
EIC ATHENA Documentation

Release 0.00.02

Yulia Furletova, Dmitry Romanov

Jan 11, 2022

GETTING STARTED:

1	ATHENA Software Plan	1
2	Software overview	3
2.1	Full Simulation	3
2.2	Validation	3
2.3	Infrastructure	4
2.4	Fast simulation	4
2.5	Legacy full simulation	4
3	Repositories	7
3.1	GitLab	7
3.2	External repositories	8
4	Containers	9
4.1	Singularity	10
4.2	Docker containers	10
4.3	Packages & versions	10
5	CVMFS & Spack	13
6	Support	15
7	Single Particles	17
7.1	Lepton endcap (N), 130 to 177 degrees	17
7.2	Barrel region (B), 45 to 135 degrees	17
7.3	Hadron endcap (P), 3 to 50 degrees	18
8	Physics Generators	19
8.1	Deep Inelastic Scattering	19
8.2	Deeply Virtual Compton Scattering	20
8.3	Timeline Compton Scattering	20
8.4	Spectroscopy	20
9	Frequently Asked Questions	21
9.1	It seems that the <i>ReconstructedParticles</i> branches are empty, but used to be filled.	21
9.2	Should the physics working groups plan to use the calorimeter clusters?	21
9.3	Which four-vectors should I plan to use?	21
9.4	Why did the files I was using disappear? Will the files disappear?	21
9.5	How is GeneratedParticles different from mcparticles?	22
10	Run fast simulation	23

10.1	Delphes	23
10.2	EicSmear	23
11	Full simulation	25
11.1	Tutorials	25
11.2	Particle gun	25
11.3	Geometry visualization	26
11.4	Geometry conversion	26
12	Use singularity	27
12.1	Install singularity	27
12.2	Install work environment	27
12.3	Detector simulation	28
12.4	Advanced information	29
13	S3 file storage	31
13.1	XRootD access	31
13.2	Installation	32
13.3	Getting access	32
13.4	Operations	32
14	BeAGLE generator	35
14.1	CVMFS use	35
15	Geometry viewer	37
16	Local installation	39
16.1	Installation	39
16.2	References	40
17	Edit this site!	41
17.1	Running on your machine	41
17.2	Development	41

ATHENA SOFTWARE PLAN

IP6@EIC SOFTWARE PLANS

Software Core Group

- Whitney Armstrong (Argonne), Sylvester Joosten (Argonne), Dmitry Romanov (JLab)
- (tentative) EIC SWG Liaison: Andrea Bressan (INFN)

Short term strategy

Support working groups to remain productive

- Fun4All & eic_smear: Kolja Kauder
- Delphes: Miguel Arratia & Stephen Sekula
- ESCalate: Dmitry Romanov, Yulia Furletova (g4e), Nathan Brei (Tracking)
- MC Samples: Andrea Bressan

Long term strategy

- Software stack based on modern, community supported (CERN & NP) tools to avoid lock-in
- Leverage Continuous Integration workflows for detector benchmarks and optimization
- Collaborate with EIC SWG

Will need involvement from entire collaboration!

Resources

- Support large range of computing facilities (BNL/SDCC, JLab, Argonne LCRC, OSG, ...)
- Distributed data storage over all facilities: Rucio (to be rolled out ASAP), support by BNL/NPPS & (tentatively) JLab
- Development repositories: Argonne Gitlab (eicweb.phy.anl.gov) for advanced CI capabilities; can mirror to EIC SWG Github where appropriate.
- CI/Continuous Analysis/Benchmarks with eicweb
- Documentation resources main priority

Medium term/Detector proposal

Full realistic simulation/reconstruction by June

- Built on investment in software by Argonne group
- First milestones (next 2 weeks): reference detector and updated beamline for IP6 in DD4hep
- Validate reconstruction benchmarks (working reconstruction with Gaudi+ACTS, can use Jana2 if supported by JLab)
- Priority hack-a-thon to add realistic detector options identified by working groups (already started with vertex tracker), to be finished by June
- Build out library of automatized Physics benchmarks with Physics working groups (examples for DVMP already in place).

Preparing for the future at EIC

To ensure the long-term success of the ATHENA collaboration the aim is to build forward looking team of software developers. Old solutions (such as EicRoot or Fun4All) should go - computing paradigms are changing quickly! In order to attract capable new software development workforce we will focus on modern scientific computing practices. At the same time we should avoid “not-invented-here syndrome”. Leverage modern, supported by large communities (like CERN) software components where possible to build upon mature, well supported, and actively developed software stack to hit the ground running. The plan is to actively work with the EICUG SWG also.

Immediate concern: successful (best!) detector proposal

As the main goal of ATHENA collaboration is to present successful (and the best!) detector proposal, we understand that there should be no disruption of WG simulation process after the DWGs and PGWs are formed. Previously WGs already invested their time into various simulation software and advanced with the simulations. Any change to their workflows at this stage would be damaging. Premature rollout of new toolkit could take focus away from crucial technology development and down-select that needs to happen these next months.

