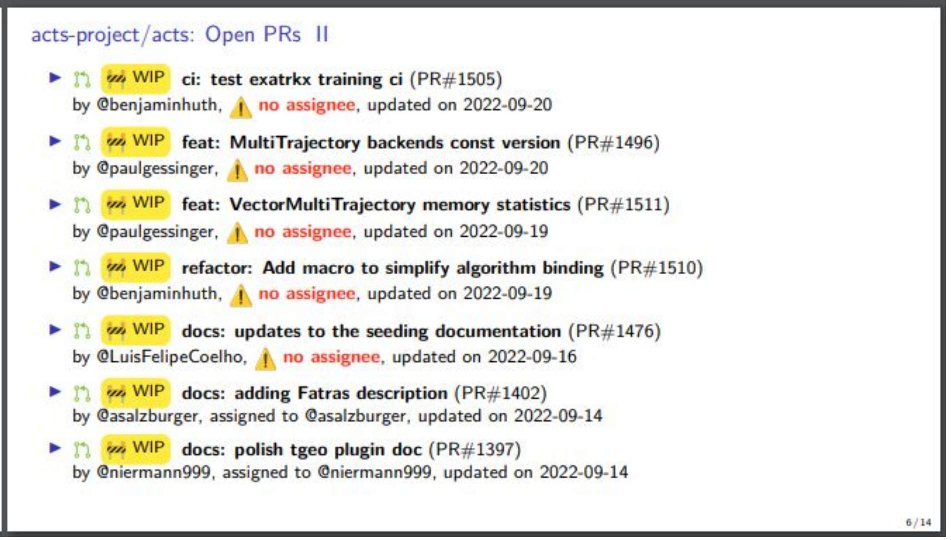
4 / 14



acts-project/acts: Open PRs I

- **refactor: improve full_chain_odd.py example** (PR#1538)
  by @andiwand, assigned to @timadye, updated on 2022-09-20
- **refactor: MTJ stores measurement as jagged vector** (PR#1512)
  by @paulgessinger, ⚠ no assignee, updated on 2022-09-20
- **feat: Hough Transform first implementation** (PR#1305)
  by @jahreda, ⚠ no assignee, updated on 2022-09-19
- **feat: Material Mapping Auto-tuning script with Orion** (PR#1464)
  by @Corentin-Allaire, ⚠ no assignee, updated on 2022-09-16
- **docs: Exa.TrkX** (PR#1517)
  by @benjaminhuth, ⚠ no assignee, updated on 2022-09-16
- **fix: Refactor and fix component merging for GSF** (PR#1364)
  by @benjaminhuth, assigned to @asalzburger, updated on 2022-09-13
- **feat: Add a tool for writing B-fields to disk in CSV format** (PR#1470)
  by @stephenswat, assigned to @stephenswat, updated on 2022-09-07

5 / 14



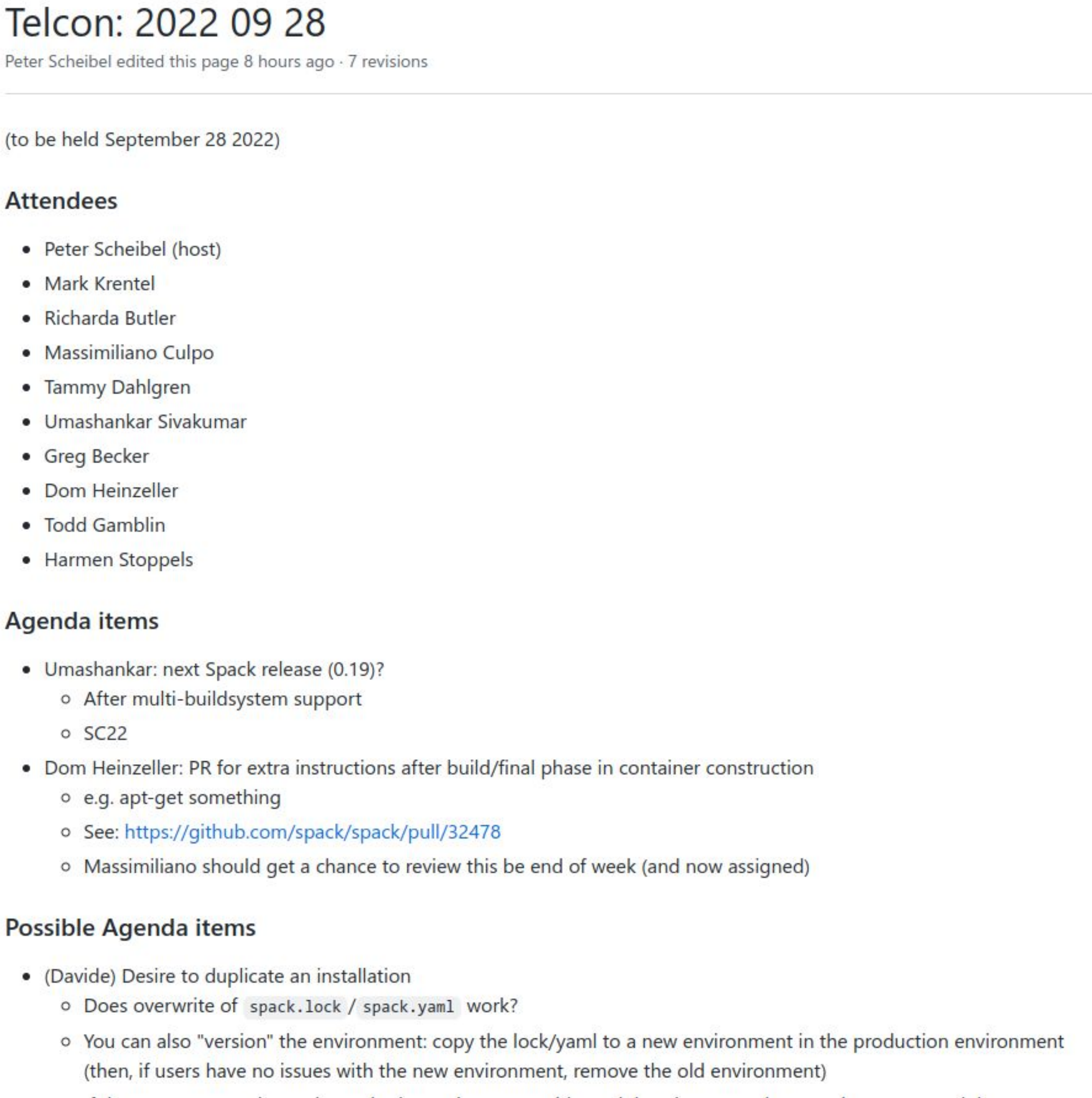
acts-project/acts: Open PRs II

- WIP **ci: test exatrkx training ci** (PR#1505)
  by @benjaminhuth, ⚠ no assignee, updated on 2022-09-20
- WIP **feat: MultiTrajectory backends const version** (PR#1496)
  by @paulgessinger, ⚠ no assignee, updated on 2022-09-20
- WIP **feat: VectorMultiTrajectory memory statistics** (PR#1511)
  by @paulgessinger, ⚠ no assignee, updated on 2022-09-19
- WIP **refactor: Add macro to simplify algorithm binding** (PR#1510)
  by @benjaminhuth, ⚠ no assignee, updated on 2022-09-19
- WIP **docs: updates to the seeding documentation** (PR#1476)
  by @LuisFelipeCoelho, ⚠ no assignee, updated on 2022-09-16
- WIP **docs: adding Fatras description** (PR#1402)
  by @asalzburger, assigned to @asalzburger, updated on 2022-09-14
- WIP **docs: polish tgeo plugin doc** (PR#1397)
  by @niermann999, assigned to @niermann999, updated on 2022-09-14

6 / 14

Community-Supported Components: Spack

- HPC with strong engagement with/from the NP/HEP community
- Weekly open telcon meetings: any users are welcome to bring up issues (also: office hours)
- Appreciation/gripes channels on slack with low-barrier user feedback



Telcon: 2022 09 28

Peter Scheibel edited this page 8 hours ago · 7 revisions

(to be held September 28 2022)

Attendees

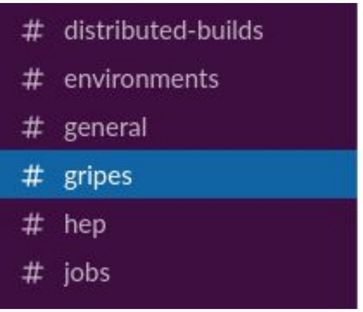
- Peter Scheibel (host)
- Mark Krentel
- Richarda Butler
- Massimiliano Culpo
- Tammy Dahlgren
- Umashankar Sivakumar
- Greg Becker
- Dom Heinzeller
- Todd Gamblin
- Harmen Stoppels

