



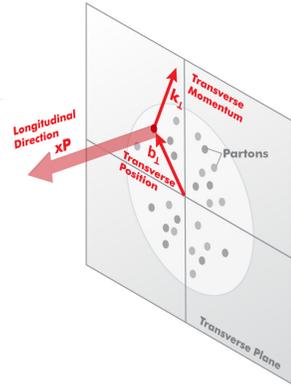
---

EIC SOFTWARE CONSORTIUM

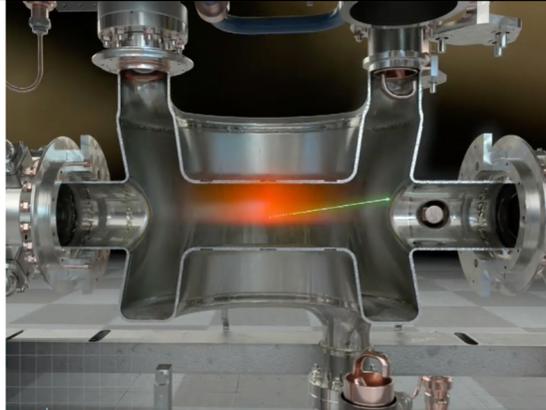
Update in EIC Generic Detector R&D meeting on July 27, 2020

# Scientific advances a new frontier in Nuclear Physics

## Quantum Chromodynamics



## Accelerator technologies



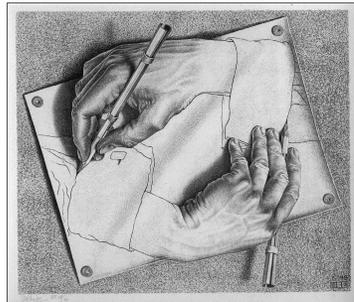
**Electron-Ion Collider:** Steady advances in all of these areas will enable precision study of the nucleon and the nucleus at the scale of sea quarks and gluons.

## Detector technologies

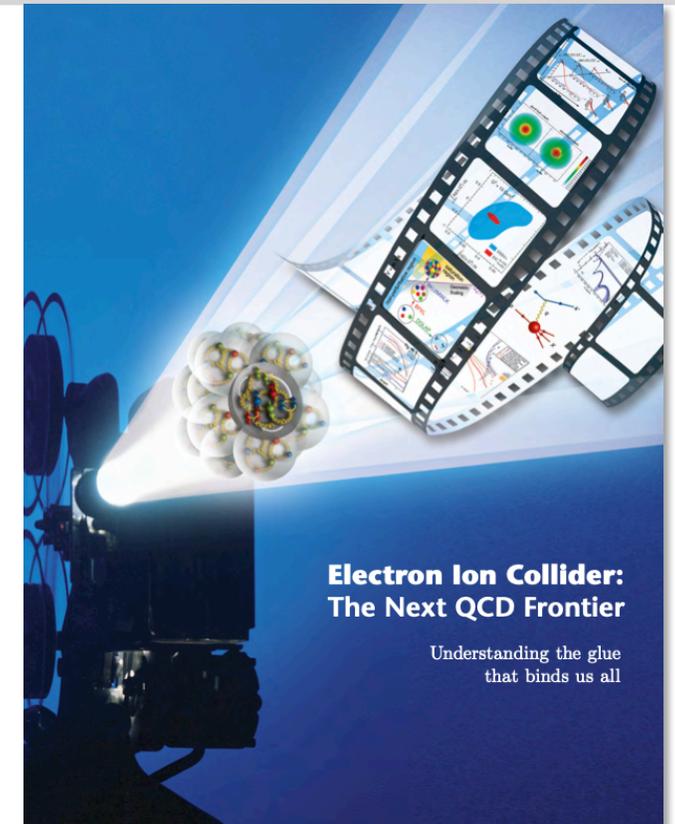
### Electron-Ion Collider Detector Requirements and R&D Handbook

DRAFT 8 - December 15, 2018

Editors:  
Alexander Kiselev  
Thomas Ullrich



## Software & Computing



# The EIC Software Consortium

September 2015

EIC Software Meeting [link](#)

Review of existing EIC software frameworks and MCEG available for the EIC

January 2016

LOI for Software Consortium

*Endorsement “A robust software environment, compatible with the existing software frameworks, is very important for the development of the physics case for the EIC.”*

July 2016

Proposal for Software Consortium

October 2016 – September 2020

EIC Software Consortium supported by EIC Generic R&D program



EIC Software Meeting at ANL

# Goals

---

## **Interfaces and integration**

- connect existing frameworks / toolkits
- identify the key pieces for a future EIC toolkit
- collaborate with other R&D consortia

## **Planning for the future with future compatibility**

- workshop to discuss new scientific computing developments and trends
- incorporating new standards
- validating our tools on new computing infrastructure

## **Organizational efforts with an emphasis on communication**

- build an active working group and foster collaboration
- documentation about available software
- maintaining a software repository
- workshop organization

# Development of the EIC Software Consortium

---

June 2018

## Working with the EIC User Group (EICUG)

**Announcement of EICUG Software Working Group** *“The working group will build on the **considerable progress made within the EIC Software Consortium (eRD20)** and other efforts.”*

September 2019

## EICUG Yellow Report Initiative

**Announcement of EIC Physics and Detector Conceptual Development** *“These studies should use, in so far as possible, the current accelerator and detector concepts and **simulations should be carried out using the EICUG developed software tools.**”*

# EICUG Software Working Group

84 members

Convener

**A. Bressan (Trieste)**

**M. Diefenthaler (JLAB)**

**T. Wenaus (BNL)**

Mailing list [eicug-software@eicug.org](mailto:eicug-software@eicug.org)

subscribe via Google Group

Repository <https://github.com/eic>

Website <http://www.eicug.org/web/content/eic-software>

Core Group



J. Adam (BNL)



M. Asai (SLAC)



N. Brei (JLAB)



A. Bressan (Trieste)



W. Deconinck (Manitoba)



M. Diefenthaler (JLAB)

EIC SOFTWARE CONSORTIUM



J. Furletova (JLAB)



Picture

S. Furletov (JLAB)



S. Joosten (ANL)



K. Kauder (BNL)



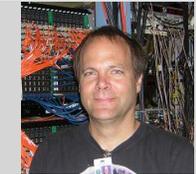
A. Kiselev (BNL)



J. Lauret (BNL)



D. Lawrence (JLAB)



C. Pinkenburg (BNL)



M. Potekhin (BNL)



D. Romanov (JLAB)



M. Ungaro (JLAB)



T. Wenaus (BNL)

# Role of the Software Working Group

**Develop**

**Support**

## Workflow environment for EIC simulations

- **to use** (tools, documentation, support) **and**
- **to grow with user input** (direction, documentation, tools)



## Involvement from EICUG

- **Coordinate simulations** with EICUG Detector and Physics Working Groups.
- Rely on expertise of EICUG:
  - **Detector design**
  - **Reconstruction algorithms**
  - **Physics analysis**

# Recent focus: Get EICUG started on simulation tools

Full simulations

EicRoot

ESCalate

Fun4All

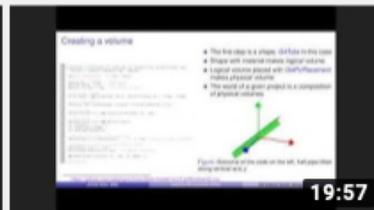
Fast simulations

eic-smear

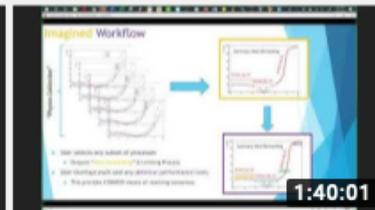
## Online tutorials on YouTube channel of EICUG



EIC Software Group: An Introduction (01/09/2020)



EIC Software Tutorial: Example Detector...



