CAF Brainstorming – Trigger

30 January 2009
Discussion on CAF activities for online/offline activities

Ricardo Gonçalo, Szymon Gadomski on behalf of the Trigger

Trigger Tasks in CAF

CAF was used in 2008 run for 3 main purposes:

1. Run High Level Trigger on Level 1-selected bytestream data
   - Test new Super Master Keys before online deployment
   - Classify High Level Trigger errors, crashes, etc
2. Run trigger offline monitoring on bytestream data from step 1
3. Produce ESDs with trigger information from step 1 bytestream data for analysis
4. Estimate trigger rates for new menus (occasional and lower priority)

Plans for the CAF in 2009:

- Initial running will be pretty much the same as 2008 (running HLT on L1-only data, etc)
- Plans for steady-state data taking :
  1. Run error analysis/classification/recovery on all debug stream events
  2. Run Data Quality monitoring jobs on some/all express stream data
  3. Run online/offline trigger result comparison on some/all express stream data
  4. Continue to test new menus and code offline in the CAF before deploying them

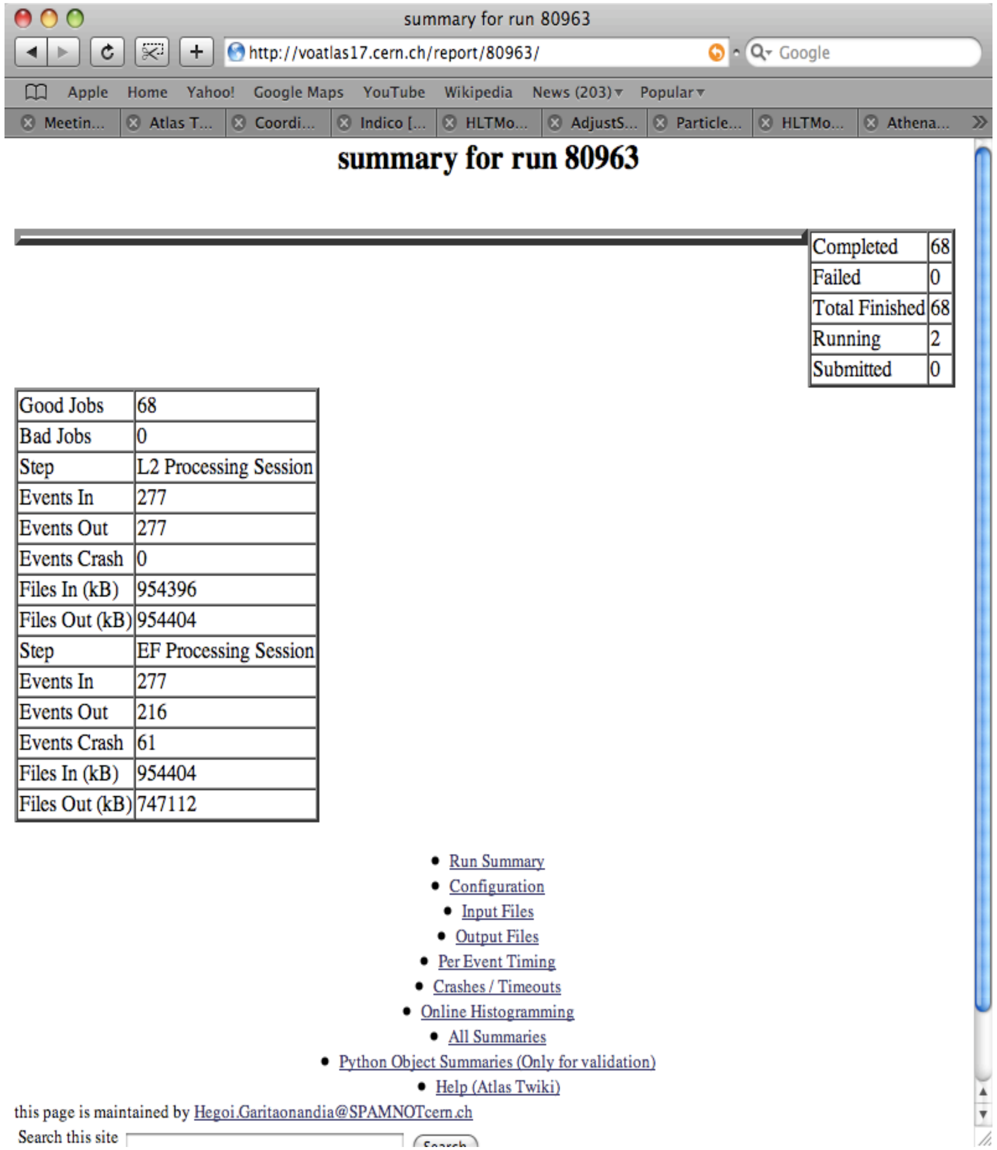
Task Management

Initial system written and developed for 2008 run:
HDEBUG framework (Hegoi Garitaonandia)

- Job submission for step 1 used HDEBUG, based on GANGA, and publishes results to web server
- Monitoring jobs run trigger monitoring tools in TrigHLTMonitoring (Tier0) and TrigHLTOfflineMon (CAF) under AthenaMon
- Monitoring and ESD (steps 2. and 3.) used simple queue submission scripts (bsub)
- Small library of useful scripts for error classification, etc

Plans for 2009:

- Automate job submission in HDEBUG framework – eliminate manual submission of jobs on debug and express stream
- Complete merger of error classification scripts into HDEBUG
- Ongoing development of analysis algorithms for online/offline comparison – to be managed by HDEBUG
- Continue to use CAF for testing new SMKs before online deployment of menu
  - Simplify submission of test jobs and make it more robust



summary for run 80963

Completed	68
Failed	0
Total Finished	68
Running	2
Submitted	0

Good Jobs	68
Bad Jobs	0
Step	L2 Processing Session
Events In	277
Events Out	277
Events Crash	0
Files In (kB)	954396
Files Out (kB)	954404
Step	EF Processing Session
Events In	277
Events Out	216
Events Crash	61
Files In (kB)	954404
Files Out (kB)	747112

- Run Summary
- Configuration
- Input Files
- Output Files
- Per Event Timing
- Crashes / Timeouts
- Online Histogramming
- All Summaries
- Python Object Summaries (Only for validation)
- Help (Atlas Twiki)

this page is maintained by Hegoi.Garitaonandia@SPAMNOTcern.ch

Search this site

Centralizing task management?

Which of these tasks would benefit from a central system - and which would not?

- This arrives a bit late… but it would be useful if it becomes the standard

- Submission of jobs on DEBUG and EXPRESS streams could benefit from central facility – being done in HDEBUG
  - Initial data: run different menu and (possibly) release than was used online
  - Steady state: run same release and menu as used online
  - Would be useful to have system to automatically send jobs to all new data from these streams – under development in trigger, but perhaps useful elsewhere

- Testing new menus: need to specify data set, menu, release (possibly nightly)
  - Need tool to un-stream data before running – avoid mixing streams after new HLT version runs on data

- Other constraints:
  - DQ, debug stream and test jobs need to publish results in web-accessible way for remote DQ
  - Need to run this asynchronously from (before) offline reconstruction
    - For the DEBUG stream this means quasi-real time
  - Farm/queue load varies mostly depending on demand for testing new menus (time critical)