My Contribution to TREND

Data Acquisition and Storage

Fabio Hernandez

fabio@in2p3.fr

Beijing, September 7th 2012

Outlook

- Controller for the data acquisition process
- Data transport and storage
- Future: distributed data processing

DAQ Controller

fabio hernandez
fabio@in2p3.fr
CAS/IHEP Computing Centre
CNRS/IN2P3 Computing Centre

DAQ launcher overview

- Thin layer of software developed for making easier the life of the data acquisition operator

- What it does?

launches several executables needed to perform the acquisition, i.e. the ones that:

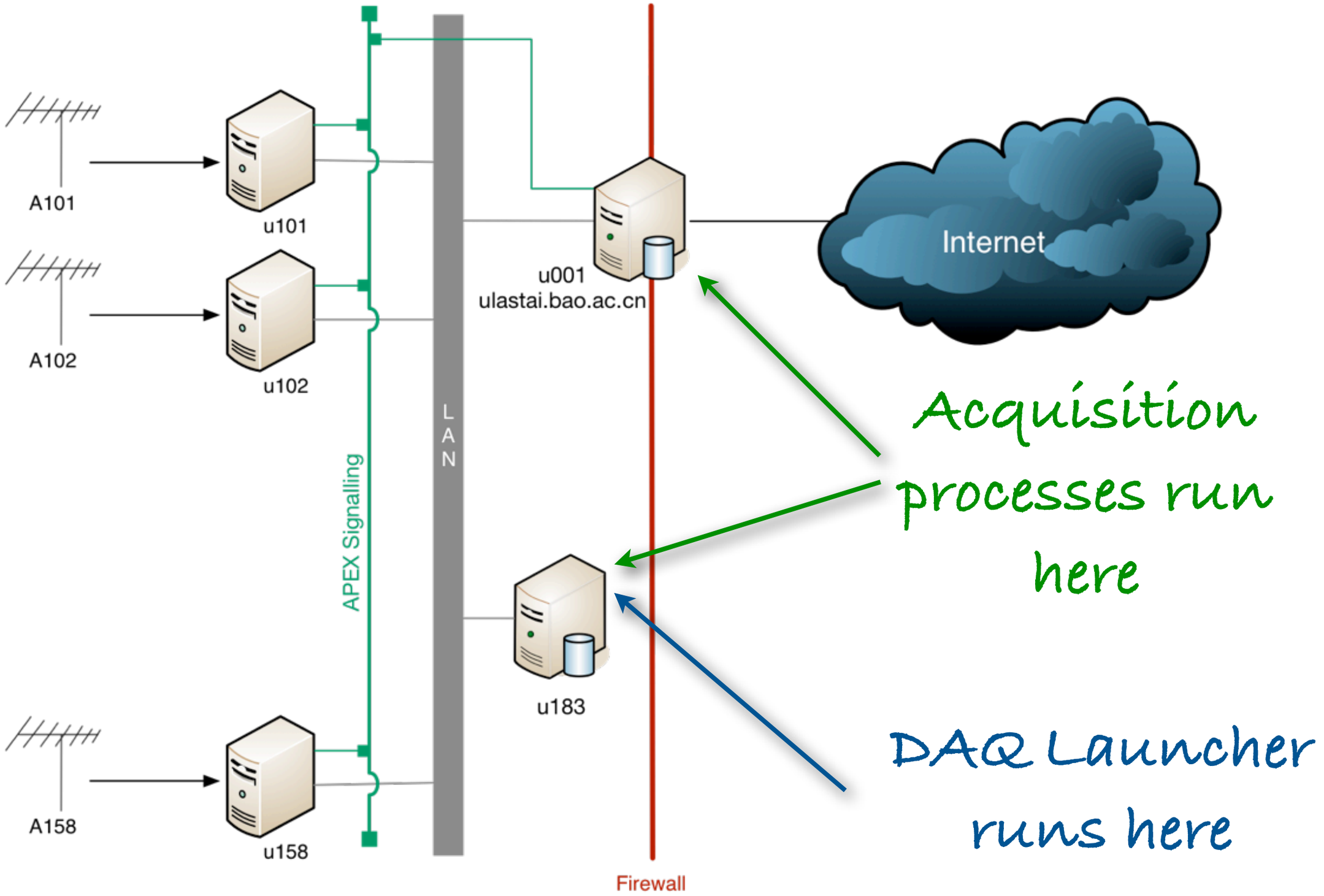
1) get all the computers ready to start acquisition

2) perform the level 0 and level 1 trigger

3) perform the acquisition of the background data

4) perform the acquisition of the PSD data

DAQ launcher overview (cont.)



DAQ launcher overview (cont.)

