Functional and Large-Scale Testing of the ATLAS Distributed Analysis Facilities with Ganga

D.C. van der Ster (CERN), J. Elmsheuser (LMU),
M. Slater (Birmingham), C. Serfon (LMU),
M. Biglietti (INFN-Napoli), F. Galeazzi (INFN-Roma III)

Overview of the talk

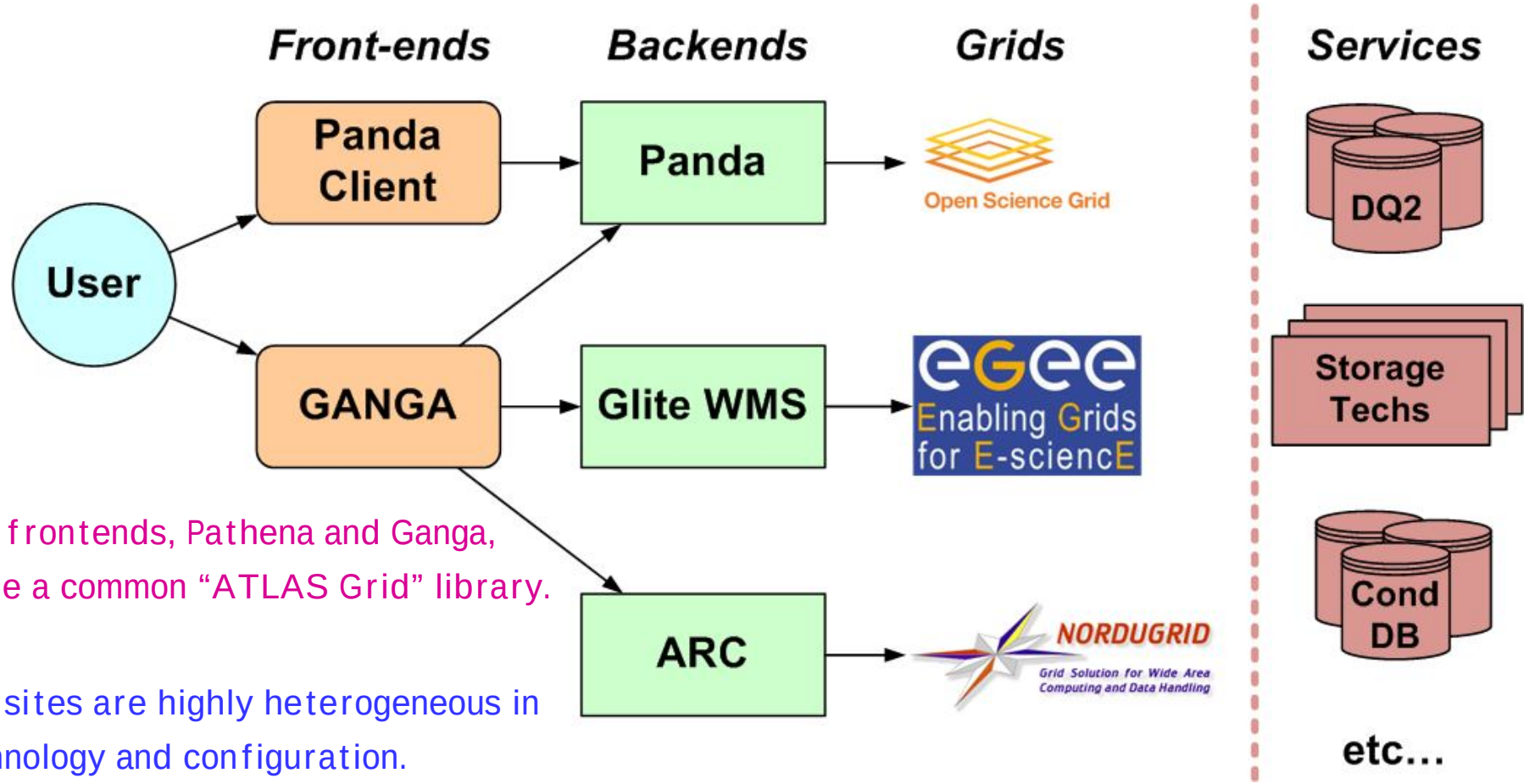
- Overview of Distributed Analysis in ATLAS:
  - What needs to be tested? Workflows and Resources
- Functional Testing with GangaRobot:
  - Daily short tests to verify the analysis workflows
- Stress Testing with HammerCloud:
  - Infrequent (~weekly) large scale tests to stress specific components

DA in ATLAS: What can the users do?