We will deliver the new software in parallel: implement subsystems in new software toolkit to prepare for full simulation & reconstruction of a realistic candidate detector by early June 2021. DWG member can act as consultant to software group with minimal overhead

Resources and support

ATHENA SWG will provide:

- Active support for fun4all and other legacy tools throughout the coming year
- Computing resources available at large range of facilities (BNL/SDCC, JLab, Argonne LCRC, OSG, ...).
- Distributed data storage over all facilities: Rucio (to be rolled out ASAP)
- Git repositories and continuous-integration support available on the eicweb server
- Documentation of resources and tools a top priority

Day 0 software

Currently as the new software is being prepared, the suggested packages to do full and fast simulations:

- **Fast simulation:**
 - Delphes
 - Eic-Smear
- **Full simulation:**
 - ESCalate
 - Fun4All

It is explained in the details further.

SOFTWARE OVERVIEW

2.1 Full Simulation

The software is currently being developed. Here is only the software short description with links. Please read the full description next.

The main repository for ATHENA software is [here](#)

2.1.1 DD4Hep

DD4hep is a software framework for providing a complete solution for full detector description (geometry, materials, visualization, readout, alignment, calibration, etc.) for the full experiment life cycle (detector concept development, detector optimization, construction, operation). It offers a consistent description through a single source of detector information for simulation, reconstruction, analysis, etc.

- [DD4Hep main repository](#)
- [EIC detector implementations](#)

2.1.2 Juggler

Gaudi based digitization and reconstruction framework. Currently it supports ACTS based tracking, calorimetry reconstruction, some PID like RICH

- [Project Juggler](#)
- [EICD](#) - EIC event data model implemented in PODIO

2.2 Validation

On any commit to detector or reconstruction repository a bunch of validation benchmarks are being automatically run. Current benchmarks repository

- [Detector benchmarks](#) - Detector benchmarks are meant to test for regressions in individual detector subsystems without a complex reconstruction
- [Reconstruction benchmarks](#) - Reconstruction specific benchmarks (calorimetry, tracking, PID)
- [Physics benchmarks](#) - Check physical processes such as DIS, DVCS, etc.
- [Cost estimation](#) - (in development) A weighted evaluation of the detector performance versus estimated cost.

2.3 Infrastructure

- [Main repository](#)
- [Rucio](#)
- [Farms](#)

The resources page on the EICUG WG site

2.4 Fast simulation

2.4.1 Delphes

DELPHES provides the simulation of a multipurpose detector for phenomenological studies. The simulation includes a track propagation system embedded in a magnetic field, electromagnetic and hadron calorimeters, and PID. Physics objects that can be used for data analysis are then reconstructed from the simulated detector response. These include tracks and calorimeter deposits and high level objects such as isolated electrons, jets, taus, and missing energy. With features such as the particle-flow reconstruction, pile-up simulation and mitigation.

Contacts: Stephen Sekula, Miguel Arratia

Support line: [ATHENA Slack](#)

- [Delphes web site](#)
- [EIC Reference Detector Delphes implementation](#)

2.4.2 EicSmear

Eic-smear is a Monte Carlo analysis package originally developed by the BNL EIC task force. It contains classes and routines for building events in a C++ object and writing them to a ROOT file in a tree data structure, performing fast detector smearing on those Monte Carlo events. The smearing portion of the code provides a collection of classes and routines that allow simple parameterizations of detector performance, provided via a ROOT script, to be applied to any of the above Monte Carlo events. Detector acceptance can also be defined.

Contacts: Alexander Kiselev ayk@bnl.gov, Kolja Kauder kkauder@bnl.gov

- [EIC-Smear main package](#)
- [EIC detector parametrizations](#)

2.5 Legacy full simulation

2.5.1 ESCalate

Contacts: Dmitry Romanov, Yulia Furletova ([g4e](#))

- [escalate.readthedocs.io](#)
- [g4e.readthedocs.io](#)

2.5.2 Fun4All

Contacts: Kolja Kauder (for ATHENA), Chris Pinkenburg and Jin Huang (developers)

- Fun4All general Tutorials https://github.com/eic/fun4all_tutorials
- Fun4All EIC specific Tutorials https://github.com/eic/fun4all_eic_tutorials
- Fun4All Build your own GEANT4 Detector https://github.com/eic/fun4all_g4exampledetector