- How does it work?
- Single input file containing a text description of the acquisition environment, i.e.

  identifiers of the acquisition devices (antennas & scintillators), their acquisition status (on or off) and their logical grouping (east or cross)

  the executables and their options for each acquisition grouping: antennas, scintillators, background, PSD

  name of the file containing the run number

- File format: JSON

  text, human and machine readable

DAQ launcher overview (cont.)

```
"acquisitionDevices": {
        "antennas": {
            "101": {
                "antennaId": "A101",
                "hostName": "u101",
                "isActive": true,
                "memberOfGroup": "east"
            },
            …
 }
"acquisitionProcesses": {
    "background": {
        "daqmode": "Slave",
        "dataloc": "/data/dev",
        "executable": "/home/pastsoft/trend/daq/background",
        "length": 1024,
        "mpirun": "dev_mpirun.py",
        "period": 1200,
        "simopts": "",
        "verbosity": "INFO"
    },
    "cosmic": {
        "groups": {
            "antennaGroups": {
                "cross": {
                    "daqmode": "Master",
                    "dataloc": "/data/dev",
                    "executable": "/home/pastsoft/trend/daq/online-trigger",
                    "mpirun": "dev_mpirun.py",
                    "multiplicity": 4,
                    "simopts": "",
                    "threshold": 8,
                    "verbosity": "DEBUG"
                },
                …
            "scintillatorGroups": {
                "scintillators": {
                    "daqmode": "Master",
                    "dataloc": "/data/dev",
                    "executable": "/home/pastsoft/trend/daq/online-trigger",
                    "mpirun": "dev_mpirun.py",
                    "multiplicity": 2,
                    "simopts": "",
                    "threshold": 80,
                    "verbosity": "DEBUG"
                }
```

DAQ launcher overview (cont.)

- Language: Python 2.7
- Location:

  under `u183:/home/pastsoft/trend`

- Usage:

```
./daqLauncher.py --help
usage: daqLauncher.py [-h] [--dev] [--go] [--dbfile DBFILE] [--version]
```

DAQ launcher overview (cont.)

- Improvements under development

  to stop the run based on the number of recorded events

  to redirect all the traces generated by the launched executables into a single file (this file could also be stored with the run's data)

  to improve the messages shown when there are problems synchronizing the acquisition machines at the start of each run

  to detect the abnormal end of one of the executables and stop the run

  to implement a mechanism for notifying people of acquisition events (run start & stop, problems with the acquisition, etc.)

Data transport and storage

fabio hernandez
fabio@in2p3.fr
CAS/IHEP Computing Centre
CNRS/IN2P3 Computing Centre

Data transport and storage

- TREND data is transported from Beijing to Ulastai by removable disk
- At IHEP, the disk contents is copied to an networked file system, accessible from any machine in the cluster
- Transport preparation phase

  the files composing each run are renamed for ensuring conformance to the agreed naming conventions

  index of the contents of each run is generated (JSON format) and stored with the run

Data transport and storage (cont.)

- Data transfer from IHEP to iRods @ CC-IN2P3

  batch transfer of files of each run and injection into iRods

  iRods client commands don't exploit the available network bandwidth when transfers involve long distance travel

  I developed a machinery for transferring several files simultaneously, without flooding the receiving server

  example of a transfer campaign:1- 2TB of data in 20.000+ files

Data transport and storage (cont.)

- Improvements to explore

  possibility of using the network for transferring the data from Ulastai to Beijing (IHEP)

  preliminary data compression tests show ratios 2:1

  we need to validate that the network link from Ulastai can be used for this

  if so, data could be transferred to CC-IN2P3 with a much shorter delay

Distributed data processing

fabio hernandez
fabio@in2p3.fr
CAS/IHEP Computing Centre
CNRS/IN2P3 Computing Centre

Distributed data processing

- Currently, offline data processing for TREND is exclusively performed at CC-IN2P3 (Lyon, France)

  TREND software has dependencies on the particular computing environment of CC-IN2P3 (file systems, batch system, software stack, etc.)

- In the future, it would be wiser to exploit the available computing capacity of several sites

  France-Asia virtual organization has been set up for providing a easy way to get started using distributed resources for scientific experiments of modest size

  this very lightweight VO could fit the needs of TREND

Distributed data processing (cont.)

- Participating computing centers

  IHEP (China), CC-IN2P3 (France), KEK (Japan), Kisti (Korea)

- CC-IN2P3 operates an instance of DIRAC that can be used by the members of the France-Asia virtual organization

DIRAC

- DIRAC

  open-source middleware initially developed for the needs of LHCb experiment, but now used by several other experiments

  http://diracgrid.org

- DIRAC integrates heterogenous computing resources of the computing centers and makes grid convenient for end-users

- Users interact with the system though command-line interfaces or from Python scripts

  easy to install on your personal computer

  easy to integrate with your jobs, in particular if you use Python: this means, DIRAC-managed jobs can interact directly with DIRAC services by using the Python API (for instance, for uploading or downloading files, or querying the system, etc.)

DIRAC (cont.)