- The ATLAS Distributed Computing (ADC) operational situation in a nutshell:
  - The grids and resources are established.
  - Distributed production and data distribution is well understood and tested.
  - Now, the priority is on **validating distributed analysis for users**

- **What do the users want to do?**
  - **"What runs on my laptop should run on the grid!"**
- Classic analyses with Athena and AthenaROOTAccess:
  - A lot of MC processing, cosmics, reprocessed data
  - Various sizes of input data: AOD, DPD, ESD
  - TAG analyses for direct data access
- Calibrations & Alignment: RAW data and remote database access
- Small MC Sample Production: transformations
- ROOT: Generic ROOT application also with DQ2 access
- Generic Executables: for everything else

DA in ATLAS: What are the resources?



The frontends, Pathena and Ganga, share a common "ATLAS Grid" library.

The sites are highly heterogeneous in technology and configuration.

How do we validate ATLAS DA?

Use case functionalities?? Behaviour under load??

ATLAS DA Operations Activities

- This talk presents two activities to work on these problems:
  - Functional Testing with GangaRobot
  - Simulated Stress Testing with HammerCloud

- The third piece of the puzzle, not covered in this talk, is what we call the "Distributed Analysis Jamboree":
  - Coordinated stress test with real users, real analyses, and generating *real* chaos.
  - US has some experience of this type of test, and worldwide distributed analysis jamborees are being organized right now.

Functional Testing with GangaRobot

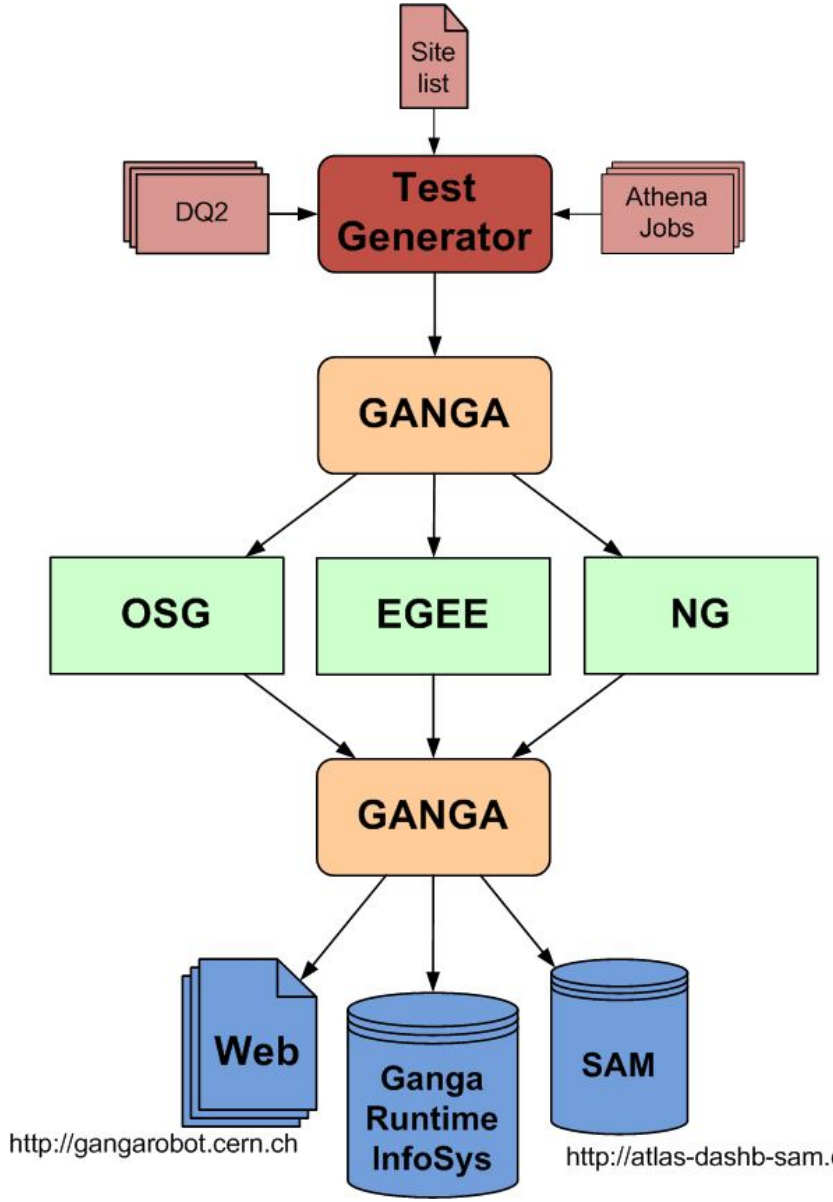
- Definitions:
  - **Ganga** is a distributed analysis user interface with a scriptable python API.
  - **GangaRobot** is both
    a) a component of Ganga which allows for rapid definition and execution of test jobs, with hooks for pre- and post-processing
    b) an ATLAS service which uses (a) to run DA functional tests

