

# The ATrig Environment

*Saul Gonzalez (University of Wisconsin - Madison)*

---

*December 8, 1998*

*ATLAS Software Week*

# Reminder: ATrig

- ATrig: tool for evaluating trigger performance

ATrig “flow and control”:

- INPUT:
  - DICE output
- CONTROL:
  - data cards
  - constants database files (title files)
- OUTPUT:
  - ASCII files (for stand-alone trigger studies)
  - Hbook files (for trigger performance studies)
  - (ZEBRA banks)

# ATrig in CVS/SRT

- ATrig is now fully maintained, developed, and used in the CVS/SRT development environment
- Program control parameters (“title files”) are included in the ATrig repository
  - easier to track new parameters
  - self-contained releases
- Also included in the repository:
  - scripts for running ATrig
  - scripts for testing ATrig
  - documentation

# Running ATrig

- Running ATrig can be confusing since:
  - multiple parameter files (data cards, title files, ATLAS geometry files)
  - many running conditions (low-lumi, hi-lumi, noise, no-noise etc.)
  - different data sources: tapes, disk files
  - strange things can happen: “*where is the hbook file?*”, “*where is my log-file?*”
- So, created a “working environment” that:
  - (for the beginner) takes care of these annoying problems,
  - (for the expert) allows full customization,
- How: setup the ATrig environment via repository-resident *perl* scripts

# Scripts for an ATrig Environment

The ATrig environment is defined with one script: **SetAtrigEnv**

- **SetAtrigEnv** collects, for a given ATrig release, all necessary information for running an ATrig job.
- **SetAtrigEnv** script stored in repository; back-compatible to previous releases

ATrig jobs are actually run with **RunAtrig**, which is set up for the user by **SetAtrigEnv**

- **RunAtrig** script stored in repository

# Script for ATrig: SetAtrigEnv

- Run once for each release or private development version. Creates a sub-directory tree under the requested version and the appropriate job submission script (**RunAtrig**).
- Example for release **0.0.13**:

0.0.13/ ← *Release version*

/job

*Links to appropriate binary file, parameter files, and job running script in repository*

/log

*Job log files*

/dmp

*Links to output ATrig ASCII files*

/his

*Links to output ATrig Hbook files*

# Example: SetAtrigEnv

```
[atlas07] ~/public > work/installed/share/job/SetAtrigEnv
/afs/cern.ch/atlas/software/dist
SetAtrigEnv:
  Will attempt to create an Atrig environment for version "0.0.14".
  This includes the creation of one directory tree under "/afs/cern.ch/user/s/saul/public".
  Do you wish to proceed? (y/n) y
```

```
Creating an ATRIG environment for release 0.0.14 (default)
Creating directory 0.0.14
Creating directory 0.0.14/log (ATRIG log files are kept here)
Creating directory 0.0.14/his (ATRIG hbook files are kept here)
Creating directory 0.0.14/dmp (ATRIG ASCII files are kept here)
Creating directory 0.0.14/job (ATRIG binaries, scripts, and parameters)
```

```
[atlas07] ~/public >
[atlas07] ~/public > ls 0.0.14/*
```

```
0.0.14/dmp:
```

```
0.0.14/his:
```

```
0.0.14/job:
```

```
AtrigBin  AtrigPar  RunAtrig
```

```
0.0.14/log:
```

# Running ATrig: RunAtrig

- Actual script to run ATrig: batch (LSF) or interactive
- Best used under **SetAtrigEnv** but not necessary
- Executable and parameter files are taken from the “official release” area (e.g., /afs/cern.ch/atlas/software/dist/0.0.14/atrig/DB/prod/hilum) unless the user requests otherwise by placing a private file in the /job directory
- Any pre-defined run conditions can then be specified at run time:

```
RunAtrig -p [hilum | lolum | nonoise | etc.]
```

# Example: RunAtrig

```
[atlas07] ~/public > 0.0.14/job/RunAtrig
```

```
RunAtrig:
```

```
-D tapeID:      Data tape identifier
-f first_file:  First (or only) file from Data tape to process
-l last_file:   Last file from Data tape to process (optional)
-n nEvent:     Number of events [99999]
-p RunCond:    Key word for run condition [lolum]
               Available key words: lolum, hilum, user defined
-s stage_size: Stage area size for Data file in Mb.
-F file_name:  Name of disk file to process (incompatible with
               above options)
-q LSFQUE:     Name of LSF queue [1nh]
-a start_time: day:hour:min to start job (optional)
```

Examples:

```
RunAtrig -D lt0023 -f 1 -l 5 -s 50 -q medium
```

will run ATRIG on files 1 to 5 of tape lt0023.

If you just want to run on 1 file 'n' then specify '-f n -l n'.

You should know how big the files are before staging them.

```
RunAtrig -F /usr/data/myfile.RZ -n 100
```

will run ATRIG on 100 events of disk file myfile.RZ

```
[atlas07] ~/public > 0.0.14/job/RunAtrig -Dy00039 -f1 -l4 -philum -q1nh -s 900
```

```
RunAtrig: Default temporary area being set to /scratch/zp/saul
```

```
Input files: y00039.1 to y00039.4
```

```
Log file: /afs/cern.ch/user/s/saul/public/0.0.14/job/./log/y00039.1.4.log
```

```
LSF queue: 1nh
```

```
Parameter files:
```

```
/afs/cern.ch/user/s/saul/public/0.0.14/job/./job/AtrigPar/atlas_progflow.tit
```

```
/afs/cern.ch/user/s/saul/public/0.0.14/job/./job/AtrigPar/hilumi_param.tit
```

```
/afs/cern.ch/user/s/saul/public/0.0.14/job/./job/AtrigPar/atrig_prod_hilum.dat
```

```
Job <104975> is submitted to queue <1nh>.
```

- Easily customized

# Conclusion

- We have a script-driven ATrig environment that can manage different running conditions
- It is suitable for both beginners and developers
- It also makes ATrig productions easier and more consistent
- As ATrig develops, will try to keep a similar environment to “ease” transition
- Documented in:  
*<http://www-wisconsin.cern.ch/~atsaul/Docs/RunningAtrig.pdf>*