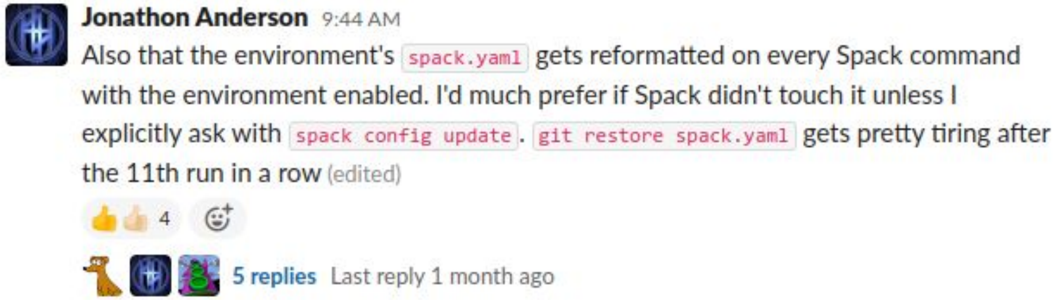
Agenda items

- Umashankar: next Spack release (0.19)?
  - After multi-buildsystem support
  - SC22
- Dom Heinzeller: PR for extra instructions after build/final phase in container construction
  - e.g. apt-get something
  - See: https://github.com/spack/spack/pull/32478
  - Massimiliano should get a chance to review this be end of week (and now assigned)

Possible Agenda items

- (Davide) Desire to duplicate an installation
  - Does overwrite of `spack.lock` / `spack.yaml` work?
  - You can also "version" the environment: copy the lock/yaml to a new environment in the production environment (then, if users have no issues with the new environment, remove the old environment)





Jonathon Anderson 9:44 AM

Also that the environment's `spack.yaml` gets reformatted on every Spack command with the environment enabled. I'd much prefer if Spack didn't touch it unless I explicitly ask with `spack config update`. `git restore spack.yaml` gets pretty tiring after the 11th run in a row (edited)

5 replies Last reply 1 month ago

Applying This to Distributed Computing Infrastructure

Underlying themes:

- **Using industry/HEP-standard community tools over home-grown code:** Experience with tools and languages that are recognized as important will help researchers get a foot in the door on the job market, while externalizing the development of software frameworks that do not require data analysis skills
- **Allowing researchers to innovate and modify quickly through modularity:** Flexibility to modify independent parts of the data flow or analysis chain allows researchers to develop the relevant data analytics skills and portfolio, rather than getting stuck trying to modify a mountain of interconnected code

This is a decision process with user input to play out over the next year.

Summary

- The **Electron-Ion Collider** brings nuclear physicists in a new and unfamiliar regime of collaboration at the scale of HEP. Traditionally centralized software and computing will need to be replaced by **distributed computing practices**.

- Navigating this change in culture requires an understanding of the users. Through **user-centered design** and user inclusion in software selection processes, we have selected large **community-supported software** and computing components (a process we continue to follow for future decisions).

User-Centered Design
at the Electron-Ion Collider

Milestones in the Electron-Ion Collider Development

- **2021:** Large detector proposal development:
  - **ATHENA**: 3T solenoid, Si+MM+GEM tracker, imaging barrel EM cal, proximity-focused RICH
  - **ECCE**: 1.5T BaBar solenoid, Si+muRWell trackers, projective SciGlass EM cal, modular RICH

    ATHENA: A Totally Hermetic Electron-Nucleus Apparatus

    ECCE *EIC Comprehensive Chromodynamics Experiment*
  - CORE: smaller effort focused on specific exclusive reaction channels at 2nd IR
- **2022:** Selection of ECCE proposal as reference for EIC project detector
  - DPAP advisory panel: ECCE design achieves physics goals with lowest risk and cost
  - Successful integration of ATHENA and ECCE communities within two months(!)
- **2023:** Detector TDR for EIC Project CD-2/3a review (by January 2024)
  - 2022: technology selection for few areas where multiple options
  - 2023: finalization of design parametrization

Milestones in the Electron-Ion Collider Software Development

- **2016**: EIC Software Consortium as part of EIC Generic R&D program
  - Activities included: interoperability and common interfaces between simulation components
  - Produce consensus-based community documents setting out vision for EIC software
- **2019**: EIC User Group Software Working Group
  - Community endorsement of software as a valuable endeavor for the EIC user group
  - Focus on preparing the community for detector (proto-)collaborations
    - Coordinate during Yellow Report preparation
    - Coordinate during proposal development process
    - Prevent (attempt to) fragmentation of software efforts, focus on modularity
- **2022**: EIC “Detector-1” Computing and Software Working Group
  - Single software stack decision process, together with EICUG SWG
  - Short term goals in organizing collaboration around single software stack: at best half of the collaboration will need to learn a lot of new things (at worst the whole collaboration).

From Two EIC Software Stacks…

Fun4All

- Data analysis & simulation framework of the (s)PHENIX experiments at RHIC, adapted for the Electron-Ion Collider and the ECCE proposal
- ROOT-based and steered through macros
- Training benefits:
  - ROOT is the same familiar user interface that everyone is already proficient in (though most users still stuck on ROOT5)
- Disadvantages:
  - Not easy to satisfy Statement of Principles

DD4hep + Gaudi

- Developed for the ATHENA proposal based on experience of LHCb and in alignment with key4HEP project
- Many different modules interfacing with each other
- Training challenges:
  - Multiple different packages with different languages and usage patterns
  - Built on new approaches not traditionally used in nuclear physics
  - Tendency of nuclear physicists to avoid anything not developed in-house

Challenge: how to convince users to abandon what they are familiar with?

… to a Single Software Stack for the EIC

- Advantage of the ability to start from a blank slate, including AI/ML
- Only way to meet the performance requirements for the EIC:
  - Streaming readout, distributed computing
  - Heterogeneous computing and flexibility for future hardware changes
- Deliberative single software selection process ongoing
  - Consider each component separately (geometry encoding, data model, reconstruction framework, code repository, data analysis preservation, etc)
  - Develop a community requirements document with stakeholder input
  - Invite open submission of proposals solutions for each component
  - Presentation and discussion of pros and cons, in the context of the statement of principles
  - Attempt at consensus building among computing/software experts
  - Open for endorsement by community members
  - Consensus positions reached: GitHub repository hosting, gitlab CI backend, DD4hep geometry definition, podio data models, EDM4hep initial data model, JANA2 reconstruction

Applying This to Distributed Computing Infrastructure

“One size fits all” solutions are overwhelming

- E.g. from raw data to final histograms in one process
- Advantages: centralized framework allows for code reuse
- Disadvantages: modifications have steep learning curve, no branch points for new development

Many monolithic “modular” frameworks in circulation

- Over-designed class hierarchies, limited to single language and workflow
- Modularity only by implementing different classes in a strict syntactically enforced environment

Monolithic Frameworks Limit Flexibility

Towards **independent processes with minimal operations (modularity)**

- Formatted input and output streams (JSON, XML)
- Each process is responsible for one task (decoding, correcting, fitting, calibrating, regression,...), modifications can be more easily overseen
- Scalability in number of cores and across connected sites
- Compartmentalization of functionality
- Full container approach (e.g. Docker)

Importance is in the **data model** in addition to **algorithms**

- With clearly defined interfaces between individual operations, individual algorithms can be swapped out easily

