

Introduction to Athena and MC Generation

W. Ehrenfeld

Universität Hamburg/DESY

23rd August 2007

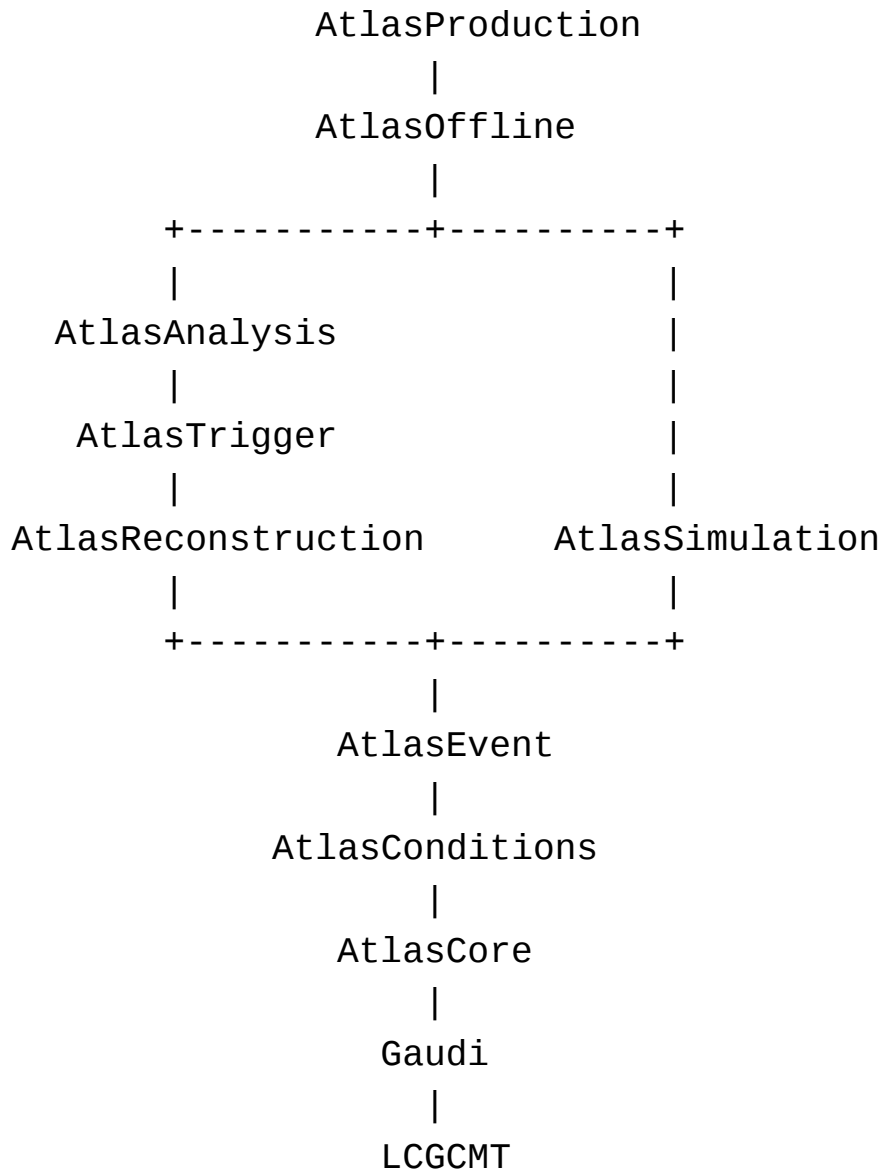
- **Athena**
- **Accounts + Setups**
- **Generators**
- **Run generators within athena**
- **Read generator output within athena**

Some useful links:

- Atlas home page <http://atlas.web.cern.ch/Atlas/index.html>
- Beginners/Computing Workbook
<https://twiki.cern.ch/twiki/bin/view/Atlas/WorkBook>
- Atlas DESY home page <http://www-atlas.desy.de/>
- Atlas DESY starter page
<http://www-atlas.desy.de/internal/newComerInfo.html>

Accounts:

- <http://www-atlas.desy.de/internal/accounts.html>
- CERN: Atlas computing account
See Sabine Kohn
Working at CERN (Ixplus), CVS access, Indico System
- DESY: atlas group and resource
access to software (std kits, DDM tools) and resources (work group servers, file server, Tier2 SE, batch system)
- GRID: grid certificate and Atlas VO
access to official and private produced files



Projects:

- default project is AtlasOffline
- Generators are in AtlasSimulation
- Job options are in AtlasProduction (AtlasOffline for r13.0.30/13.1.0)
- Production scripts are in AtlasProduction