REPOSITORIES

3.1 GitLab

MAIN GITLAB SERVER LINK

- **Detectors - DD4Hep based detectors implementations:**
 - [ATHENA detector](#) The reference detector for ATHENA detector
 - [IP6 beamline](#)
- [EICD](#) - EIC event data model implemented in PODIO
- [Project Juggler](#) - Gaudi based digitization and reconstruction framework
- **[NPDet](#) - Nuclear physics detectors library. A set of ready to plugin detectors, examples and tools for DD4Hep for NP**
 - [documentation](#)
 - [reference API](#)
- [Datasets](#) - scripts for generating data for benchmarks
- **[Benchmarks](#) - detector, physics and other analyses that are run automatically by CI when changes to detectors or reconstruction are made**
 - [Detector benchmarks](#) - Detector benchmarks are meant to test for regressions in individual detector subsystems without a complex reconstruction
 - [Reconstruction benchmarks](#) - Reconstruction specific benchmarks (calorimetry, tracking, PID)
 - [Physics benchmarks](#) - Check physical processes such as DIS, DVCS, etc.
 - [Cost estimation](#) - (in development) A weighted evaluation of the detector performance versus estimated cost.
 - [Common benchmarks](#) - Code common to all benchmarks
- [Documentation \(dir\)](#) - this website, tutorials
- [Container](#) - source of docker images
- [MC Convert](#) - Pythia6, Beagle and others -> hepmc2/3 converter

3.2 External repositories

- DD4hep
- PODIO
- Gaudi
- Delphes EIC

CONTAINERS

The containers are released both for docker and singularity. Singularity images are automatically propagated to CVMFS:

- [Containers repository](#)
- [Docker Hub](#)
- CVMFS path:

```
singularity run /cvmfs/singularity.opensciencegrid.org/eicweb/jug_xl:3.0-  
↪stable
```

The versioning is:

- **nightly** - recreated every night, using master branches
- **testing** - recreated every night, using fixed versions
- **3.0-stable** - the latest stable version from 3.0 branch
- **unstable** (unstable-vX.x in future) - the latest build from triggered by CI from changed repo
- **3.0.0** - exact tagged version

Nightly and testing are different in that nightly uses the master branch of the software, while testing uses whatever version is given at the top of the *gitlab-ci.yml* <https://eicweb.phy.anl.gov/containers/eic_container/-/blob/master/.gitlab-ci.yml#L68>. So testing is the precursor to the next release version.

Images structure is:

- **debian_base** - is a container generic base container based on amd64/debian:testing
- **jug_dev** - have all major HENP packages such as ROOT, Geant4 and DD4HEP but without detector and reconstruction. The image is used for testing purposes and automation.
- **jug_xl** - intended to be used to run simulation and work on detectors for users. *jug_dev* + full simulation packages

4.1 Singularity

The below command creates the right working environment. It checks if there are CVMFS images available (which is true for JLab and BNL farms) and links them or automatically download images (which is a scenario for users laptops). It also creates `eic_shell` with the right environment setup, prepares the current dir to work with detector or etc.

```
curl https://eicweb.phy.anl.gov/containers/eic_container/-/raw/master/install.sh | bash
```

Please follow Use singularity how-to for full details.

4.2 Docker containers

Containers are available at eicweb namespace at [the dockerhub](#)

1. To load the container environment in your run scripts, you have to do nothing special. The environment is already setup with good defaults, so you can use all the programs in the container as usual and assume everything needed to run the included software is already setup.
2. If using this container as a basis for a new container, you can direction access the full container environment from a docker RUN shell command with no further action needed. For the most optimal experience, you can install your software to `/usr/local` to fully integrate with the existing environment. (Note that, internally, `/usr/local` is a symlink to `/opt/view`).

4.3 Packages & versions

- Included software:
 - gcc@10.2.0
 - cmake@3.18.4
 - fmt@7.1.2
 - spdlog@1.5.0
 - nlohmann-json
 - heppdt@3.04.01
 - clhep@2.4.1.3
 - eigen@3.3.8
 - python@3.7.8 with pip, numpy, pyyaml, pyafp, matplotlib, ipython, scipy
 - xrootd@4.12.3
 - root@6.22.06
 - pythia8@8303
 - hepmc3@3.2.2 +python +rootio
 - stow@2.3.1
 - podio@0.13
 - geant4@10.6.2
 - dd4hep@1.14.1

- acts@1.00.0
- gaudi@34.0
- dawn@3.91a
- dawncut@1.54a
- opencascade
- The singularity build exports the following applications:
 - eic_shell: a development shell in the image
 - container_dev: same as EIC shell
 - ipython

CVMFS & SPACK

SUPPORT

- [Mail lists](#)
- Quick help and chat is available at [our Slack channel](#)

People responsible for the software:

Core Group:

Whitney Armstrong (Argonne), Andrea Bressan (INFN), Sylvester Joosten (Argonne), Dmitry Romanov (Jefferson Lab)

Existing software support:

- **Fun4All** : Kolja Kauder
- **eic_smear** : Kolja Kauder
- **Delphes** : Miguel Arratia, Stephen Sekula
- **ESCalate** : Dmitry Romanov, Yulia Furletova (g4e), Nathan Brei (Tracking)
- **MC Samples** : Andrea Bressan

SINGLE PARTICLES

Single particle productions are split by particle type, angular regions, and thrown energy. All productions are thrown uniformly in $\cos(\theta)$ and fixed in energy.