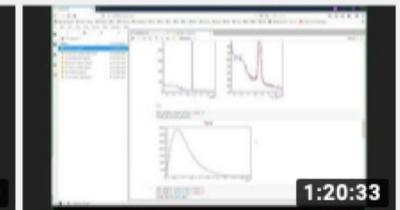
EICUG Software Working Group: Greenfield



EIC Software Tutorial: Detector Full Simulations in...

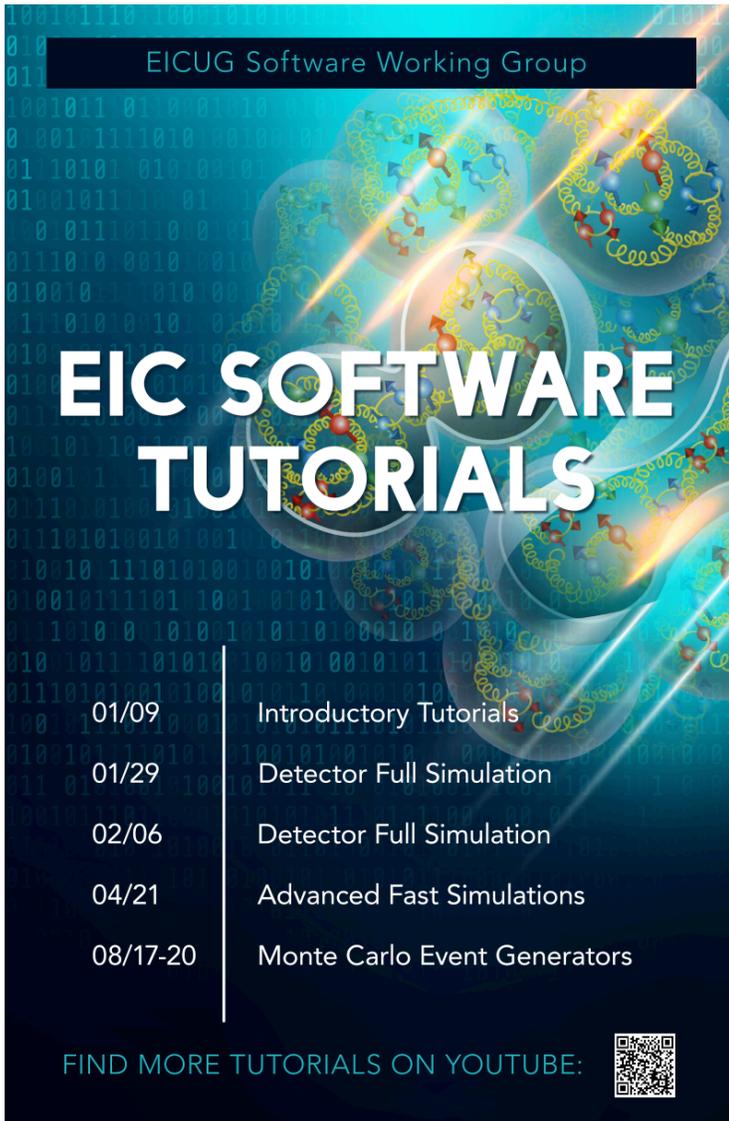


EIC Software Tutorial: Detector Full Simulations in...



EIC Software Tutorial: Fast Simulations (01/09/2020)

# Next steps

A vertical poster for EIC Software Tutorials. At the top, it says 'EICUG Software Working Group'. The main title is 'EIC SOFTWARE TUTORIALS' in large white letters. Below the title is a list of dates and topics: 01/09 Introductory Tutorials, 01/29 Detector Full Simulation, 02/06 Detector Full Simulation, 04/21 Advanced Fast Simulations, and 08/17-20 Monte Carlo Event Generators. At the bottom, it says 'FIND MORE TUTORIALS ON YOUTUBE:' followed by a QR code. The background features a blue and green abstract design with binary code and particle tracks.

EICUG Software Working Group

## EIC SOFTWARE TUTORIALS

01/09	Introductory Tutorials
01/29	Detector Full Simulation
02/06	Detector Full Simulation
04/21	Advanced Fast Simulations
08/17-20	Monte Carlo Event Generators

FIND MORE TUTORIALS ON YOUTUBE: 

## Introduce modern MCEGs to EIC community

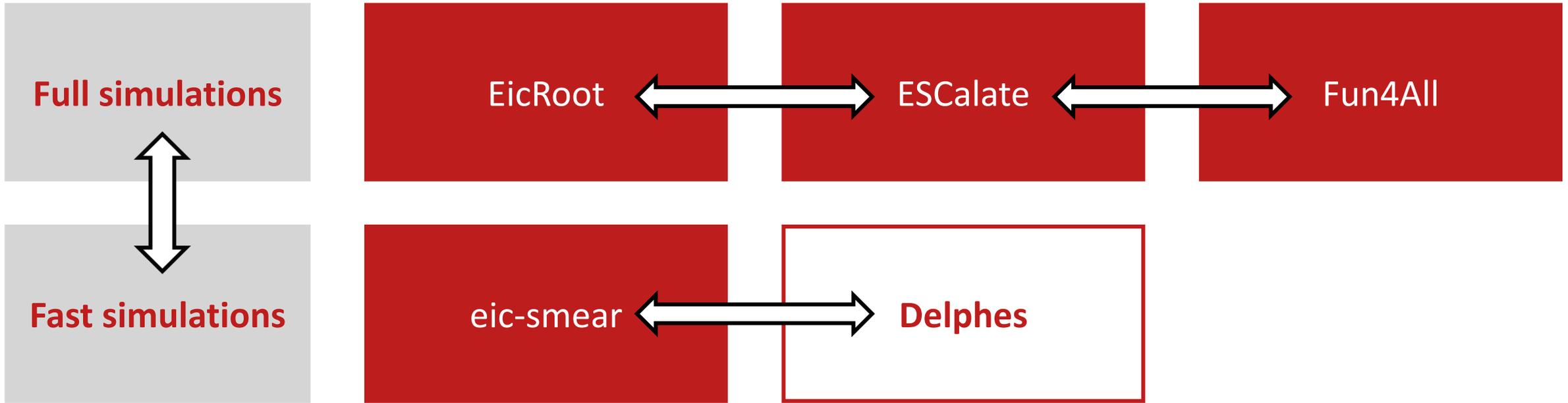
- Integration in EIC simulation software ongoing
- Tutorials:

08/17	<b>Pythia 8</b> Stefan Prestel (LUND)
08/18	<b>Rivet</b> Christian Bierlich (LUND)
08/19	<b>Herwig 7</b> Simon Plätzer (Vienna)
08/20	<b>Sherpa</b> Stefan Hoeche (FNAL)

## Validation of modern MCEGs with DIS data

- **HERA H1, ZEUS, HERMES**
- **COMPASS**

# Current focus: Benchmarks & validation



## Cross-tool validation

- **Unified format** for the output ROOT tree of all tools (work in progress)
- Collect simulation configuration and physics analyses (work in progress)
  - Any study that is shared with the SWG can be used for **benchmark & validation**.
  - Based on analysis scripts and macros for the given study, the SWG can reproduce the results and build up a **validation scheme** on top of it.



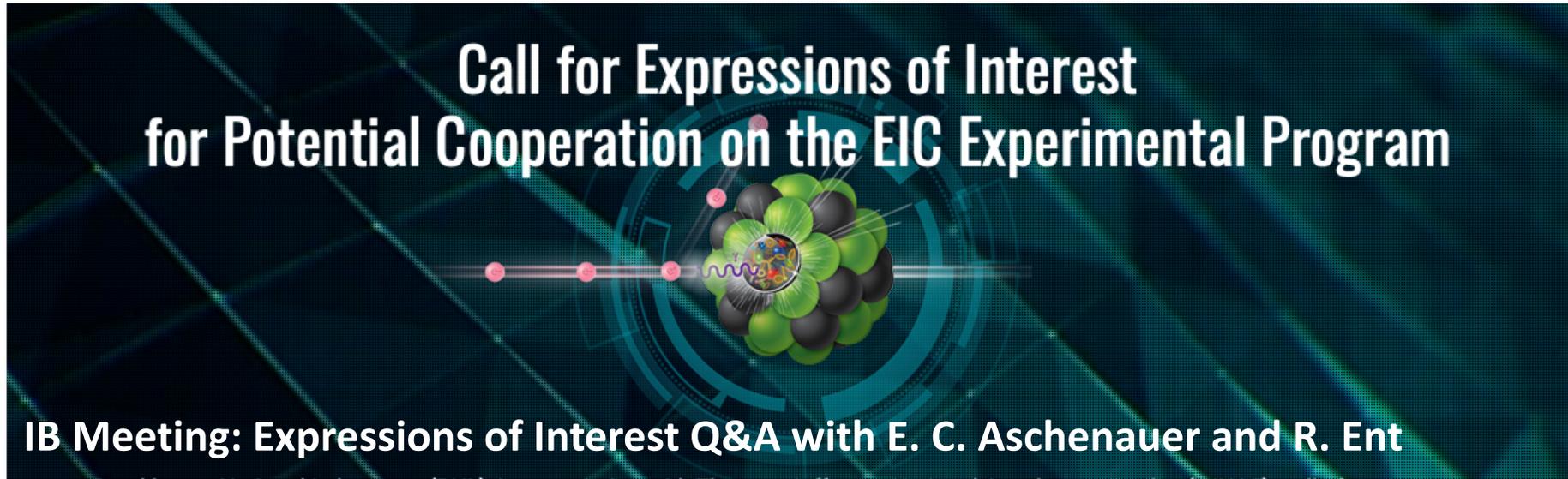
## ELECTRON ION COLLIDER USER GROUP SOFTWARE NEWS

The Software Working Group is working on physics and detector simulations that enable a quantitative assessment of the measurement capabilities of the EIC detector(s) and their physics impact for the Yellow Report Initiative. The common simulation tools and workflow environment being set up by the working group allows the EICUG to pursue the Yellow Report studies in a manner that is accessible, consistent, and reproducible.

### Table of contents

- [General Update](#)
  - [Communication](#)
  - [Detector Working Group: Detector Matrix Version 0.1](#)
  - [GitHub for the EICUG](#)
  - [Petrel: Worldwide data storage and sharing solution](#)
  - [Support us to support you better](#)
  - [Tutorials](#)
- [Software Update](#)
  - [EicRoot](#)
  - [eic-smear](#)
  - [ESCalate](#)
  - [Fun4All](#)

# Expressions of Interest (EOI)



Call for Expressions of Interest  
for Potential Cooperation on the EIC Experimental Program

IB Meeting: Expressions of Interest Q&A with E. C. Aschenauer and R. Ent

The image is a promotional banner with a dark teal background featuring a grid of glowing lines. In the center, there is a stylized molecular or atomic structure with green and black spheres and a central globe. The text is white and bold, with the main title at the top and the meeting information at the bottom.

## EOI FAQ: Can institutions submit more than one EOI?

Institutions may very well consist of different groups that have more than one interest. However, individual institutions should not submit more than one EOI, and that EOI should be all-inclusive. **Institutions may also join other groups and/or consortia in their EOI, with cross-referencing to the individual institution EOI.**

## Consortium: Software

EIC collaborations will determine for themselves what they do for software, but that will likely include **common software.**

# Expression of Interest for EIC Software

- All scientists of all levels, worldwide, should be enabled to actively participate in EIC simulations and analyses.
- To achieve this goal, we must develop simulation and analysis software using modern and advanced technologies while hiding that complexity (user-centered design).
- Input by the wider scientific and software communities is encouraged.