- Workload management

  Central task queue

  DIRAC submits pilot jobs to the execution sites

  Pilot jobs fetch the end-user job and his proxy (credentials)

  The end-user job is executed using his credentials, so it can interact with other services that require those credentials

- Data management

  a DIRAC job can interact with grid storage elements

  DIRAC provides also a file catalog for storing user metadata

Job plots as user@BES_Production

besdirac02.ihep.ac.cn/DIRAC/BES_Production/user/systems/accountingPlots/job#ds9:_plotNames18:Job%20execution%20rates9:_groupings4:Sites13:_timeSelectors6:604800s9:_typeNames3:Jobe

System Jobs Views Tools — Selected setup: BES_Pro

Jobs history plot generation

Plot to generate:
Job execution rate

Group by:
Site

Time span
- Last Day
- Last Week
- Last Month
- Manual Select

Initial time: YYYY-mm-dd

End time: YYYY-mm-dd

- By Quarter

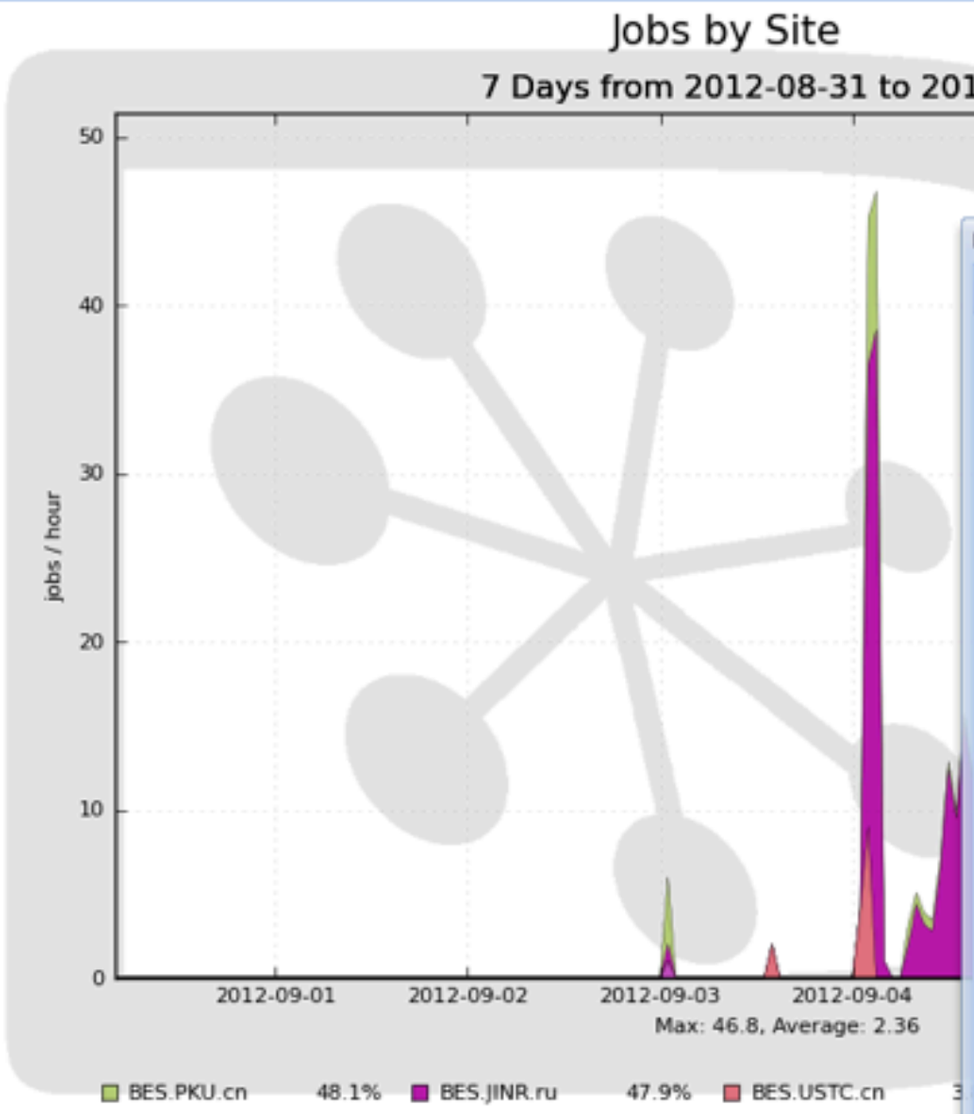
Selection conditions

Advanced options

Plot | Plot in new tab | Reset

Job execution rate by Site for last week

Refresh CSV data — Auto refresh :



Launchpad

Proxy Status: Valid — Add Parameters — Clear Sandbox

JDL

JobName:	DIRAC_fabio_163381
Executable:	/bin/ls
Arguments:	-ltrA
OutputSandbox:	std.out, std.err

Input Sandbox — Browse...

Submit | Reset | Close

System > Accounting > Job plots

fabio@ user (/O=GRID-FR/C=FR/O=CNRS/OU=CC-IN2P3/CN=Fabio

Questions & Comments

fabio hernandez
fabio@in2p3.fr
CAS/IHEP Computing Centre
CNRS/IN2P3 Computing Centre