Agile Management of Scientific Software Projects

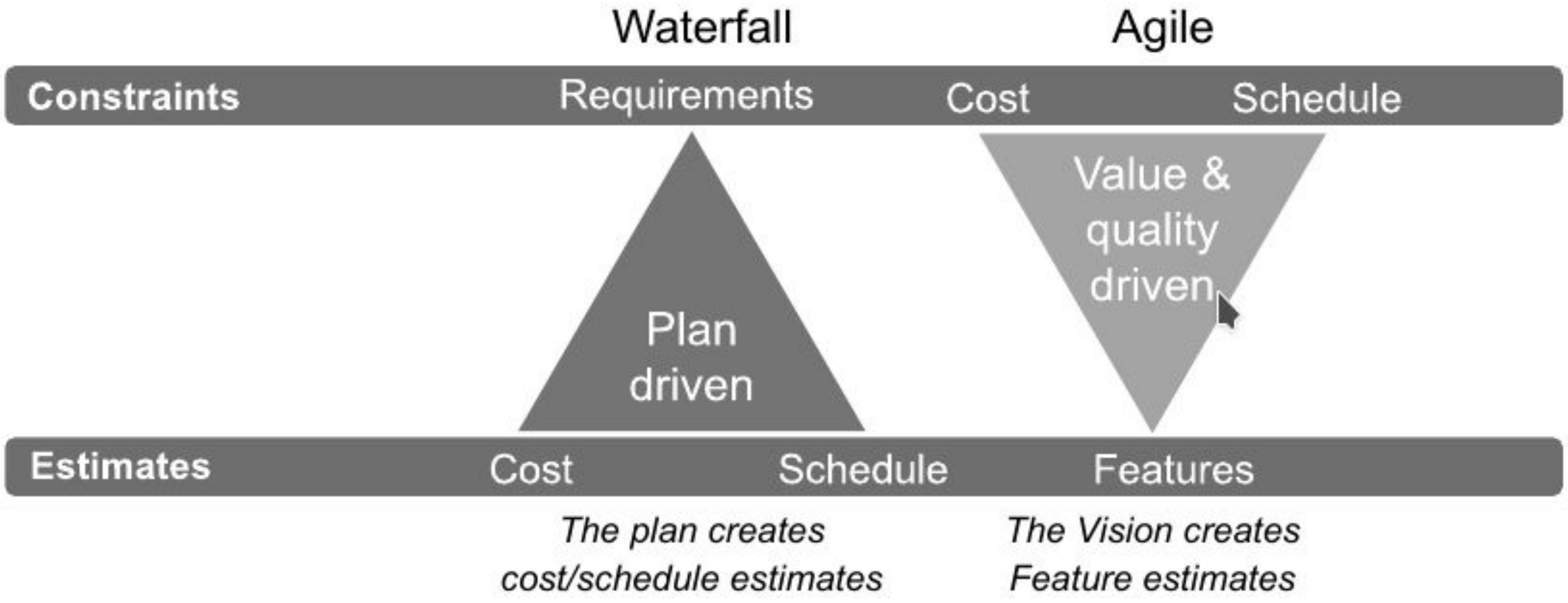
Project Management: Waterfall vs. Agile

Waterfall

- DOE/DOD/NASA: WBSes, gantt charts
- Large projects, extensive planning
- Reqts drive cost and schedule

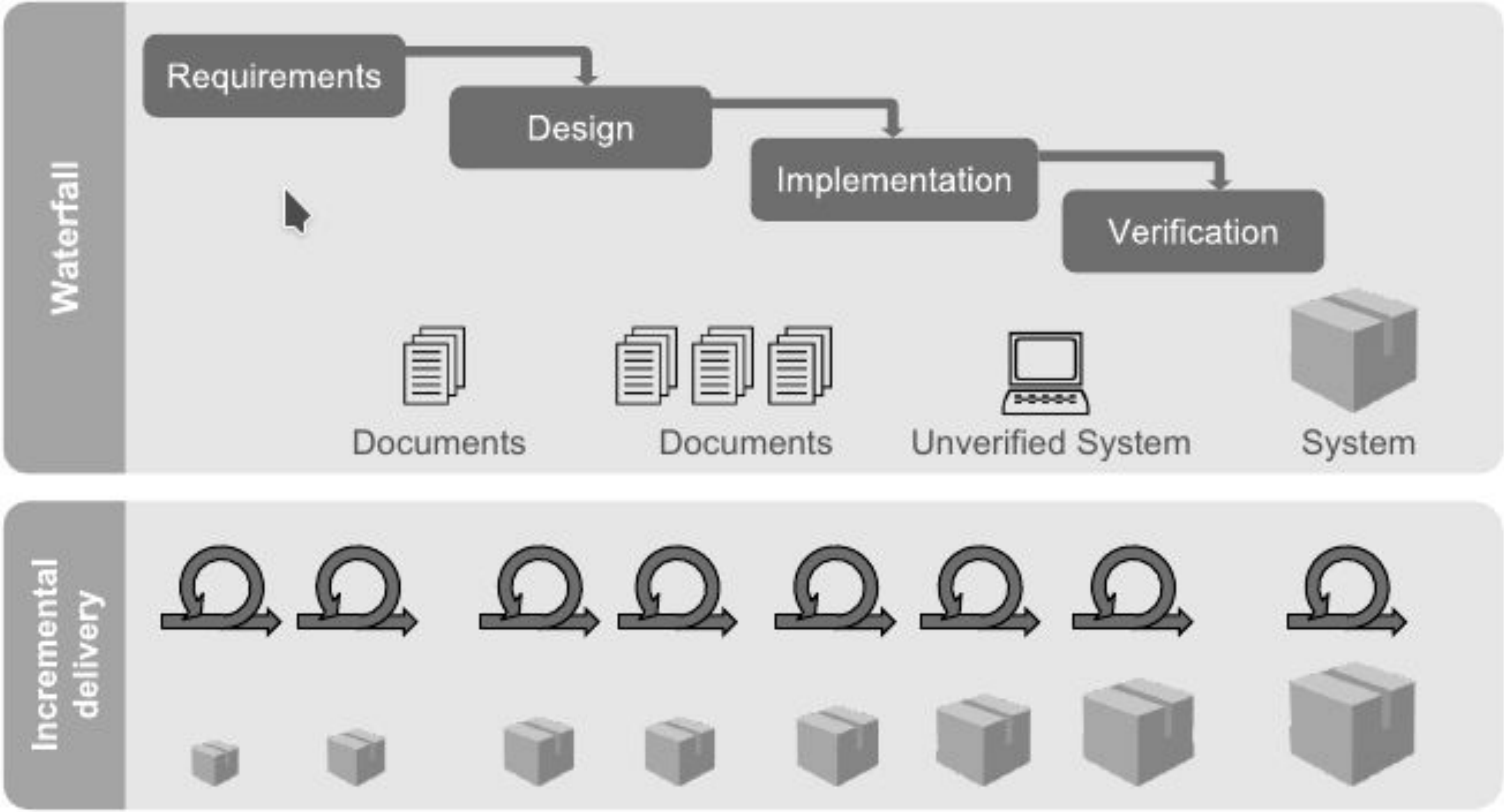
Agile

- Start-ups and collections of small teams, changing reqts
- Current cost and schedule drive features/priority iteratively



Source: Scaled Agile Inc.

Project Management: Waterfall vs. Agile



Source: Scaled Agile Inc.

Project Management: Waterfall vs. Agile

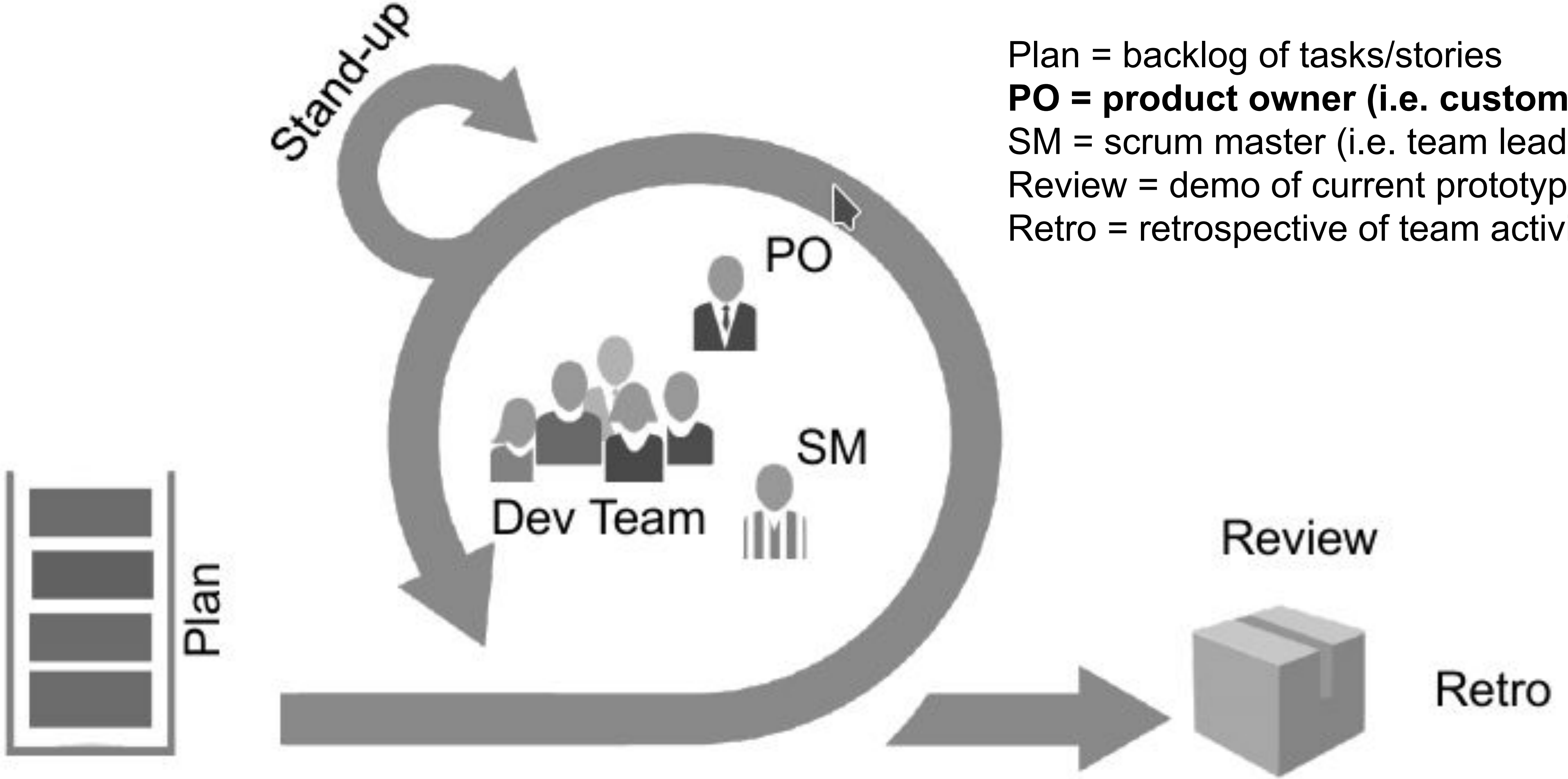
Agile Manifesto (2001, http://agilemanifesto.org/)

- **Individuals and interactions** over processes and tools
- **Working products** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Related initiatives: User-Centered Design, Lean, Toyota Production System, DevOps,... Even novels have been written about these approaches:
The Goal (Eliyahu Goldratt), *The Phoenix Project* (Gene Kim et al.)