## Project Greenfield: Community-wide effort leveraging everyone's experience

- define requirements for EIC Software
  - **Software components** Simulation, reconstruction, physics analyses, streaming readout, online monitoring etc.
- work together on common software projects based on these requirements
  - **Examples for common software** DD4hep, Geant4, ACTS, Gaudi, JANA2, ROOT, Jupyter
  - **Emerging technologies** Artificial Intelligence and Quantum Computing

# Timeline for Expression of Interest for EIC Software

**July 15**

**Announcement in EIC User Group Meeting**

**Mid of August**

**Project Greenfield meeting** Explore common software goals we can work together as a group.

**Date** Based on your availability **Time** 10:00 a.m. – 1:00 p.m. (EDT)

**September –  
October**

**Cooperatively edit Expression of Interest**

**November 1**

**Deadline for submission**

# Next steps

---

For questions or input: [eicug-software-conveners@eicug.org](mailto:eicug-software-conveners@eicug.org)

Add your availability on: <https://doodle.com/poll/wfzsfytfayayebfk>

Announcement of meeting: July 31

# EIC Software Consortium in the last years

## In-person meetings

1. BNL (Oct. 17 2016)
2. BNL (Feb. 8 – 10 2017)
3. JLAB (May 1 – 2 2017)
4. SLAC (Jul. 6 – 7 2017)
5. ANL (Oct. 16 – 17 2017)
6. William & Mary (May 17 2018)
7. Trieste (May 20 – 21 2019)

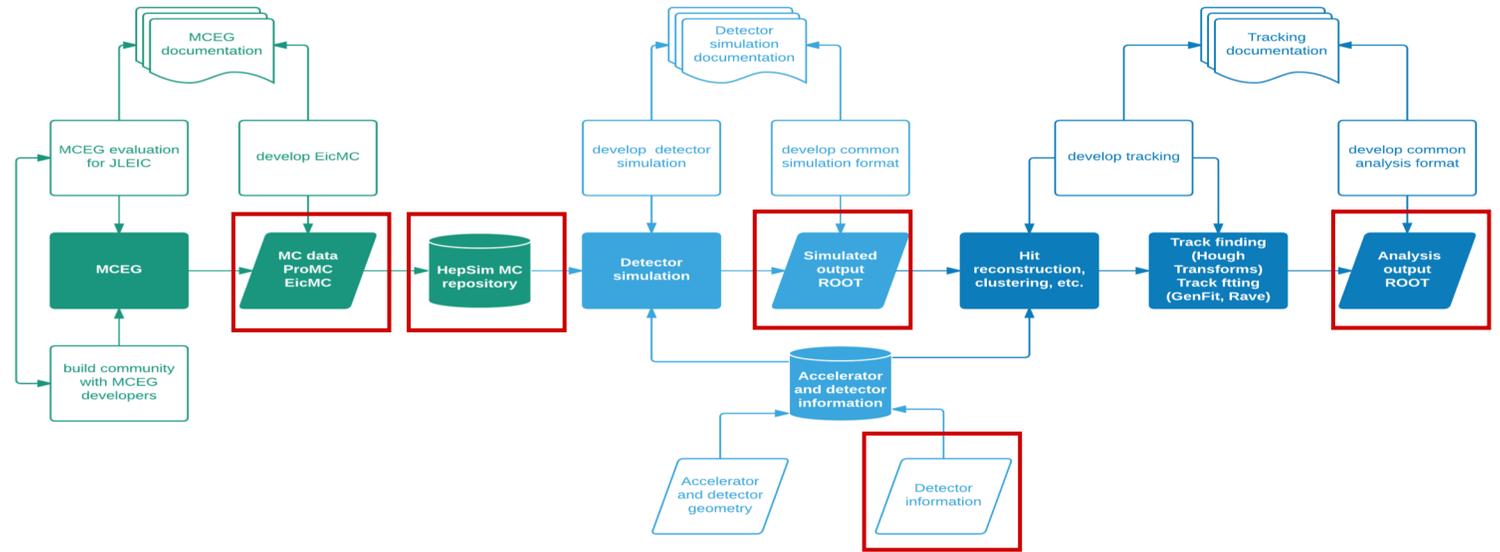


**EIC Software Meeting in Trieste**

# Common interfaces

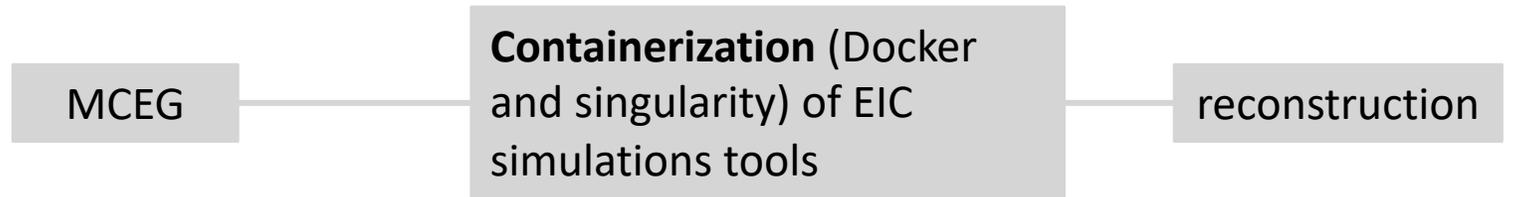
## Advice from ILC effort

- facilitate interoperability
- focus on exchange detector designs and data
  - get the event data model right and keep it open
  - pick a detector definition which is exchangeable



## Norman Graf (SLAC)

*“It’s very difficult to herd cats keep physicists from re-inventing the wheel and writing new software packages.”*



# Geant4 infrastructure for EICUG

## Requirements

- **EIC Generic Detector R&D program (T. Ullrich)** “*a simple lite setup with a well defined geometry description standard that is easy to use*”
- **EICUG** Flexible accelerator and detector interface with full support of existing IR designs and detector concepts

## Approach

- common repository for detector R&D for EIC
- common detector description in Geant4 (C++) and not yet DD4hep (sub-detectors developed in Geant4 (C++))
- common detector naming convention for EIC
- possible common hits output structure
- tutorials on how to implement and integrate subdetectors (in Geant4) in EIC detector concepts

## Discussion

- **two in-person meetings**
  - 07/10 EIC Software Meeting at BNL ([minutes](#))
  - 09/24 EIC Software Meeting at JLAB ([minutes](#))
- **evaluation** 09/30, 10/21, 10/28, 11/18, 11/25

## Two solutions proposed

1. detector simulations in **ESCalate**
2. detector simulations in **fun4all**

**EicRoot** used in Yellow Report Initiative, in particular for tracking studies

# Consensus-based documents

## ESC Container Guidelines

Edited by D. Blyth, W. Deconinck, M. Diefenthaler, A. Dotti, A. Kiselev, D. Lawrence

### Introduction

This document provides guidelines for producing container images that provide EIC software. The primary goal of the images is to provide an easy means for scientists to start running EIC software. Specifically targeted are persons with limited time to invest in building the software and all of its prerequisites. The guidelines will help ensure that software developed at different sites (ANL, BNL, JLab) is packaged in a similar way simplifying cross comparisons and will evolve over time to reflect changes in the EIC software or in the requirements of the EIC community.

### Goals

- Make it easier for users to work on the physics program and detector design for the EIC, e.g., by minimizing the "installation overhead" for new users.
- Allow EIC users to run the EIC software interactively on any computer (personal desktop or laptop) through the use of containers; one should be able to run the same container image under any modern 64-bit Linux, Mac OS, or MS Windows system.
- Provide consistency between software generated at different facilities.

### Technology

- Docker generates initial image; all the content should go into Dockerfile and associated input files if needed.
- Consideration should be given to repeatability of builds from the Dockerfile.
- Consideration should be given to using the image with Singularity or Shifter by simple import from Docker (not be worked out in detail for the first versions).
- Consideration should be given to how the image may eventually be used as an appliance (also not be worked out in detail for the first versions).