Release numbering:

- Stable production release: 12.0.3, 13.0.20
- Nightly production release: 13.0.X
- Stable development release: 12.3.0, 13.1.0
- Nightly development release: 13.X.0
- Validation release
- Migration release

- See Computing Workbook
- Algorithms
- Services
- (Alg)Tools
- others

- **CERN:** see work book
- **DESY:** use local kits (here 12.0.6)
 - see <http://www-atlas.desy.de/internal/sw-HH.html>
 - source `/afs/desy.de/group/atlas/etc/login.sh` (always)
 - take requirement file from <http://www-atlas.desy.de/internal/sw-HH.html>
 - source `/afs/desy.de/group/atlas/software/12.0.6/CMT*/mgr/setup.sh`
 - `cmt config`
- **DESY:** use CERN software
 - get AFS token for CERN: `klog -cell cern.ch -principal username -tmp`
 - source `/afs/cern.ch/user/u/username/cmthome/setip.sh -tag=...`
- **DESY:** use CERN software
 - get AFS token for CERN: `klog -cell cern.ch -principal username -tmp`
 - get CERN requirements file
 - source CERN cmt version
 - `cmt config`

See Computing Workbook for a simple example.

Needed or available components:

- Generator
- Postgenerators (Tauola, Photos)
- Event filter
- Dumper
- Consistency tool
- Pool tools

Production is done in a different way. Uses JobTransforms from AtlasProduction project:
`csc_evgen_trf.py`

<https://twiki.cern.ch/twiki/bin/view/Atlas/AtlasPythonTrf>

Official JobOptions:

- r12: Generators/DC3_joboptions package
- r13: Generators/EvgenJobOptions package
- use `get_files` for copying official JO from release

Documentation:

- MC working group:
<https://twiki.cern.ch/twiki/bin/view/Atlas/MonteCarloWorkingGroup>
- code: Container package is Generators (CVS or LXR)
- TWiki: <https://twiki.cern.ch/twiki/bin/view/Atlas/McGeneratorsForAtlas>
- TWiki: *Generatorname*ForAtlas
- generator manuals

Select only a subset of generated events fulfilling certain cuts.

package: Generators/GeneratorFilters

Examples:

- LeptonFilter: η and p_t cut
- MultiLeptonFilter: η , p_t and N cut
- WMultiLeptonFilter: specify the grandmother of a electron or muon, either it comes from Top- $\rightarrow$ W- $\rightarrow$ e/mu or H- $\rightarrow$ W- $\rightarrow$ e/mu
- JetFilter

Others: MuonFilter, MultiMuonFilter, PhotonFilter, TauFilter, HiggsMultiLeptonFilter, TTbarPlusJetsFilter, ...

<http://alxr.usatlas.bnl.gov/lxr/source/atlas/Generators/GeneratorFilters/>

The generator output is stored as McEventCollection (package Generators/GeneratorObjects) using the HepMC package.

<http://atlas-computing.web.cern.ch/atlas-computing/links/nightlyDocDirectory/GeneratorObjects/html>

<https://twiki.cern.ch/twiki/bin/view/Atlas/McEventCollection>

<https://twiki.cern.ch/twiki/bin/view/Atlas/HepMC>

McEventCollection is a collection (vector) of GenEvent. GenEvent has the particle and vertex information. Pile up and cavern background can result in more than one GenEvent per event.

Generator output is written to a pool file. Standard extension is evgen.EVT.pool.root.

There are three different objects/StoreGate keys available, but only one key is available at one processing step/output file:

- generation: GEN_EVENT (unmodified output from generator)
- simulation: TruthEvent (merged from GEN_EVENT and Geant4 list)
- digitization: TruthEvent
- ESD: TruthEvent
- AOD: GEN_AOD (slimmed)
- (AOD: SplcMC)

checkFile.py shows information about pool file contents (size, class name, storegate key)

To reduce the size of the MC container in the AOD, a slimming between ESD and AOD creation is done using McParticleTools/EtaPtFilterTool.