Project Management: Waterfall vs. Agile



Plan = backlog of tasks/stories
PO = product owner (i.e. customer)
SM = scrum master (i.e. team leader)
Review = demo of current prototype
Retro = retrospective of team activity

Source: Scaled Agile Inc.

Core components of agile frameworks

- Easy visualization of work in progress
  - Reduce work in process, “batch size of one” in manufacturing, avoid multi-tasking
  - Make status of work visible, kanban boards (ready, doing, done)
- Frequent feedback on project progress and people performance
  - Daily or near-daily short stand-up meetings
  - Iterative sprints with customer
  - Retrospections on team performance
- Continuous improvement
  - Learn by doing, learn by failing
  - Encourage experimentation
  - Affordable loss principle instead of focusing on possible gain

These are desirable aspects in education as well as project management.

Conclusion

Get to know the user community

Get to know the user community incentives for using software

Involve the user community in your software development

User-Centered Design at the EIC

Who Are Scientific Computing Users?

- **Long-term** researchers (in particular faculty) are largely disconnected from direct low-level software tasks, instead looking at physics outputs
    - common complaint: students don't see the physics, only the code!
    - suggested solution: maybe they shouldn't be writing as much code then?
- **Short-term** researchers are trained or in training, and not (typically) proficient in software development until later in their student career (learning by doing, but often without training wheels)
- Yet, **at least 50% of their time is spent on writing code** instead of thinking about physics (e.g. graduate students at LHC)
- Nuclear physics data analysis has gotten a reputation of being coding intensive at the expense of generating physics insights

Short-Term Researchers

Bachelor degrees in physics

- **Only 1 out of 6 physicists** continues to PhD degree (AIP SRC)
- All other physicists not included in “traditional physicists” interpretation

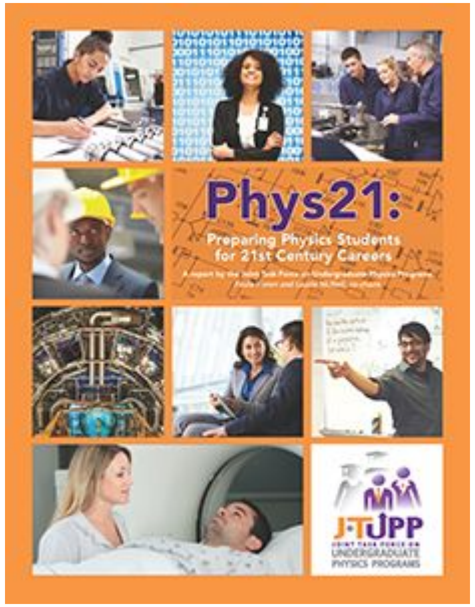
PhD degrees in physics

- **Majority** of the permanent jobs is **outside of academic research**
- About 1700 physics PhDs per year, but significantly fewer jobs
- All other physicists not included in “traditional physicists” interpretation

Mismatch between curriculum and reality of physics teaching

- How can we prepare short-term researchers better for their most likely career?
- How can we align the training of short-term researchers with the skills they will benefit from?

Joint Task Force on Undergraduate Physics Programs


Phys21:
Preparing Physics Students for 21st Century Careers
A report by the Joint Task Force on Undergraduate Physics Programs
Paula Heron and Laurie McNeil, co-chairs
J-TUPP
JOINT TASK FORCE ON UNDERGRADUATE PHYSICS PROGRAMS

Findings (https://www.compadre.org/JTUPP/report.cfm)

- “The overwhelming majority of physics bachelor’s recipients are employed outside academia for all or part of their careers.”
- “Since only about one-third of physics Ph.D. recipients end up in academic careers, even students who plan to obtain graduate degrees will benefit from developing skills and knowledge that are valued outside the academic community.

Promote career readiness: Scientific and technical skills

- “**Competencies** in instrumentation, software, coding, and data analytics.”
- “Introduce students to **industry-standard tools** and software packages.” (what we call “transferable skills” in other contexts)

Who Are Scientific Computing Users?

User-centered design of scientific computing in nuclear physics

- The users (and sometimes developers) of scientific computing in nuclear physics are predominantly the **short-term researchers**, i.e. graduate students and postdoctoral researchers
- User-centered design means **alignment** of scientific computing **with goals** of its users: only that will ensure adoption of our products

Physicists Are Not Preferred Data Scientists Anymore

Over the previous two decades...

- Rapid expansion of data collection capabilities outside of research enterprise, surpassing what high energy and nuclear physics have routinely dealt with for years
- This expansion and presence of physicists with large-scale data analytics training resulted in employment opportunities for many physicists

Physicists Are Not Preferred Data Scientists Anymore

But the educational environment has adapted rapidly

- Nearly every college or university has now implemented a data science concentration, minor, or major, with relevant content based on requirements in industry: relevant algorithms, languages and platforms
- Nuclear and high energy physics have remained largely the same: no major changes in languages, no major adoption of new methodologies (maybe one exception: GPUs)
- **Physicists have lost their edge** in the big data analytics job market
- **Discipline has lost access to a lot of short-term researchers** with training and interest in careers in big data analytics

Intermediate Conclusion

Physics graduate students and postdocs participating in nuclear physics data analysis research are likely to find **permanent employment outside of academic research**. If we are to continue attracting outstanding researchers, we should provide researchers with the scientific and technical skills for future careers inside **and outside academia**, in particular in data science. In the development of a user-centered scientific computing environment, we must keep the motivations of our short-term researchers in mind.

EIC State of Software Survey

ELECTRON ION COLLIDER USER GROUP

USER SURVEY AND FOCUS GROUPS

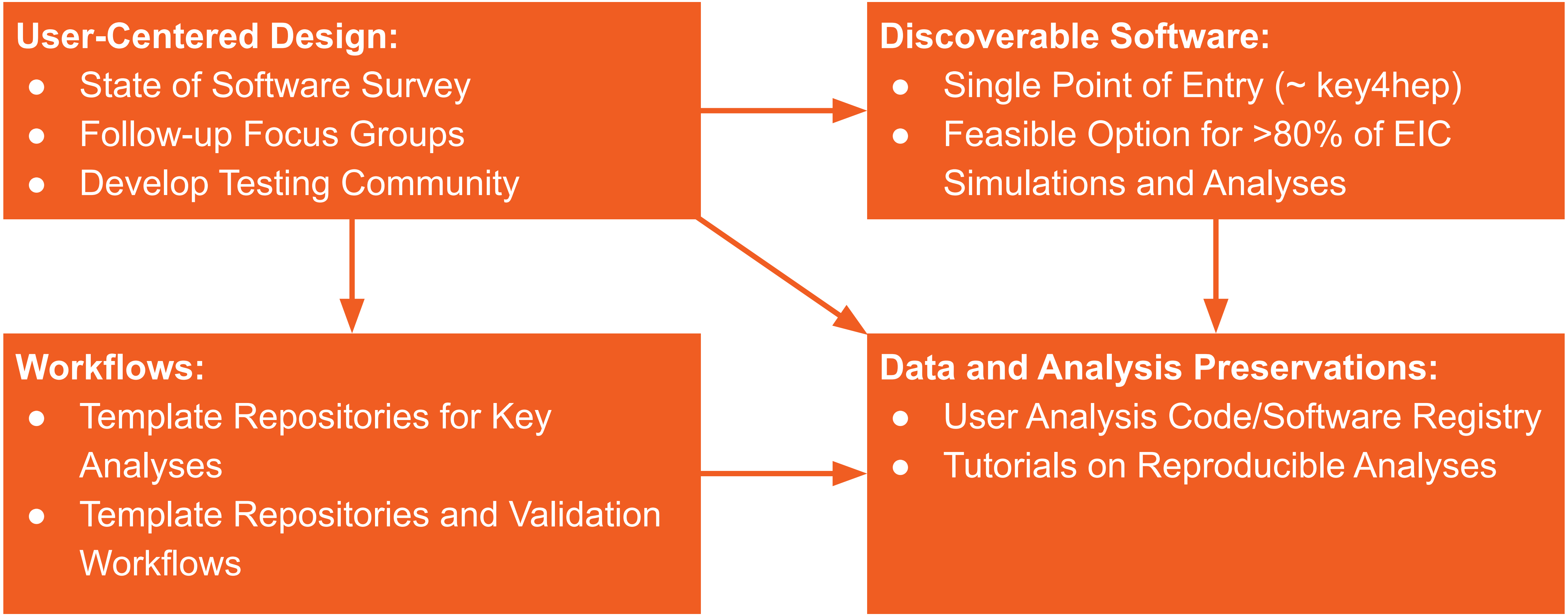
First Insights

1. State of Software Survey: First annual survey of software use in the EIC User Group.

2. User Focus Groups: In-depth follow-up discussions, at first based on career stage.

by Dave Chopard (JLab), Wouter Deconinck (Manitoba), Markus Diefenthaler (JLab), Rebecca Duckett (JLab), Sylvester Joosten (ANL), and Kolja Kauder (BNL).