All productions are relative to the following locations on S3 and XRootD: * <https://dtn01.sdcc.bnl.gov:9000/minio/eic/test/ATHENA/> (see `s3_file_storage.rst` for instructions) * `root://sci-xrootd.jlab.org//osgpool/eic/ATHENA/` (see `xrootd_file_storage.rst` for instructions)

Links point to the S3 location; the conversion to XRootD URIs should be trivial. Mirroring from S3 to XRootD may take up to one day.

7.1 Lepton endcap (N), 130 to 177 degrees

- **geometry master: (1M each, last update: 2021-08-14)**
 - e-: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20

7.2 Barrel region (B), 45 to 135 degrees

- **geometry master: (1M each, last update: 2021-08-14)**
 - e-: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - gamma: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - kaon0L: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - mu-: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - neutron: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - pi+: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - pi0: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20

7.3 Hadron endcap (P), 3 to 50 degrees

- **geometry master: (1M each, last update: 2021-08-14)**
 - pi+: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20
 - kaon0L: E [GeV] = 0.1, 0.2, 0.5, 1, 2, 5, 10, 20

PHYSICS GENERATORS

Physics productions are divided by physics working group or by process type.

All productions are relative to the following locations on S3 and XRootD: * <https://dtn01.sdcc.bnl.gov:9000/miniio/eic/test/ATHENA/> (see `s3_file_storage.rst` for instructions) * `root://sci-xrootd.jlab.org/osgpool/eic/ATHENA/` (see `xrootd_file_storage.rst` for instructions)

Links point to the S3 location; the conversion to XRootD URIs should be trivial. Mirroring from S3 to XRootD may take up to one day.

In case of questions or problems with this data, please let the Software Working Group know in the `#software-helpdesk` channel, or via email to `eic-ip6-software-1@lists.bnl.gov`.

8.1 Deep Inelastic Scattering

- **geometry: *master* (1M each, last update: 2021-08-18)**
 - 5x41: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 5x100: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 10x100: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 10x275: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 18x275: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
- **geometry: *acadia-v1.0-alpha* (1M each, last update: 2021-08-19)**
 - 5x41: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 5x100: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 10x100: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 10x275: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`
 - 18x275: `minQ2=1, minQ2=10, minQ2=100, minQ2=1000`

8.2 Deeply Virtual Compton Scattering

- geometry: *acadia-v1.0-alpha* (last update: in progress)
 - 18x275 (2M events)
 - 10x100 (2M events)
 - 5x41 (1M events)

8.3 Timeline Compton Scattering

- geometry: *acadia-v1.0-alpha* (last update: in progress)
 - 18x275
 - 5x100
 - 5x41

8.4 Spectroscopy

- geometry: *acadia-v1.0-alpha* (last update: in progress)
 - 10x100 psi2s, x, y

FREQUENTLY ASKED QUESTIONS

9.1 It seems that the *ReconstructedParticles* branches are empty, but used to be filled.

ReconstructedParticles used to be filled with just smeared info (so analyzers could start developing scripts even if the values in there were not from proper track reconstruction and event building). In the first *acadia* production we had an unexpected issue filling *ReconstructedParticles*, but that should be resolved soon. Plan on having access to the actual momentum from track reconstruction in *ReconstructedParticles* (with exact PID from truth, and energy from momentum and PID).

9.2 Should the physics working groups plan to use the calorimeter clusters?

We are still working on full event building that will also take into account energy from clusters, so in *acadia* the energy is based on the momentum and truth PID.

9.3 Which four-vectors should I plan to use?

As for four-vectors, this is somewhat up to you. We are not providing four-vectors in *ReconstructedParticles*. For some particles they will be four-vectors, though: particles without info from either tracking or calorimetry. Depending on the analysis you are performing, you may want to use momentum direction and energy (say, higher energy central electrons) while for others you may want to use momentum direction and momentum magnitude (say, low energy central electrons).

9.4 Why did the files I was using disappear? Will the files disappear?

- Files on S3 will not disappear, though RECO files may be replaced by updated reconstruction.
- Files on XRootD may disappear when the entire directory is deleted to stay within the required space allocations.

9.5 How is `GeneratedParticles` different from `mcparticles`?

`mcparticles` contains the full event generator record, including internal particles that are not in the final state (think: a π^0 with a strong decay into two pions). It is a record that uses `geant4` steps as the basis for its structure, and therefore includes start and end values.

`GeneratedParticles` contains only the event generator final state particles only (those with HepMC status 1). It is structured in exactly the same way as `ReconstructedParticles` and allows for comparison of truth with reconstruction.

RUN FAST SIMULATION

10.1 Delphes

<https://delphes-eic.readthedocs.io/en/latest/>

10.2 EicSmear

<https://github.com/eic/eic-smear#procedure>

FULL SIMULATION

11.1 Tutorials

Tutorials are available on a dedicated websites:

11.1.1 Detector simulation

- Full simulation tutorials
- Quick start guide
- Part I. Simple detector
- Part II. Part 2: Modifying and Adding Detectors

Participate in feedback!