    ➢ In this talk, **GangaRobot** is (b).

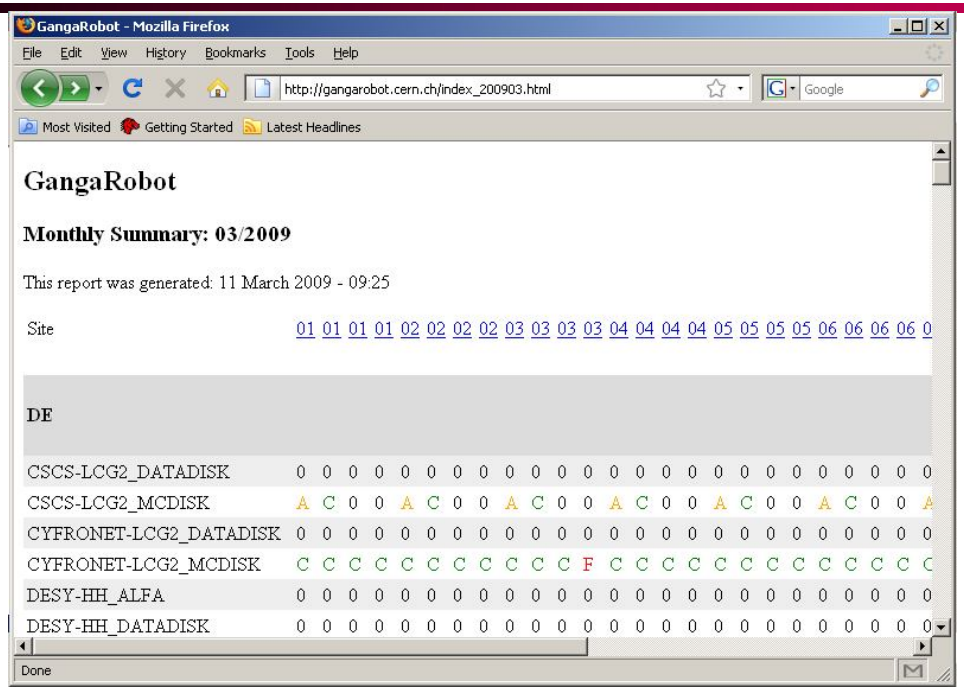
- So what does GangaRobot test and how does it work?