User-Centered Design: Listen to the Users, and/then Develop Software



User-Centered Design:

- State of Software Survey
- Follow-up Focus Groups
- Develop Testing Community

Discoverable Software:

- Single Point of Entry (~ key4hep)
- Feasible Option for >80% of EIC Simulations and Analyses

Workflows:

- Template Repositories for Key Analyses
- Template Repositories and Validation Workflows

Data and Analysis Preservations:

- User Analysis Code/Software Registry
- Tutorials on Reproducible Analyses

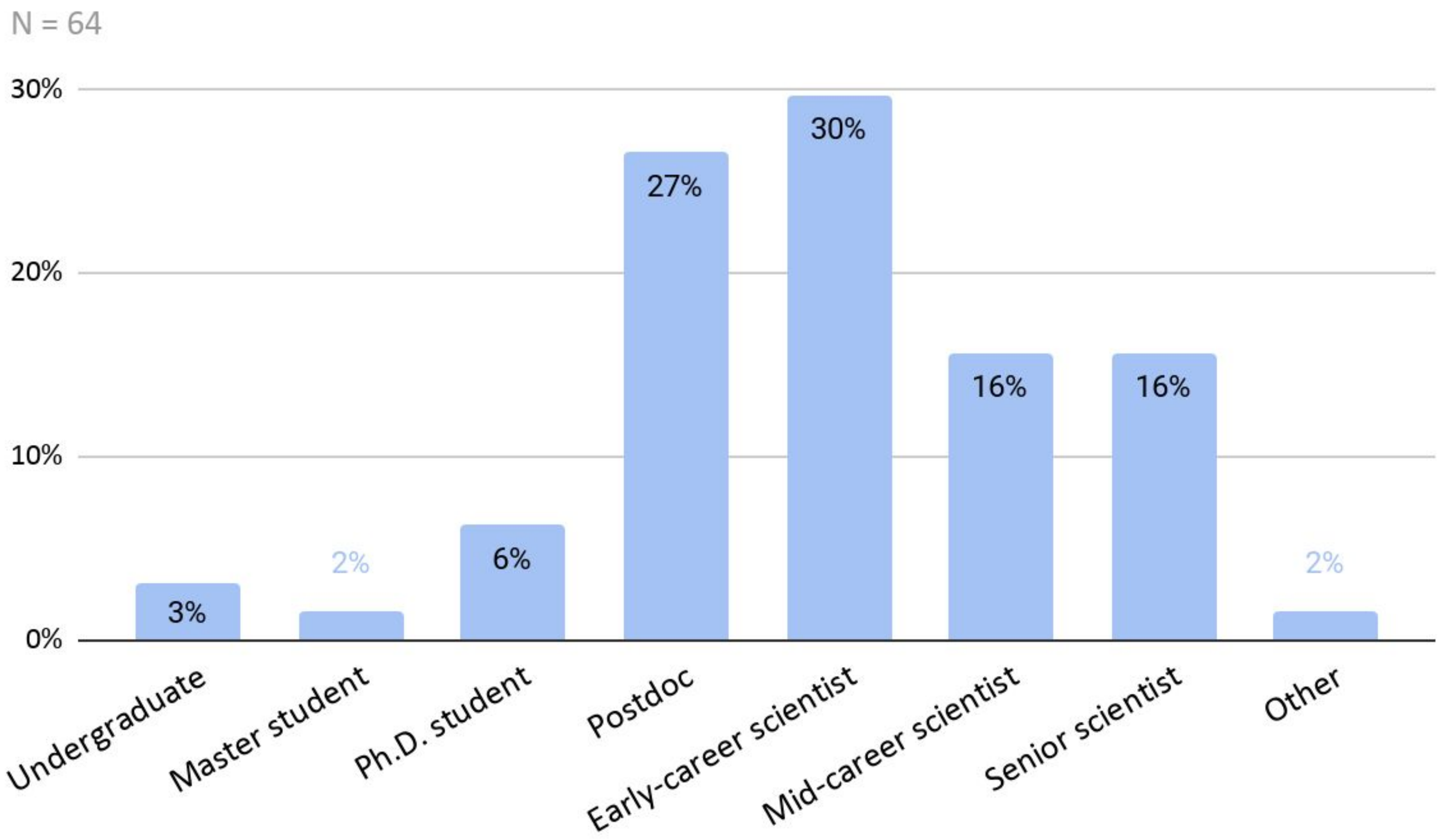
ELECTRON ION COLLIDER USER GROUP

1. STATE OF SOFTWARE SURVEY

Notes:

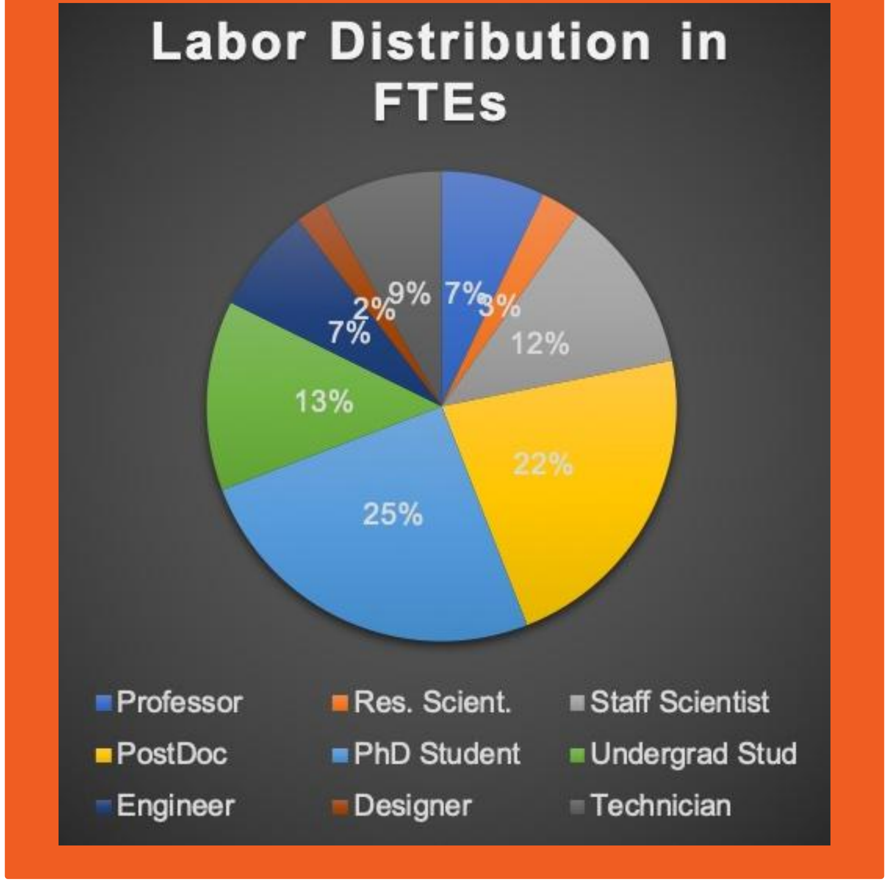
- Distribution methodology: Emails and reminders to EICUG mailing list.
- This year’s data collection has been completed. Your suggestions may be implemented in the next state of software survey (schedule for early 2022).
- A careful balance between collecting detailed information and avoiding survey fatigue means that some tools could not be include as predetermined options.

What is your current role in the EIC project?