11.1.2 Tracking

- (in development) ACTS tracking repo

11.2 Particle gun

There are at least 2 ways of running a particle gun out of the box:

- using npsim command line
- using geant macro file and invoke gps

Using npsim (wrapper around ddsim) command line:

```
# Assumed to run from the root of Athena detector repo
# no spread
npsim --compactFile=athena.xml --runType=run -G -N=2 --outputFile=test_gun.root --gun.
↪position "0.0 0.0 1.0*cm" --gun.direction "1.0 0.0 1.0" --gun.energy 100*GeV --part.
↪userParticleHandler=''

# uniform spread inside an angle:
npsim --compactFile=athena.xml -N=2 --random.seed 1 --enableGun --gun.energy 2*GeV --gun.
↪thetaMin 0*deg --gun.thetaMax 90*deg --gun.distribution uniform --outputFile test.root
```

11.3 Geometry visualization

There are many ways to see the geometry and tracks:

1. Through Geant4 event display
2. geoDisplay (root geoViewer)
3. ddeve (root EVE based event display...)
4. dd_web_display (using browser and jsroot library)

11.3.1 Geant4 event display

Example command:

```
# Assumed to run from the root of Athena detector repo
npsim --runType vis --compactFile athena.xml --macro macro/vis.mac --outputFile test.
↵root --enableG4GPS --enableQtUI
```

11.4 Geometry conversion

11.4.1 Convert to GDML

There is a `convert_to_gdml.py` script in the detector repository (https://eicweb.phy.anl.gov/EIC/detectors/athena/-/blob/master/scripts/convert_to_gdml.py). That can be used to export ALL of ATHENA to gdml, but not individual detector systems.

This is actually *done* on every commit and the results are saved as job artifacts.

The latest `athena.gdml` from the master branch

11.4.2 Convert to root

One can use `dd_web_display` to actually just save root geometry

```
dd_web_display --export athena.xml # will create a .root file with the geometry
```

USE SINGULARITY

12.1 Install singularity

Eic image require singularity > 3.0

The [official installation instructions](#) have many steps.

If you have ubuntu, there is a [debian repo with 3.5.2 version](#), which works pretty nicely (You will have to install dependencies, it will print them...)

```
sudo apt install containernetworking-plugins
wget http://ftp.fi.debian.org/debian/pool/main/s/singularity-container/singularity-
↪container_3.5.2+ds1-1_amd64.deb
sudo dpkg -i singularity-container_3.5.2+ds1-1_amd64.deb
```

Please don't install NeuroDebian repo from the repo as it holds v2.6 and eic image require singularity > 3.0.

12.2 Install work environment

The below command automatically creates the right working environment for detector development and running the reconstruction. It checks if there are CVMFS images available (which is true for JLab and BNL farms) and links them or downloads images (which is a scenario for users laptops). It also creates eic_shell with the right environment setup, prepares the current dir to work with detector or etc.

```
# Easy to remember link:
curl -L get.athena-eic.org | bash

# Which is an alias to:
curl https://eicweb.phy.anl.gov/containers/eic_container/-/raw/master/install.sh | bash
```

install.sh checks if it is run on BNL or JLab farms, so existing CVMFS images are used and installation is almost instant. On local systems singularity images will be downloaded.

It might be handy to copy install.sh locally and control where singularity images are being copied, disable CVMFS behaviour, and other parameters:

```
curl -L get.athena-eic.org -o install.sh
chmod +x install.sh
./install.sh --help
```

Flag	Description
-p,--prefix	Working directory to deploy the environment (D: /home/romanov/anl)
-t,--tmpdir	Change tmp directory (D: /tmp)
-n,--no-cvmfs	Disable check for local CVMFS (D: enabled)
-c,--container	Container version (D: jug_xl)
-v,--version	Version to install (D: nightly)
-h,--help	Print this message

Example of controlling the container version and image location:

```
# installs testing variant and stores image at /mnt/work/images
# (!) one has to create <prefix>/local/lib for images
mkdir -p /mnt/work/images/local/lib/
./install.sh -v testing -p /mnt/work/images
```

12.3 Detector simulation

After the installation you should have an executable script named **eic-shell** which basically just runs singularity setting the proper environment (more information about the script is below)

12.3.1 Precompiled detector

The `jug_xl` container comes with precompiled detector repository. It could be used out of the box for simulations or even changing detector parameters that doesn't require recompilation.