### EIC computing environment

Different groups within the ESC are developing software stacks with pieces that are independent from other groups. Some of the development environments are incompatible making it impractical at this time to impose a single OS/compiler. This document describes a set of software tools common to the field and directory structure to be used in the container. This common format should make it easier for EIC users to use any of the containers once they have familiarized themselves with one.

### Software

Images will contain at a minimum:

- Development tools cmake3, Geant4, git, gcc, make, python
- HepSim support Google protocol buffer

In addition, the following packages are recommended:

- Analysis tools CLHEP, ROOT6
- [Graphical environment](#)

EIC Software Consortium

## Geometry Description and detector interface

### Abstract

This document summarizes a possible path forward for the geometry description for the simulations of EIC detectors. It contains the list of what we believe should be the requirements for a EIC geometry description system. The considerations in this document are probably general enough and can be applied to any geometry system independently on the specific technology choice, the focus however is on the I/O of geometry and on the link to sensitivity information, since these two aspects have been the focus of our discussions in ESC meetings in FY2017 .

### Initial considerations

It is safe to assume that in the time-scale for which detector simulations for EIC are needed Geant4 will continue to be the de-facto standard for detector simulations, we should thus consider the paradigms implemented in Geant4 (e.g. hierarchical geometry, concepts of sensitive detectors and hits) as general guidelines for our future works.

There are two main use-cases that drive the development of a geometry module: simulation and reconstructions. It is an obvious requirement that the same geometry description should be used between the two subsystems. How to implement this paradigm is mainly left to the specific choices of experiments, and currently no real detector-independent framework has emerged so far as a widely adopted standard. Many projects have tried to propose such frameworks (among the one mentioned in our meetings are SLIC and DD4hep) with somewhat limited success. It is very important to stress that we do not believe that the lack of wider adoption is not due to inherent quality of the artifacts (that on the contrary is usually quite high), but since no large experiment has adopted these tools as standard the community behind these tools has remained small and fractionated.

Simulation requires the description of geometry in increased level of complexity: from the simplified ideal detectors used for concept studies, to the full detailed simulations of running experiment. The data reconstruction as a general idea requires a more *conceptual* description of the geometry in terms of read-out elements instead of physical placements. In particular the mapping between sensitive geometry element and hits is of crucial importance.

We identified two possible ways of defining the geometry of a detector for simulation.

### Geometry implementation via code

The first approach, to write code that uses directly geometry primitives, is usually preferred for smaller applications, e.g. the majority of the examples distributed with Geant4 toolkit create the detector geometry in this way. In case of ROOT based frameworks this approach is quite natural, because you can add to this basic scenario some I/O and scripting capabilities (TGeo classes to describe the geometry in a program, ROOT I/O to write geometry elements, and ROOT scripts, that are programs by themselves, to steer the process).

# Foster collaboration

## High Energy Physics

**CERN ROOT**  
In frequent contact

**HEP Software Foundation**  
Knowledge exchange

**MCnet**  
Starting Collaboration

**SLAC Geant4**  
Established collaboration

**Reached out** ACTS, ATLAS, DD4hep, Exascale Project, HDF5, ILC project, PyHEP, Project Jupyter

## Nuclear Physics

**EICUG Software Working Group**

**Same software suite** Seamless data processing from DAQ to data analysis using AI

**EIC Streaming Readout Consortium**  
**EICUG Yellow Report - Readout and DAQ subgroup**

# Collaboration on detector simulations

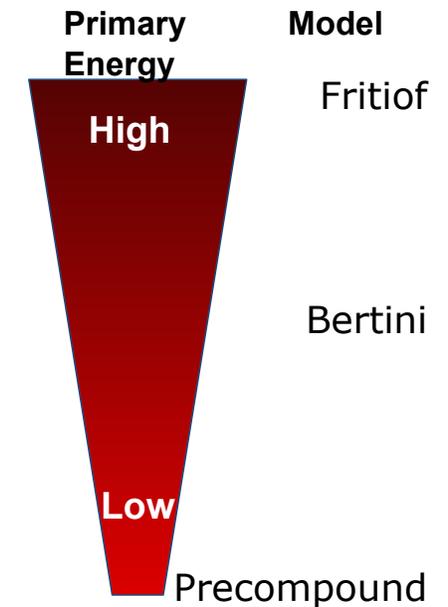
- collaboration with Geant4 International Collaboration
  - **liaison** Makoto Asai (SLAC)
- **Geant4 for EIC**
  - coordinate input for Geant4 validation based on EIC physics list maintained by (former) SLAC Geant4 group
  - Geant4 10.6 recommended (released Dec. 6)

## 09/24/19 Geant4 Technical Forum on EIC

- EIC detector and physics simulations rely on Geant4
- knowledge transfer (e.g., sub-event parallelism or tessellated solids)
- maintain EIC physics lists
- **request** improved photo-nuclear and electro-nuclear reactions

## EIC

- energy range is different from LHC
- validation, tuning and extension including test beam studies



# Collaboration on physics event generation

## Unique MCEG requirements for EIC Science

- MCEG for polarized ep, ed, and eHe<sup>3</sup>
  - including novel QCD phenomena: GPDs, TMDs
- MCEG for eA

## MCEG community

- focus of last two decades: **LHC**
  - **lesson learned** high-precision QCD measurements require high-precision MCEGs
  - MCEG not about tuning but about physics
- ready to work on ep/eA