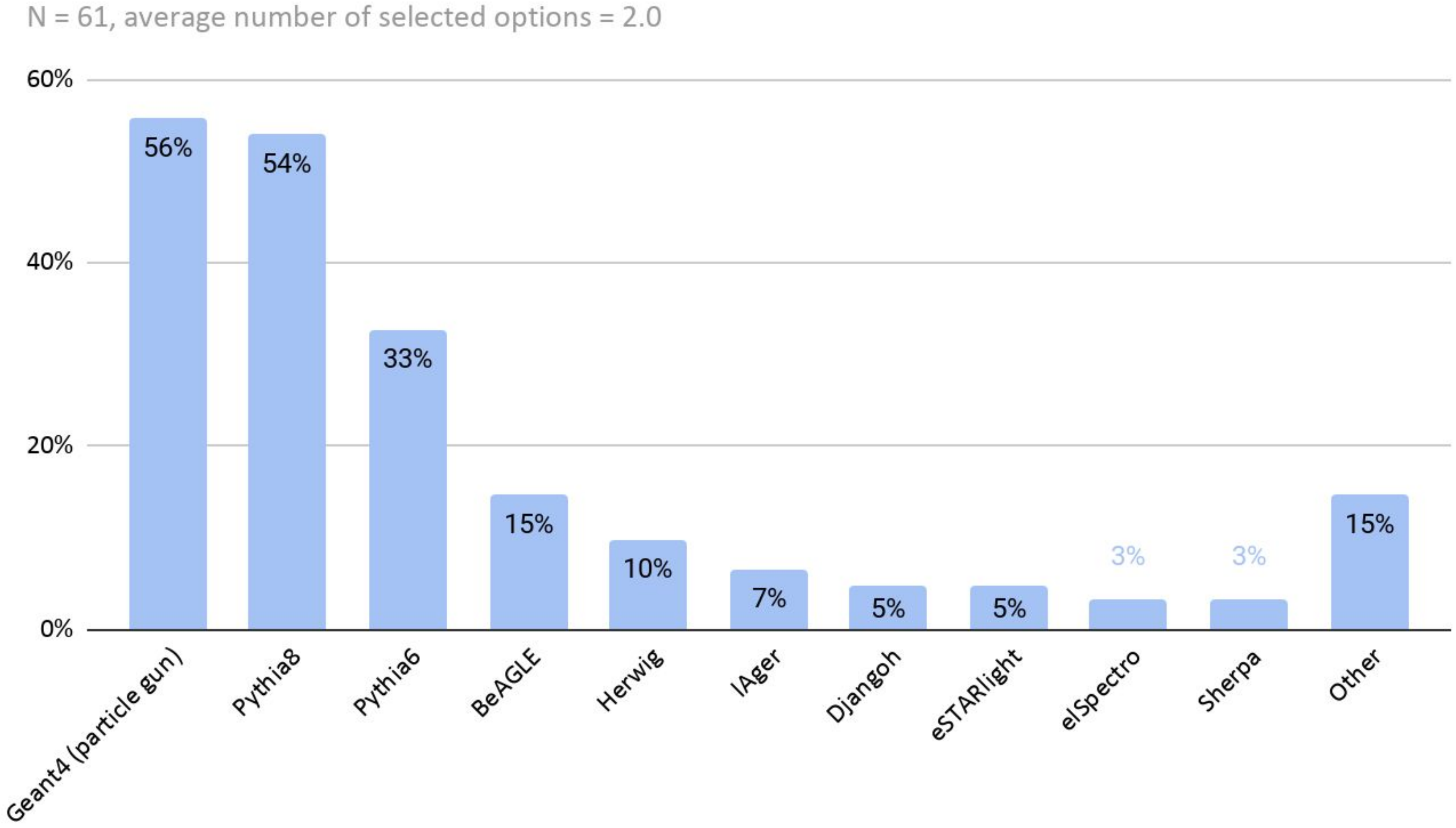


Other (N = 1): semi-retired senior researcher

Feedback from Expression of Interests (link)
Contributions from Ph.D. students will increase over time.

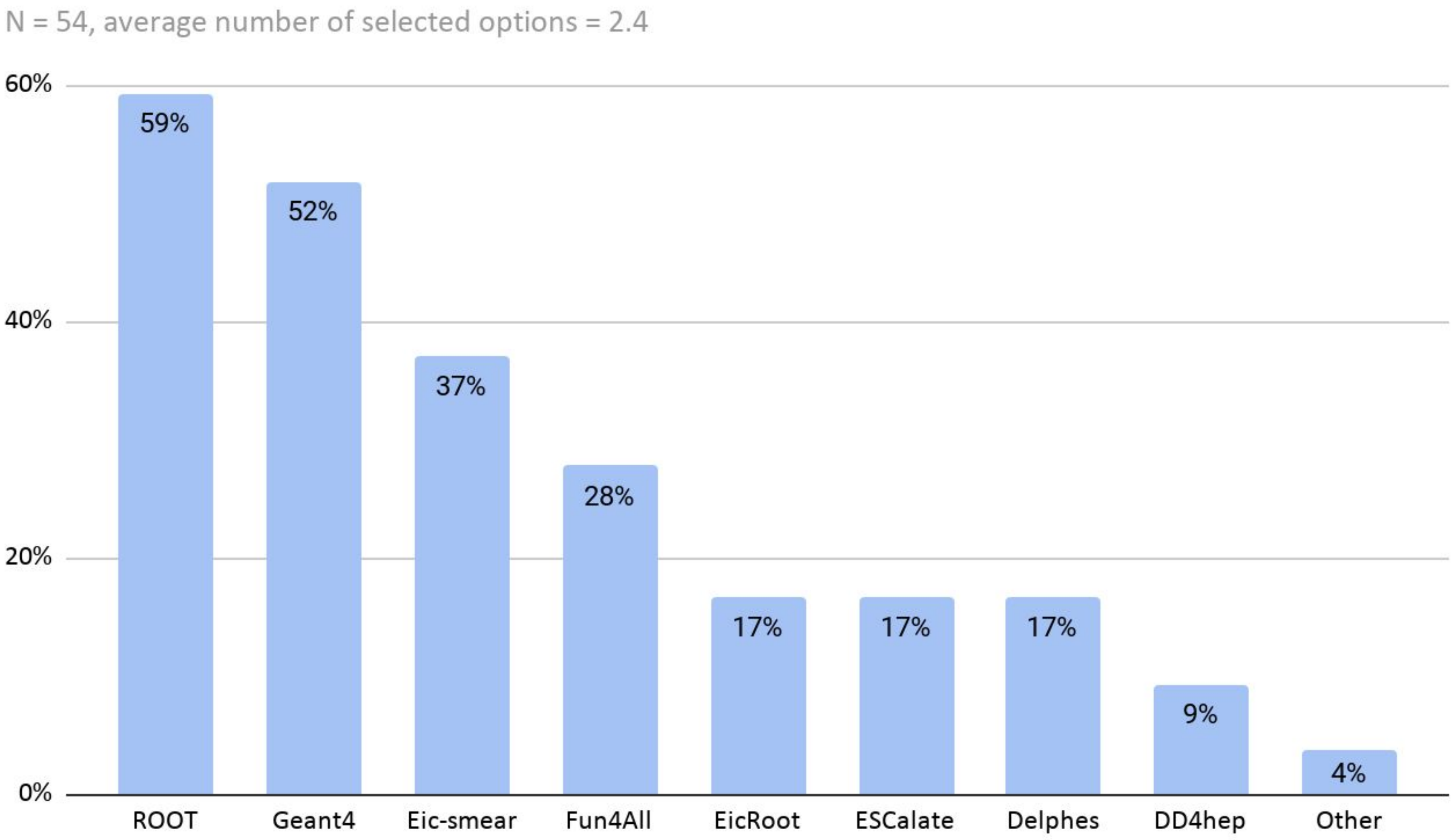


Over the past year, which physics event generation tools did you use for EIC simulations?