The precompiled detector is installed in `/opt/detector` directory. And can be used like this:

```
# Setup the proper detector environemnt
source /opt/detector/setup.sh

# Run particle gun simulation
npsim -N2 --compactFile=$DETECTOR_PATH/athena.xml --random.seed 1 --enableGun --gun.
↪energy 2*GeV --gun.thetaMin 0*deg --gun.thetaMax 90*deg --gun.distribution uniform --
↪outputFile ~/test.root
```

12.3.2 Clone detector from repository

```
git clone https://eicweb.phy.anl.gov/EIC/detectors/athena.git
git clone https://eicweb.phy.anl.gov/EIC/detectors/ip6.git
ln -s ../ip6/ip6 athena/ip6

# Build athena
mkdir athena/build && cd athena/build
cmake ../ -DCMAKE_INSTALL_PREFIX=$ATHENA_PREFIX
cmake --build ./ --target all -- -j 8
cmake --install ./
cd -
```

(continues on next page)

(continued from previous page)

```

# Build IP6
mkdir ip6/build && cd ip6/build
cmake ../ -DCMAKE_INSTALL_PREFIX=$ATHENA_PREFIX
cmake --build ./ --target all -- -j 8
cmake --install ./
cd -

# Test run geometry browser:
dd_web_display athena/athena.xml

# Test run particle gun:
npsim -N2 --compactFile=athena/athena.xml --random.seed 1 --enableGun --gun.energy 2*GeV
↪--gun.thetaMin 0*deg --gun.thetaMax 90*deg --gun.distribution uniform --outputFile ~/
↪test.root

```

12.4 Advanced information

12.4.1 CVMFS

For farms like at BNL or JLab the images are automatically replicated to CVMFS:

```

/cvmfs/singularity.opensciencegrid.org/eicweb/jug_xl*

# example to run
singularity run /cvmfs/singularity.opensciencegrid.org/eicweb/jug_xl:4.0-acadia-stable

```

12.4.2 eic-shell explained

There are actually two eic-shell scripts. One is created by the install scripts and the other lives in the container.

The one outside the container just sets ATHENA_PREFIX and runs singularity like:

```

# $PREFIX here is where you installed everything (by default where install.sh executed)
export ATHENA_PREFIX=$PREFIX/local
export SINGULARITY_BINDPATH=/mnt
singularity exec $PREFIX/local/lib/jug_xl-nightly.sif eic-shell $@

```

The **eic-shell** inside the container loads the proper environemnt and SHELL look correctly

```

## Properly setup environment
. /etc/eic-env.sh

# What eic-env does in the end is
export LD_LIBRARY_PATH=$ATHENA_PREFIX/lib:$LD_LIBRARY_PATH
export PATH=$ATHENA_PREFIX/bin:$PATH

# Run bash shell

```


S3 FILE STORAGE

To share files S3 (Simple Storage Service) is used. In this tutorial we use [MinIO client](#) to work with it.

You can get access, browse and download files using your browser, using this link:

[ATHENA S3 Space](#)

The LOGIN and PASSWORD for read access are available for all members (but can't be published at open places like this site). Please ask on [#software-helpdesk Slack channel](#) or email one of the SWG members.

One can access S3 directly using TFile like this

```
TFile* file = TFile::Open("s3https://dtn01.sdcc.bnl.gov:9000/eicctest/ATHENA/<path>.root",  
↳ "AUTH=<login>:<password>");
```

Or set environment variables `S3_ACCESS_KEY` and `S3_SECRET_KEY` instead of using AUTH and putting login and password to code;

```
export S3_ACCESS_KEY=<login>  
export S3_SECRET_KEY=<password>
```

If you encounter *Forbidden (403)* errors at BNL RCF, try adding the `NOPROXY` option:

```
TFile* file = TFile::Open("s3https://dtn01.sdcc.bnl.gov:9000/eicctest/ATHENA/<path>.root",  
↳ "NOPROXY AUTH=<login>:<password>");
```

13.1 XRootD access

One can utilize XRootD server to get access to files without credentials using:

```
root://sci-xrootd.jlab.org//osgpool/eic/ATHENA/<path to file>
```

```
std::string fileName = "RECO/JETS/crossDivNrgCrab/DIS_NC_Q2gt10_crossDivNrgCrab_25mRad_  
↳ 18x275_v1.0001.root";  
TFile* file = TFile::Open("root://sci-xrootd.jlab.org//osgpool/eic/ATHENA/" + fileName);  
// ...
```

(!) Currently (July 2021) only RECO is on xrootd (since we only have a 2.5 TB limit there)

13.2 Installation

Installing MinIO client:

```
wget https://dl.min.io/client/mc/release/linux-amd64/mc
mkdir -p $HOME/bin && mv mc $HOME/bin/mc && chmod +x $HOME/bin/mc
alias mc="$HOME/bin/mc"
mc --autocompletion
```

> You may name the command differently if mc aka midnight commander which also uses *mc* is installed

13.3 Getting access

Public read-only access

```
mc config host add S3 https://dtn01.sdcc.bnl.gov:9000 <login> <password>
```

Private write-enabled access

```
mc config host add S3rw https://dtn01.sdcc.bnl.gov:9000/ $user $key
```

Private write-enabled access at BNL

```
mc config host add S3rw http://eicoss01.sdcc.bnl.local:9000 $user $key
```