# Exploring user-centered design

## Simulations on the web

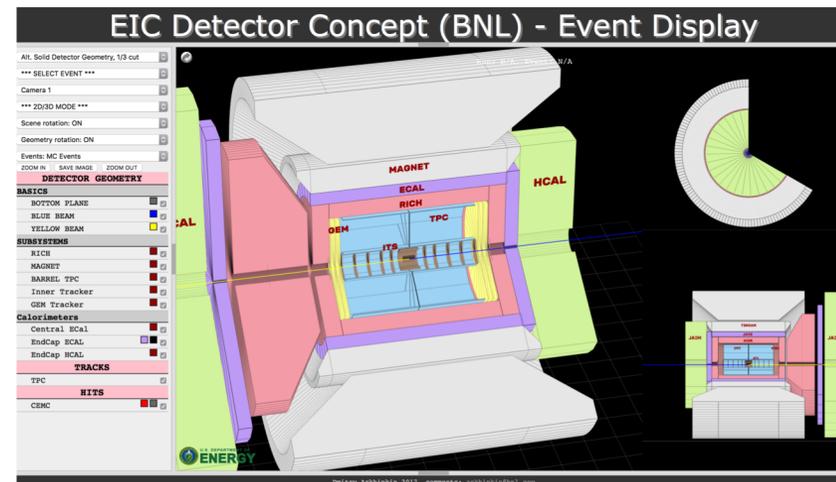
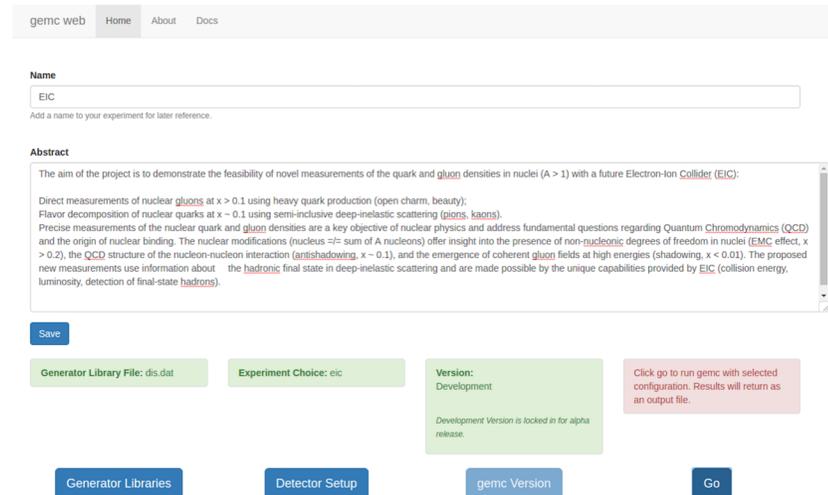


Sam Markelon (UConn)

## Universal web-based event display

- validation of the EIC simulations
- comparison of different detector designs using an unified approach
- public outreach

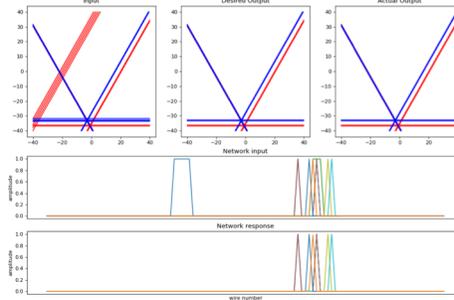
STAR software group effort



# Exploring machine learning for EIC

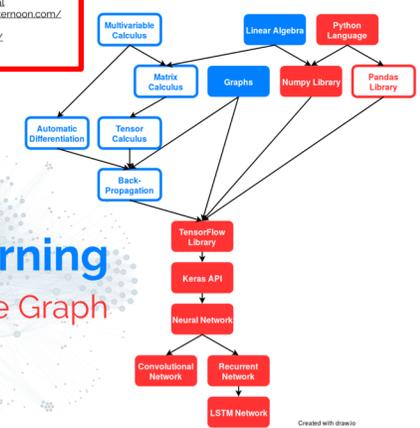


## Track finding in Jlab 12 data

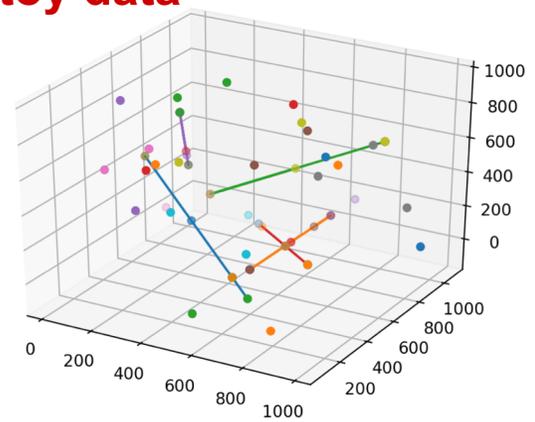


TensorFlow (TF)

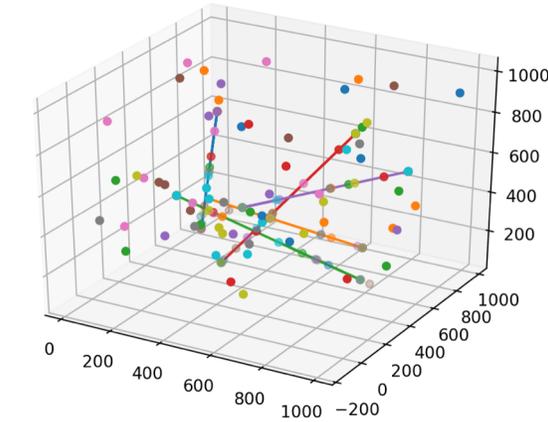
- Repository of Jupyter notebook TF tutorials for beginners: <https://github.com/hvass-Labs/TensorFlow-Tutorials>
- Repository of Jupyter notebook TF tutorials for beginners: <https://github.com/zlnitz/TensorFlow-Tutorials>
- Written tutorial for beginners on the basics of TF: <https://www.datacamp.com/community/tutorials/tensorflow-tutorial>
- Written tutorial for beginners on the basics of TF: <https://hackernoon.com/machine-learning-with-tensorflow-887fdee2b68>
- Paper describing the internal workings of TF: <https://arxiv.org/pdf/1610.01778.pdf>



## Track finding in toy data



using JLab JupyterHub at <https://jupyter.jlab.org/>



Documentation available

# Self-descriptive file formats

---

## Google protocol buffer based

- flexible
- portable
- no external dependencies

## Development history within ESC

- Idea & original version (ProMC) by **S.Chekanov** for HepSim repository
  - limited functionality MC application
- Second version (EicMC) by **A. Kiselev**
  - MC application with several advanced features
- Present development (ProIO) by **D.Blyth**
  - General-purpose format with multi-language support

# EicMC – self-descriptive MC files

by-product of ProMC import incorporation into EicRoot framework

→ compact and portable C++ code (~3k lines) developed by Alexander Kiselev (BNL)