The following particles are kept:

- all particles at the hard scattering vertex
- all particles which fall into the following η/p_T slices (configurable through jobOption):
 - $p_T > 0.3 \text{ GeV}$ and $|\eta| < 3.5$
 - $p_T > 1.0 \text{ GeV}$ and $|\eta| < 5.5$

Examples:

- Workbook
<https://twiki.cern.ch/twiki/bin/view/Atlas/WorkBookGeneration>
- transform job options r12
 - single_part: DC3_joboptions/DC3.007000.singlepart_e_Et10.py
 - pythia: DC3_joboptions/DC3.005010.J1_pythia_jetjet.py
 - filter: DC3_joboptions/DC3.008130.AlpgenJimmyZeeNp0LooseCut.py
 - susy: DC3_joboptions/DC3.005403.SU3_jimmy_susy.py
- transform job options r13
 - single_part: EvgenJobOptions/CSC.007000.singlepar_e_Et10.py
 - pythia: EvgenJobOptions/CSC.005010.J1_pythia_jetjet.py
 - susy: EvgenJobOptions/CSC.005403.SU3_jimmy_susy.py

```
[atlas-wgs01] ~/atlas/testarea/12.0.6/run.tutorial $ csc_evgen_trf.py --help
```

```
JobTransform csc_evgen version EvgenJobTransforms-00-04-17
```

Generate events for physics process defined in jobOptions file given in jobConfig argument

```
usage: csc_evgen_trf.py [options] <runnumber> <firstevent> <maxevents> <randomseed> <jobconfig> <outputfile>
```

Arguments can be passed in order (positional) or as name=value (named). Arguments in [] are optional

Option -h,--help: get detailed help

Options:

-h,--help	Print detailed help
-l <value>,--loglevel=<value>	Output message level. Possible values: ['INFO', 'ALL', 'VERBOSE']
-t,--test	Run in test mode: skip some checks
-a <value>,--athenaopts=<value>	Options to be passed on to athena

Arguments:

- 1 runNumber (int) # each run number corresponds to one physics process
- 2 firstEvent (int) # number of the first event in the output data file
- 3 maxEvents (int) # Maximum number of events to process
- 4 randomSeed (int) # random seed for physics generators
- 5 jobConfig (list) # jobOptions fragment containing the physics and the configuration settings
- 6 outputEvgenFile (str) # Output file that contains generated events
- [7 histogramFile] (str) default='NONE' # Output file that contains histograms.
- [8 ntupleFile] (str) default='NONE' # Output file that contains ntuples.
- [9 inputGeneratorFile] (str) default='NONE' # Input file used by the particle generator to generate events

```
INFO: read 112 error categories from file /afs/desy.de/group/atlas/software/12.0.6/AtlasProductionTools/
```

There are many ways to look at generator output:

- PrintMC/DumpMC
- AthenaROOTAccess (for r13) [TWiki:AthenaROOTAccess](#)
- Custom algorithm (see Computing or Physics Analysis workbook)
[TWiki:PhysicsAnalysisWorkBookExMC](#)
- generate CBNT ← example
- AtlFast → CBNT
- AtlFast → HighPtView
- full simulation → CBNT
- full simulation → HighPtView

JO for PrintMC:

```
#-- PrintMC
theApp.Dlls += ["TruthExamples" ]
theApp.TopAlg += ["PrintMC"]
PrintMC = Algorithm( "PrintMC" )
PrintMC.McEventKey = "GEN_EVENT" // the storegate key for reading
PrintMC.VerboseOutput = TRUE // dumps the event to stout (TRUE/FALSE)
PrintMC.PrintStyle = "Barcode" // Event print style (Vertex/Barcode)
PrintMC.VertexInfo = FALSE // print the production vertex position as well
PrintMC.FirstEvent = 1 // First event to print
PrintMC.LastEvent = 10 // Last event to print or all after FirstEvent are printed
```

JO for DumpMC:

```
#--- DumpMC
theApp.Dlls += ["TruthExamples" ]
theApp.TopAlg = ["DumpMC"]
DumpMC=Algorithm("DumpMC")
DumpMC.McEventKey='GEN_EVENT'
DumpMC.McEventOutputKey='GEN_EVENT'
```



Example - Generate in 12.0.6

JO and log files for this example are in `~efeld/public/Atlas/MCTutorial`

```
source ~/cmthome/setup.sh -tag=12.0.6,AtlasProduction
athena.py myGenOptions-1.py| tee output.1.log
athena.py myGenOptions-2.py| tee output.2.log
csc_evgen_trf.py 5144 1 5000 1 DC3.005144.PythiaZee.py mc12.005144.PythiaZee.e
```

The combined ntuple (CBNT or AACBNT or AANT) is a standard method to write a object to a ntuple file. Standard support in reco jobs.

TWiki:ComBinedNTupleAthena
TWiki:CBNTVariableList

JO and log files for this example are in `~efeld/public/Atlas/MCTutorial`

```
source ~/cmthome/setup.sh -tag=12.0.6,AtlasProduction  
athena.py myCBNT.py | tee cbnt.log
```

Write a TTree (name: CollectionTree) into root file cbnt.root.

Happy generating!