13.4 Operations

Listing contents (the eicetest is historical and may change)

```
mc tree S3/eicetest/ATHENA
mc ls S3/eicetest/ATHENA
mc find S3/eicetest/ATHENA --name '*.hepmc*'
mc du S3/eicetest/ATHENA
```

Copying contents out

```
mc cp --insecure S3/eicetest/ATHENA/EVGEN/JETS/crossDivNrgCrab/test_crossDivNrgCrab_
↪25mRad_18x275_v1.hepmc .
```

Mirroring directories

```
mc mirror --insecure S3/eicetest/ATHENA/EVGEN/JETS/crossDivNrgCrab/ crossDivNrgCrab/
```

Streaming contents out

```
mc cat --insecure S3/eicetest/ATHENA/EVGEN/JETS/crossDivNrgCrab/test_crossDivNrgCrab_
↪25mRad_18x275_v1.hepmc | head -n 20
mc cat S3/eicetest/ATHENA/EVGEN/JETS/crossDivNrgCrab/test_crossDivNrgCrab_25mRad_18x275_
↪v1.hepmc.gz | gunzip -c | head -n 20
```

Copying contents in

```
mc cp /gpfs02/eic/bpage/home/eicBeamSimu/Pythia8/headonTestJin/test_crossDivNrgCrab_*  
↪ S3rw/eictest/ATHENA/EVGEN/JETS/crossDivNrgCrab
```

Streaming contents in

```
cat /gpfs02/eic/bpage/home/eicBeamSimu/Pythia8/headonTestJin/test_crossDivNrgCrab_25mRad_  
↪ 18x275_v1.hepmc | mc pipe S3rw/eictest/ATHENA/EVGEN/JETS/crossDivNrgCrab/test_  
↪ crossDivNrgCrab_25mRad_18x275_v1.hepmc  
gzip -c /gpfs02/eic/bpage/home/eicBeamSimu/Pythia8/headonTestJin/test_crossDivNrgCrab_  
↪ 25mRad_18x275_v1.hepmc | mc pipe S3rw/eictest/ATHENA/EVGEN/JETS/crossDivNrgCrab/test_  
↪ crossDivNrgCrab_25mRad_18x275_v1.hepmc.gz
```


BEAGLE GENERATOR

BeAGLE - Benchmark eA Generator for LEptoproduction

The documentation on eic.github.io

Currently hosted at <https://gitlab.in2p3.fr/BeAGLE/BeAGLE>

Contacts:

Mark Baker mdbaker@bnl.gov Kong Tu zhoudunming@bnl.gov

14.1 CVMFS use

Here my hero lays the quest. No mere mortal is capable of building BeAGLE on his system and not everyone is able to run it on farms. But if you are still dare to try, here are the steps:

```
# Everything in CSH, you are doomed with your bash, zsh and others

setenv EICDIRECTORY /cvmfs/eic.opensciencegrid.org/x8664_sl7/MCEG/releases/env/EIC2021a/
setenv BEAGLESYS $EICDIRECTORY/PACKAGES/BeAGLE
setenv LHAPDF_DATA_PATH /cvmfs/sft.cern.ch/lcg/external/lhapdfsets/current
setenv LHAPDF_DATA_PATH ${LHAPDF_DATA_PATH}:${EICDIRECTORY}/share/LHAPDF/
setenv LHAPATH /cvmfs/sft.cern.ch/lcg/external/lhapdfsets/5.9.1/share/PDFsets
setenv LHAPDF5 /cvmfs/sft.cern.ch/lcg/external/lhapdfsets/5.9.1/share/PDFsets
setenv LD_LIBRARY_PATH $EICDIRECTORY/gcc-8.3/lib:$LD_LIBRARY_PATH
source /cvmfs/eic.opensciencegrid.org/gcc-8.3/opt/fun4all/core/gcc/8.3.0.1-0a5ad/x86_64-
↳centos7/setup.csh

# (nice idea would be to add this to a single file to source. But for hardcore knights_
↳of keyboard and console
# it is adviced not to do so, memorize the commands and type them in manually every_
↳time to feel the glory)

# Now when you are equipped, let the quest begin:
mkdir My_Beagle_Quest      # We name it awkwardly, so it is inline with everything else
cd My_Beagle_Quest
mkdir outForPythiaMode

cp $BEAGLESYS/nuclear.bin .
cp $BEAGLESYS/Examples/eD_18x135_Q2_1_10_y_0.01_0.95_test40k_Shd1_tau7_kt=ptfrag=0.32_
↳shdfac=1.32.Jpsidiffnodecay.highpf.inp example.inp
cp $BEAGLESYS/Examples/eAt1dfJn .
```

(continues on next page)

(continued from previous page)

```
$BEAGLESYS/BeAGLE < example.inp
```

```
#!) Beware, example.inp has 1mil events to generate. You probably would like to  
↳ decrease the number for the first run
```


GEOMETRY VIEWER

The latest geometry for **master** branch is available:

- [Central detector \(no obj browser\)](#)
- [Central detector \(+ obj browser\)](#)
- [Full detector with beamline \(no browser\)](#)
- [Full detector with beamline \(+ obj browser\)](#)

It might take several minutes to load due to a number of ECal fibers. We are working on fixing the performance issue.