Other (N = 9): personal computer codes (N = 2), ACT, CLASDIS, ComptonRad, GRAPE-DILEPTON, MADX, MILOU, OPERA, RAYTRACE, Sartre, Topeg, ZGOUBI

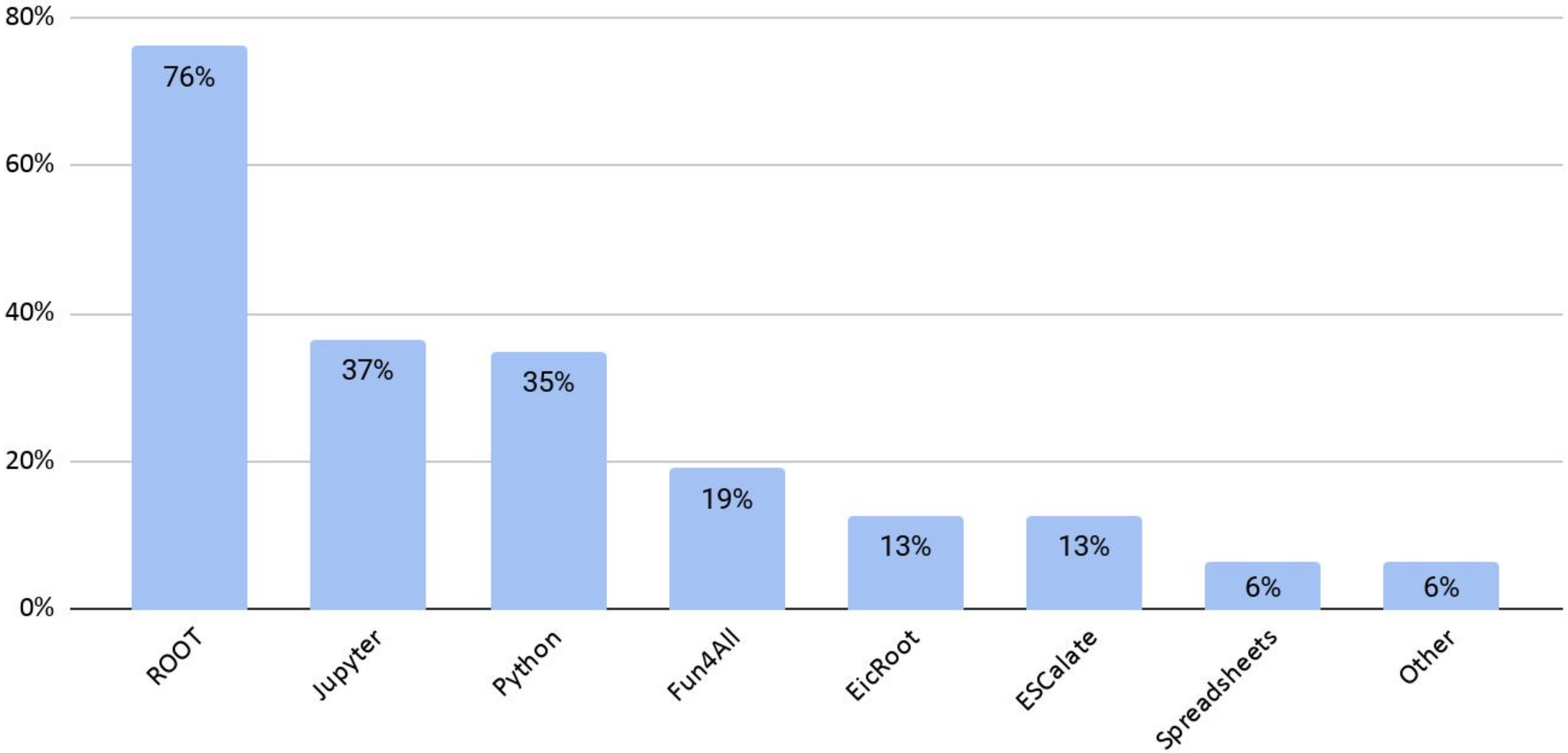
Over the past year, which detector simulation tools did you use for EIC simulations?



Other (N = 2): GEMC, RAYTRACE

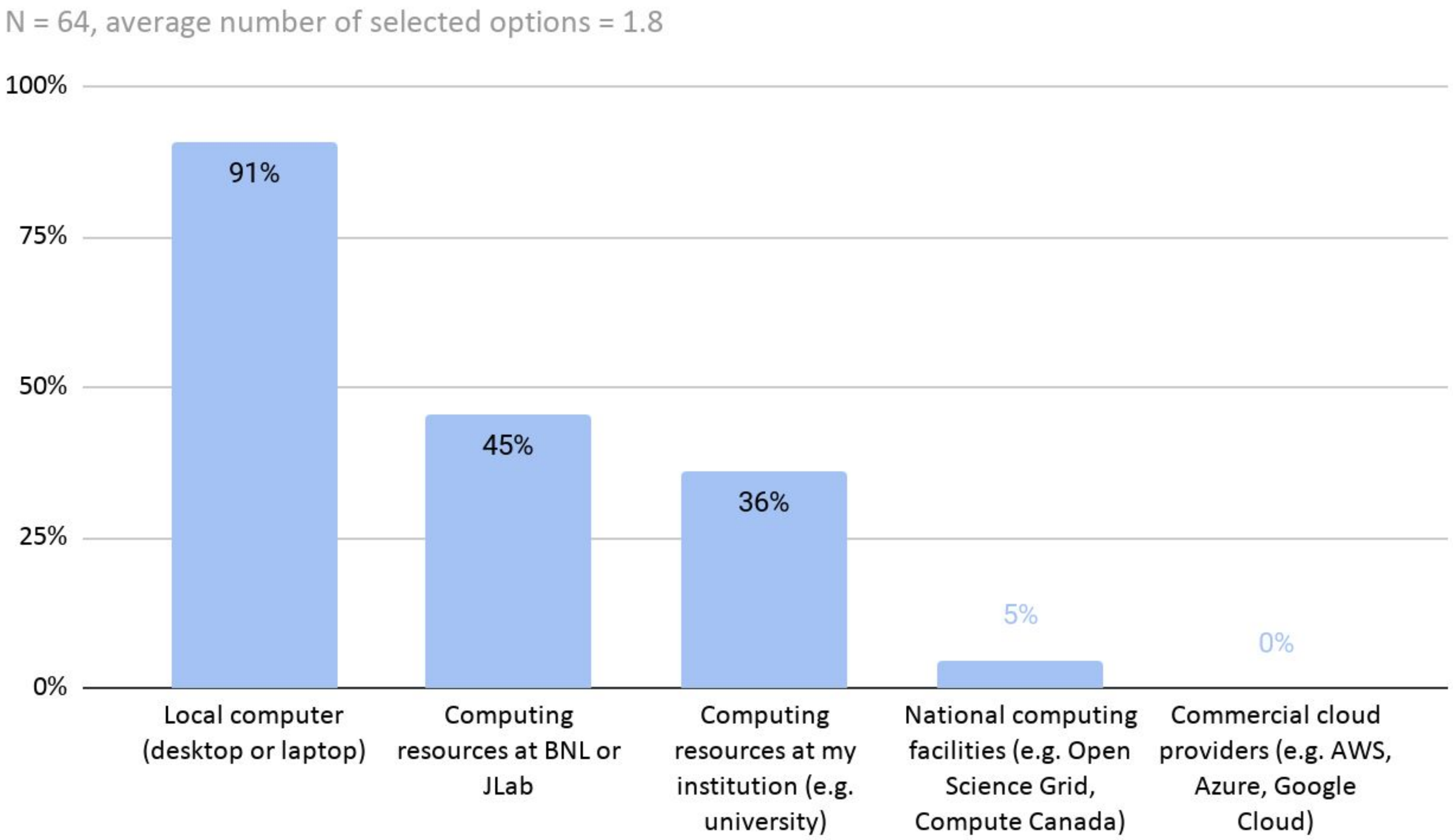
Over the past year, which analysis tool(s) did you use for EIC simulations?