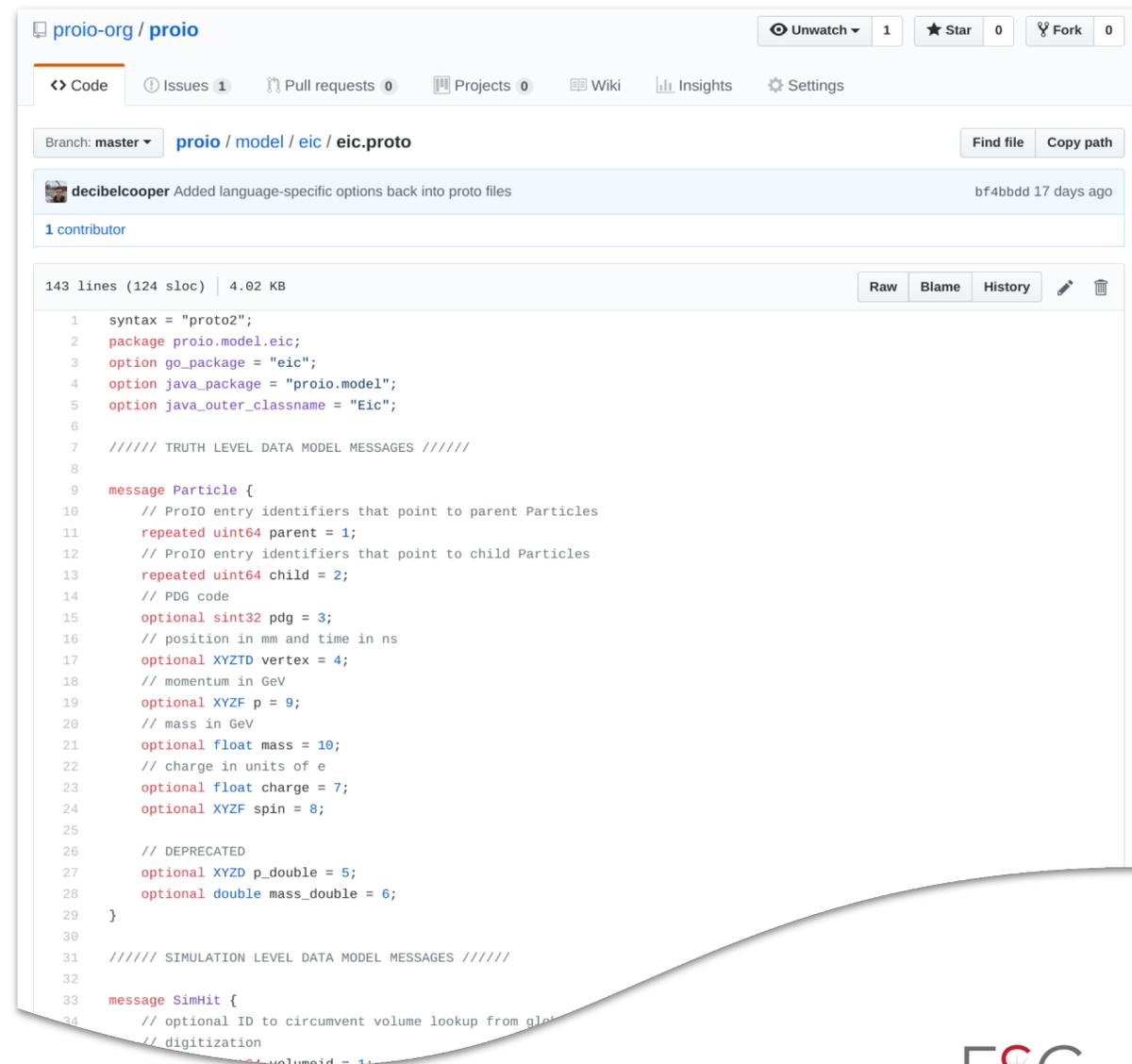
tested with Geant4

**EicMC** feature-rich, flexible, self-descriptive

- no external dependencies on the user (input) side other than **Google Protobuf** library
- unified extendable binary MCEG format (all generators from eic-smear are supported)
- true automatic self-description built into the format core .proto file and the user library
- 64-bit implementation → no 16-/32-bit limitations on file size, record count
- flexible set of compression schemes (for file size, input speed, floating point precision)
- both sequential and direct access to event records (with scalable multi-level catalogues)
- convenient user interface
- performance (file size, speed) similar to or better than ProMC and ROOT equivalents

# ProIO: Support of a collaborative data model

- In contrast to using C++ classes to form a data model, **ProIO** uses Google's Protocol Buffers to implement a **language-independent data model**
- This means that C++, Python, Java, Go, etc. are all first-class.
- Native Python code, e.g., can be `pip` installed in seconds on any OS to enable full read/write support of data (see [py-proio](#))
- ProIO supports multiple compression algorithms, self description, and metadata

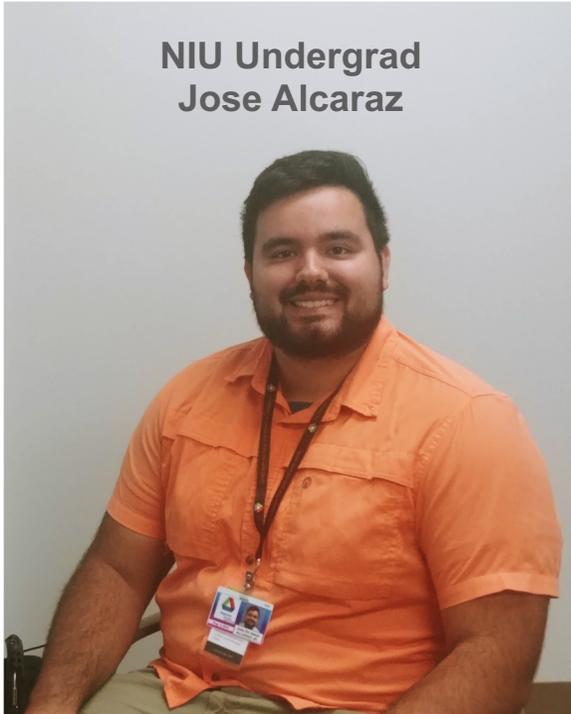


The screenshot shows a GitHub repository for 'proio-org / proio'. The file 'eic.proto' is open, showing Protobuf definitions for a Particle and SimHit. The Particle message includes fields for parent and child identifiers, PDG code, position, momentum, mass, and charge. The SimHit message includes fields for volume lookup and digitization.

```
1 syntax = "proto2";
2 package proio.model.eic;
3 option go_package = "eic";
4 option java_package = "proio.model";
5 option java_outer_classname = "Eic";
6
7 // TRUTH LEVEL DATA MODEL MESSAGES //
8
9 message Particle {
10     // ProIO entry identifiers that point to parent Particles
11     repeated uint64 parent = 1;
12     // ProIO entry identifiers that point to child Particles
13     repeated uint64 child = 2;
14     // PDG code
15     optional sint32 pdg = 3;
16     // position in mm and time in ns
17     optional XYZTD vertex = 4;
18     // momentum in GeV
19     optional XYZF p = 9;
20     // mass in GeV
21     optional float mass = 10;
22     // charge in units of e
23     optional float charge = 7;
24     optional XYZF spin = 8;
25
26     // DEPRECATED
27     optional XYZD p_double = 5;
28     optional double mass_double = 6;
29 }
30
31 // SIMULATION LEVEL DATA MODEL MESSAGES //
32
33 message SimHit {
34     // optional ID to circumvent volume lookup from global
35     // digitization
36     optional uint64 volumeid = 1;
```

# ProIO initiative

NIU Undergrad  
Jose Alcaraz



## Summer research on ProIO usage for EIC

- developing tools for inspecting the content of ProIO streams
- integrating the ProIO C++ library with Pythia8 MCEG
- performing benchmarks of ProIO files

### HepSim

Repository with Monte Carlo simulations for particle physics

Dataset: "gev35ep\_pythia8\_dis1q2ct14lo\_v3"