On each commit to the **master** branch detector geometry is saved to ROOT TGeo and GDML formats. File links are:

- **detector_geo_full.root** https://eicweb.phy.anl.gov/api/v4/projects/473/jobs/artifacts/master/raw/geo/detector_geo_full.root
- **detector_geo.root** https://eicweb.phy.anl.gov/api/v4/projects/473/jobs/artifacts/master/raw/geo/detector_geo.root
- **athena.gdml** <https://eicweb.phy.anl.gov/api/v4/projects/473/jobs/artifacts/master/raw/geo/athena.gdml>

If one gets 404 or 400 error retrieving those file, that is most probably means, that a master branch code and its artifacts (resulting files) are being built right now and not yet ready. You can check the status at [repository pipelines](#)

LOCAL INSTALLATION

The supported local installation is done with Spack

Spack is a package management tool designed to support multiple versions and configurations of software on a wide variety of platforms and environments. It was designed for large supercomputing centers but can also be used by a regular users.

Spack installation allows to install ATHENA stack natively on your system (as much as possible). It is easier to work with in terms of IDEs, OpenGL (Geant4 event display), debugging, etc. The downside of this is that it requires of compilation of lots of packages, which takes time (hours) and is prone to errors depending on your system. If setup fails, it require some spack-kung-foo to debug and recover the installation. Thus:

THIS TYPE OF INSTALLATION IS CONSIDERED FOR EXPERTS. DO IT ON YOUR OWN RISK.

16.1 Installation

To install Spack and EIC-Spack repository it is recommended to use master branch of spack as several of issues we submitted are not yet on any tag.

```
git clone https://github.com/spack/spack.git
. spack/share/spack/setup-env.sh           # Add this to your e.g. ~/.bashrc
git clone https://github.com/eic/eic-spack.git
spack repo add eic-spack
```

Create environment file **athena.yaml** with the next content:

```
spack:
  specs:
    - acts +dd4hep +digitization +examples +fatras +geant4 +identification +json +tgeo
↵+ipo
    - boost@1.76.00
    - cmake
    - clhep cxxstd=17
    - dd4hep +geant4 +assimp +hepmc3 +lcio +ipo
    - eigen
    - gaudi +ipo
    - geant4 +ipo +qt +opengl -python +threads +vecgeom cxxstd=17
    - genfit +ipo
    - hepmc3 +interfaces +python +rootio
    - lcio +ipo
    - mesa -llvm swr=none
```

(continues on next page)

(continued from previous page)

```
- nano
- podio +ipo
- pythia8 +fastjet
- root cxxstd=17 +fftw +fortran +gdml +mlp +pythia8 +root7 +tmva +vc +xrootd +ssl
- xrootd cxxstd=17 +python
- igprof
- npdet +geocad
- eicd
- log4cxx
- afterburner +zlib +root
- juggler
- athena-eic +reconstruction
- py-pandas
- py-matplotlib
concretization: together
config:
  install_missing_compilers: true
  # Optional: install path and view path
  # install_tree:
  # root: /opt/software
  #view: /opt/local
```

Create the environment from the file

```
spack env create athena athena.yaml
spack env activate athena
spack concretize
spack install --fail-fast -v -j8
```

16.2 References

- [Spack main documentation](#)
- [Spack environments tutorial](#)
- [EIC-spack repository](#)

EDIT THIS SITE!

eic.phy.anl.gov/ip6/

This site is open to everyone. Users are encouraged to contribute to this documentation! If you would like to add/edit the information on this site - you are welcome to do this!

The pages are automatically updated, when changes are committed to the master branch of

[ATHENA SWG documentation GitLab repo](#)

P.S. The suggested update workflow is by [creating merge requests](#)

17.1 Running on your machine

For more extensive editing it is convenient to run the site on your local machine to instantly see the changes.

```
# get the website
git pull https://eicweb.phy.anl.gov/EIC/documentation/athena athena_doc

# install sphinx python libraries
python3 -m pip install sphinx sphinx_rtd_theme recommonmark sphinx-autobuild

# start the site
cd athena_doc
sphinx-autobuild docs docs/_build/html
```

By default the site will be available at <http://127.0.0.1:8000/> (you can look at sphinx-autobuild flags to change that)

17.2 Development

More formal explanation:

17.2.1 Requirments:

- [Sphinx](#) - Python documentation generator
- [Read the Docs Sphinx Theme](#) - Theme for final output
- [recommonmark](#) - Markdown pages support Sphinx

17.2.2 Building

```
# from project root
sphinx-autobuild docs docs/_build/html

# from docs root
sphinx-autobuild . _build/html

# one time build without watching for changes
sphinx-build docs docs/_build/html
```