N = 63, average number of selelected options = 2.1

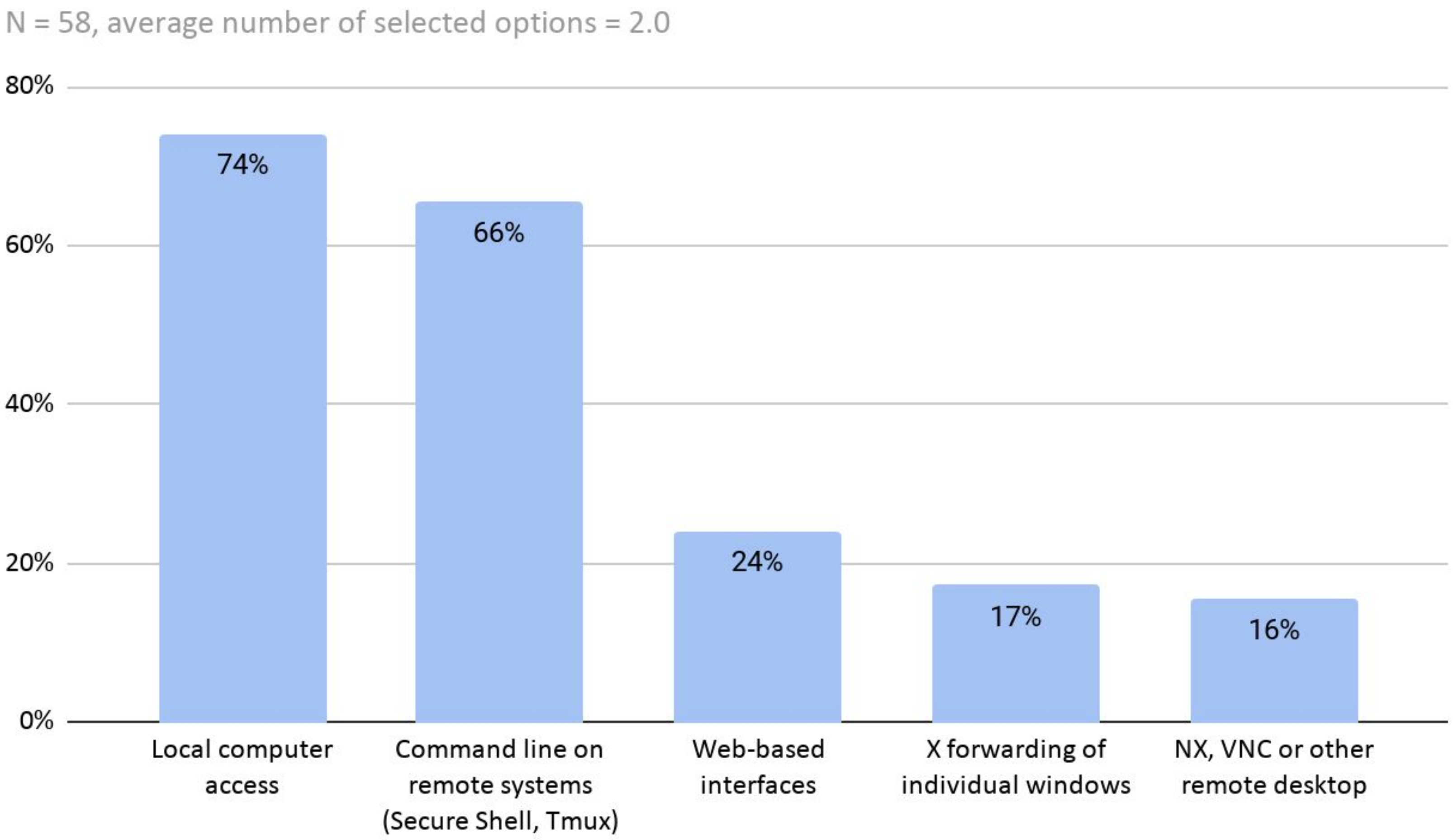


Other (N = 4): Rivet, ACE3P, jas4pp, custom codes

Over the past year, which resources did you use for EIC simulation and analysis?



Over the past year, how did you access the computing resources for EIC analysis?



Do you have any comments on your current experience with EIC Software?

N = 9

There are too many generators and simulation tools used at the moment. **5 x**

- Lack of documentation.
- More tutorials would be beneficial.

3 x

The group should focus on full Geant4 simulation. **1 x**

ELECTRON ION COLLIDER USER GROUP

2. USER FOCUS GROUPS

Notes:

- Selection methodology: Volunteered during focus groups or personal contacts.
- This year’s data collection based on career stage has nearly been completed.
- Future focus groups may be targeted based on software survey responses.

First Round of Focus Groups

Grouping Criterion: Career Stage

- Graduate students (n = 2)
- Junior postdocs (n = 4)
- Senior postdocs (n = 3)
- Staff scientists (n = 5)
- Professors (n = 5, to be scheduled)
- Physicists in industry (n = 10, to be scheduled)

Approach: Prompted Discussion, 1 hour or longer

- What software are you currently using, and why?
- Are you able and do you feel comfortable performing the computational aspects?
- What software or computing barriers do you encounter in your research?
- What do you think stopping physicists from participating in simulation or analysis?
- Which software tools do you particularly enjoy using, even if not EIC related?

Towards Personas and User Profiles: Attributes

Attributes of users:

- Low vs. high experience with physics computing
- Self-identification as user vs. developer
- Desire for guidance vs. self-starter mentality
- Need for custom software vs. availability of off-the-shelf functionality
- Positive vs. negative attitudes towards the process of writing software
- Positive vs. negative attitudes towards other users of the software
- Emotional or career investment in software
- Low vs. high ability to influence community, through positional power or power of expertise

Connection with general personality traits:

- Openness to the risk of new experiences: conservative vs. creative
- Conscientiousness: quick-and-dirty hack jobs vs. elaborate architectures
- Ability to compromise: autonomous vs. cooperative development

Towards Personas and User Profiles: Methodology

We score all participants on each attribute, normalize, then use k-means clustering and principal component analysis to identify the groups and distinguishing directions in attribute space.

OR

We follow a qualitative approach where similar statements by multiple participants and recurring quotes are foregrounded, and where we attempt to identify common themes across focus groups.

?

Towards Personas and User Profiles

Along one set of axes (~experience, ~attitude towards writing software):

- Starting Scientist
  - new to the field, don’t know how to get information, dependent on others
- Starting Scientist with CS/CE Experience
  - new to the field, but with programming experience, dependent on others
- Software Using Scientist I
  - not interested in programming but competent, wishes for more documentation
- Software Using Scientist II
  - likes programming, contributes documentation to projects
- Software Developing Scientist I
  - active developers of large projects or frameworks
- Software Developing Scientist II
  - high-level perspective from experience
- Software Project Owning Scientist
  - in charge of an entire software project

Towards Personas and User Profiles

Along another set of axes (~attitudes towards users):

- Software as a necessary tool: “I like software, to the extent that it helps me get physics done. Give me a good example, and I’ll use your software.”
- Software is not my strong suit: “I am a bad programmer. I know. I write crappy code. Don’t force me to share my code with others. I am ashamed of it.”
- Software as part of my research: “I use software tools for my research project. I feel comfortable in using the software and modifying it for my needs. I share my modifications but software development is not my priority.”
- Software is a social activity: “I like to write software with and for others. I know I can write software pretty well, and I want to help people who don’t like it.”
- Software emperors: “I write the best software. I know how physics software should be written. Just follow my software-imposed rules.”

Towards Personas and User Profiles: Comments

There is still a reluctance to having analysis live in jupyter notebooks.

Don’t tell me to sign up for a mailing list: send an invite around that I can click on.

Being in multiple experiments makes it hard to get proficient in the software tools used in each.

I don’t use git. No one is going to use my code anyway.

Maybe we need dedicated people who only write documentation.

Installing software can be frustrating due to obscure dependencies.

Tutorials seem aimed at showing off software, not really teaching how to do something.

Lost lot of time due to lack of documentation, out-of-date documentation.

Debugging code in containers often was laborious.

More than 70% of the time I have to go through the source code to figure it out.

If I am stuck, I send an email or ask on Mattermost.

I don’t like to bother the main developer and prefer asking a peer first.

ELECTRON ION COLLIDER USER GROUP
USER SURVEY AND FOCUS GROUPS

Thank you to everyone who participated in the EICUG State of Software survey and Focus Groups. The Software Working Group will repeat the survey at the end of 2021 to compare results as we continue to design and build the Electron-Ion Collider.

Next steps

- We will organize further focus group discussions that will result in personas and user stories.
- The user stories will provide input to software developers as to which users they are writing software for.

Appendix: Survey Questions

Q1. What is your current role in the EIC project?

○ Undergraduate student
○ Graduate student (M.Sc.)
○ Graduate student (Ph.D.)
○ Postdoctoral researcher
○ Early-career scientist (pre-tenure, assistant professor, staff scientist < 5 years)
○ Mid-career scientist (tenure, associate professor, staff scientist 5-10 years)
○ Senior scientist (full professor, staff scientist > 10 years)
○ Other (please specify)

____

Q2. Over the past year, which physics event generation tool(s) did you use for EIC simulations? Check all that apply.

□ Geant4 (particle gun)
□ BeAGLE
□ Djangoh
□ eSTARlight
□ Herwig
□ lAger
□ Pythia8
□ Pythia6
□ Sartre
□ Other (please specify)

____

Q3. Over the past year, which detector simulation tool(s) did you use for EIC simulations? Check all that apply.

□ ROOT
□ Geant4
□ DD4hep
□ Delphes
□ Eic-smear
□ EicRoot
□ ESCalate
□ Fun4All
□ Other (please specify)

Q4. Over the past year, which analysis tool(s) did you use for EIC simulations? Check all that apply.

□ ROOT
□ EicRoot
□ ESCalate
□ Fun4All
□ Jupyter
□ Python (NumPy/Pandas/...)
□ Spreadsheets
□ Other (please specify)

____

Q5. Over the past year, which resources did you use for EIC simulation and analysis? Check all that apply.

□ Local computer (desktop or laptop)
□ Computing resources at my institution (e.g. university)
□ Computing resources at BNL or JLab
□ National computing facilities (e.g. Open Science Grid, Compute Canada)
□ Commercial cloud providers (e.g. AWS, Azure, Google Cloud)

____

Q6. Over the past year, how did you access the computing resources for EIC analysis? Check all that apply.

□ Local computer access
□ Command line on remote systems (Secure Shell, Tmux)
□ X forwarding of individual windows
□ NX, VNC or other remote desktop
□ Web-based interfaces

____

Q7. Do you have any comments on your current experience with EIC Software?

____

Q8. Are you interested in volunteering for future focus group discussions on EIC Software? If so, please enter your email address.