	Summary	Comments
Name:	gev35ep_pythia8_dis1q2ct14lo_v3	
Collisions:	e-p	
CM Energy:	0.035 TeV	
Entry ID:	325	
Topic:	SM	
Generator:	<a href="#">PYTHIA8</a>	
Calculation level:	LO+PS+hadronisation	
Process:	DIS events at $Q^2 > 1$ GeV <sup>2</sup>	
Total events:	495530	
Number of files:	10	
Cross section ( $\sigma$ ):	$3.350E+05 \pm 805.1965$ pb	Estimated from file
Luminosity (L):	$1.4790$ pb <sup>-1</sup> (or) $0.0015$ fb <sup>-1</sup> (or) $1.479E-06$ ab <sup>-1</sup>	
Format:	ProIO	
Download URL:	<a href="http://mc1.hep.anl.gov/web/hepsim/events/ep/35gev/pythia8_dis1q2ct14lo_v3/">http://mc1.hep.anl.gov/web/hepsim/events/ep/35gev/pythia8_dis1q2ct14lo_v3/</a>	Status: Available
Missing:		

# Exploring JupyterLab environment for EIC

- **bridge to modern data science**, e.g.,



- *Nature* **563**, 145-146 (2018): “Why Jupyter is data scientists’ computational notebook of choice”
- more than three million Jupyter Notebooks publicly available on GitHub

- **collaborative workspace** to create and share Jupyter Notebooks
- **web-based interactive analysis environment** accessible, consistent, reproducible analyses
- **fully extensible and modular** build a collection of analyses and analysis tools

## Jupyter Notebooks

- **writing analysis code**

```
[4]: jana.plugin('hepmc_reader') \
     .plugin('jana', nevents=10000, output='hepmc_sm.root') \
     .plugin('eic_smear', detector='jleic') \
     .plugin('open_charm')

[4]: eJana configured
     plugins: hepmc_reader,eic_smear,open_charm

[5]: jana.source('../data/herwig6_20k.hepmc')

[5]: eJana configured
     plugins: hepmc_reader,eic_smear,open_charm
     sources:
     ../data/herwig6_20k.hepmc

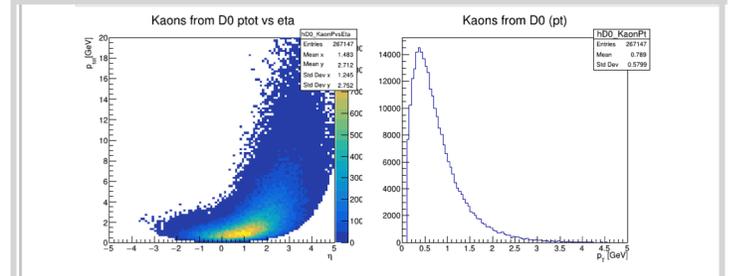
[6]: jana.run()

Total events processed: 10001 (~ 10.0 kevt)
```

Python

Root/C++

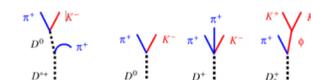
- **visualization of results**



- **narrative of the analysis**

### Open charm

The high luminosity at the EIC would allow measurements of open charm production with much higher rates than at HERA and COMPASS, extending the kinematic coverage to large  $x_B \gg 0.1$  and rare processes such as high- $p_T$  jets. Heavy quark production with electromagnetic probes could for the first time be measured on nuclear targets and used to study the gluonic structure of nuclei and the propagation of heavy quarks through cold nuclear matter with full control of the initial state.



# Future Trends in Nuclear Physics Computing

**FUTURE TRENDS IN  
NUCLEAR PHYSICS  
COMPUTING**

Jefferson Lab, Newport News, VA  
SYMPOSIUM: MAY 2  
WORKSHOP: MAY 3-5

We will examine our hardware and software strategy at a time horizon of ten years. Our goal is to work towards the definition of a common vision for Nuclear Physics (NP) computing and data and recommend future directions for development.

Themes: Resource management • Interplay of I/O, compute and storage • Machine learning for enhancing scientific productivity • Task based approaches • Software portability and reusability • Common infrastructure components

**PROGRAM COMMITTEE:**  
Wes Bethel (JLab)  
Amber Boehnlein (JLab)  
Kyle Cranmer (NYU)  
Markus Diefenthaler (JLab)  
Graham Heyes (JLab)  
Alexander Kiselev (BNL)  
Jerome Lauret (BNL)  
Katherine Riley (ANL)  
Tom Rockwell (FRI/NSCL)  
Torre Wenaus (BNL)

[www.jlab.org/conferences/trends2017](http://www.jlab.org/conferences/trends2017)

**FUTURE TRENDS IN  
NUCLEAR PHYSICS  
COMPUTING**

SYMPOSIUM: MAY 2 • 1:00 p.m.  
Main Auditorium • Free Admission

**NUCLEAR PHYSICS IN A DECADE**  
Donald Geesaman (ANL)

**NUCLEAR PHYSICS COMPUTING IN A DECADE**  
Martin Savage (INT)

**MONTE-CARLO EVENT SIMULATION IN A DECADE**  
Stefan Hoeche (SLAC)

**SYNERGY OF COMPUTING AND THE NEXT GENERATION OF NUCLEAR PHYSICS EXPERIMENTS**  
Rolf Ent (JLAB)

RECEPTION TO FOLLOW

[www.jlab.org/conferences/trends2017](http://www.jlab.org/conferences/trends2017)

**BROOKHAVEN NATIONAL LABORATORY & Jefferson Lab**

**FUTURE TRENDS IN  
NUCLEAR PHYSICS  
COMPUTING**

SEPT. 29 - OCT. 1, 2020

The workshop focuses on the Nuclear Physics Software & Computing community. We will identify what is unique about our community and we will discuss how we can strengthen common efforts and chart a path for Software & Computing in Nuclear Physics for the next ten years.

**TOPICS:**

- Common Scientific Software
- The Role of Data Centers in Scientific Discovery
- Unique Software Challenges for Nuclear Physics

**PROGRAM COMMITTEE:**  
Alexander Kiselev (BNL)  
Amber Boehnlein (JLAB)  
Graham Heyes (JLAB)  
Mark Ito (JLAB)  
Markus Diefenthaler (JLAB)  
Ofer Rind (BNL)  
Paul Laycock (BNL)  
Torre Wenaus (BNL)

<https://indico.bnl.gov/event/9023/>

Thank you so much for your guidance and support in the last four years



---

## EIC SOFTWARE CONSORTIUM **2016 – 2020**

The **EIC Generic Detector R&D funding** allowed us to build an active working group and foster collaborations in NP and HEP.

We will continue to work with the EIC community, now solely as EICUG Software Working Group, and focus on:

- Ensure high-quality simulations for the EIC detector development
- Common software projects that are ideally chosen by the EIC collaboration(s)