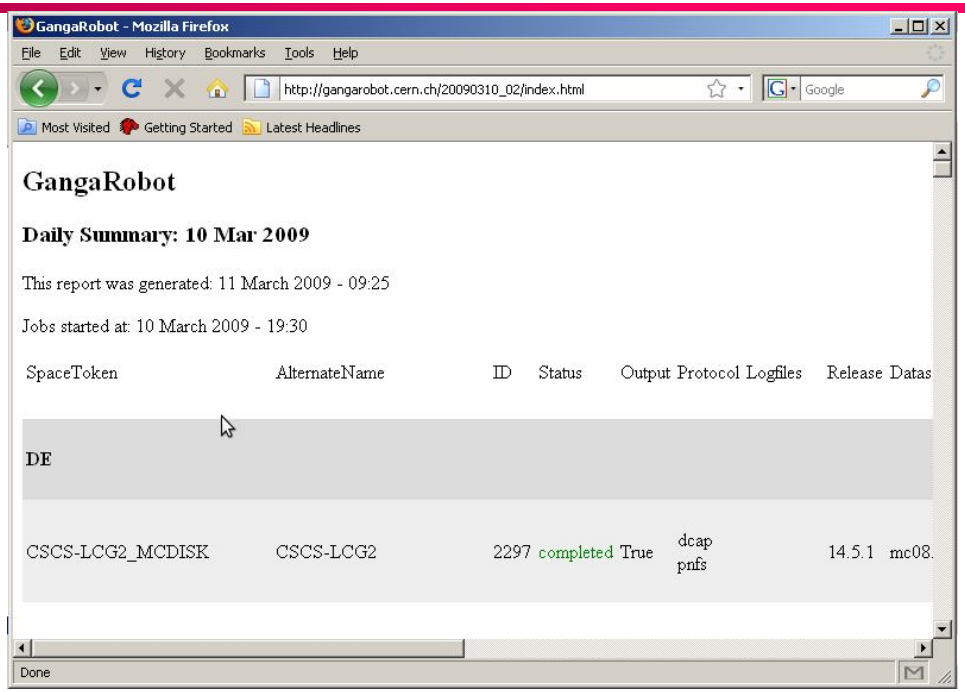
Functional Testing with GangaRobot

1. Tests are defined by the GR operator:
   - Athena version, analysis code, input datasets, which sites to test
   - Short jobs, mainly to test the software and data access
2. Ganga submits the jobs
   - To OSG/Panda, EGEE/LCG, NG/ARC
3. Ganga periodically monitors the jobs until they have completed or failed
   - Results are recorded locally
4. GangaRobot then publishes the results to three systems:
   - Ganga Runtime Info System, to avoid failing sites
   - SAM, so that sites can see the failures (EGEE only, OSG in deployment)
   - GangaRobot website, monitored by ATLAS DA shifters
     - GGUS and RT tickets sent for failures



What happens with the results?

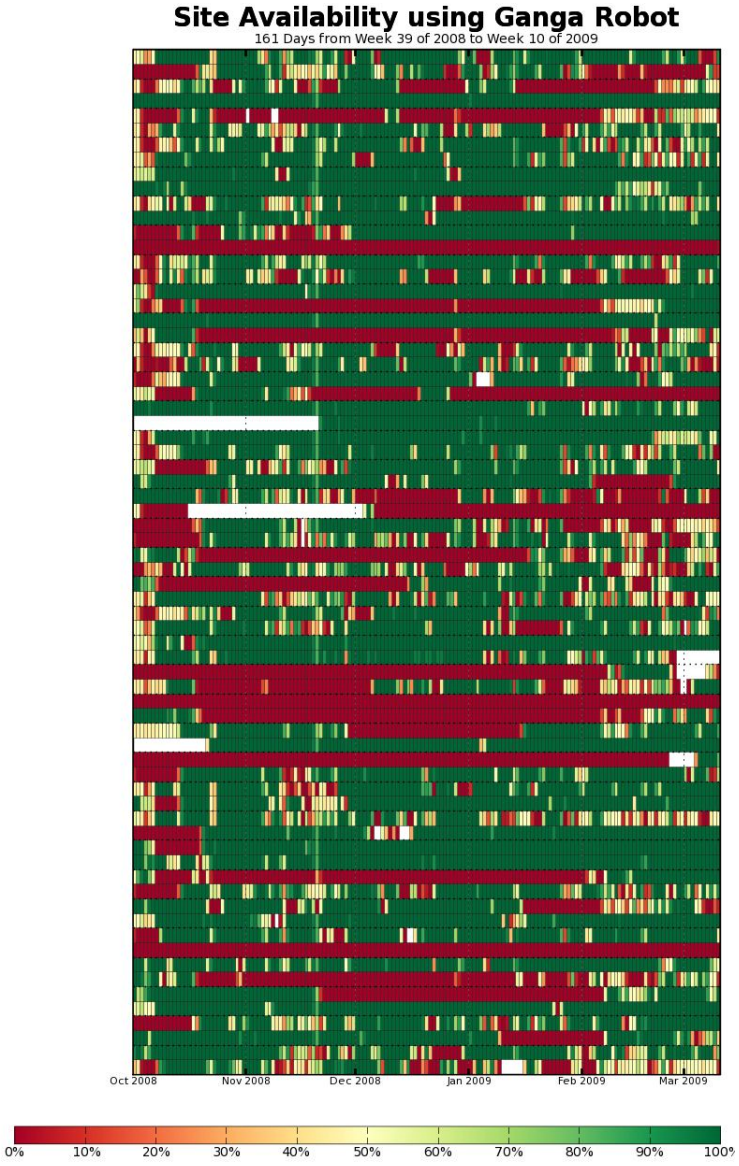


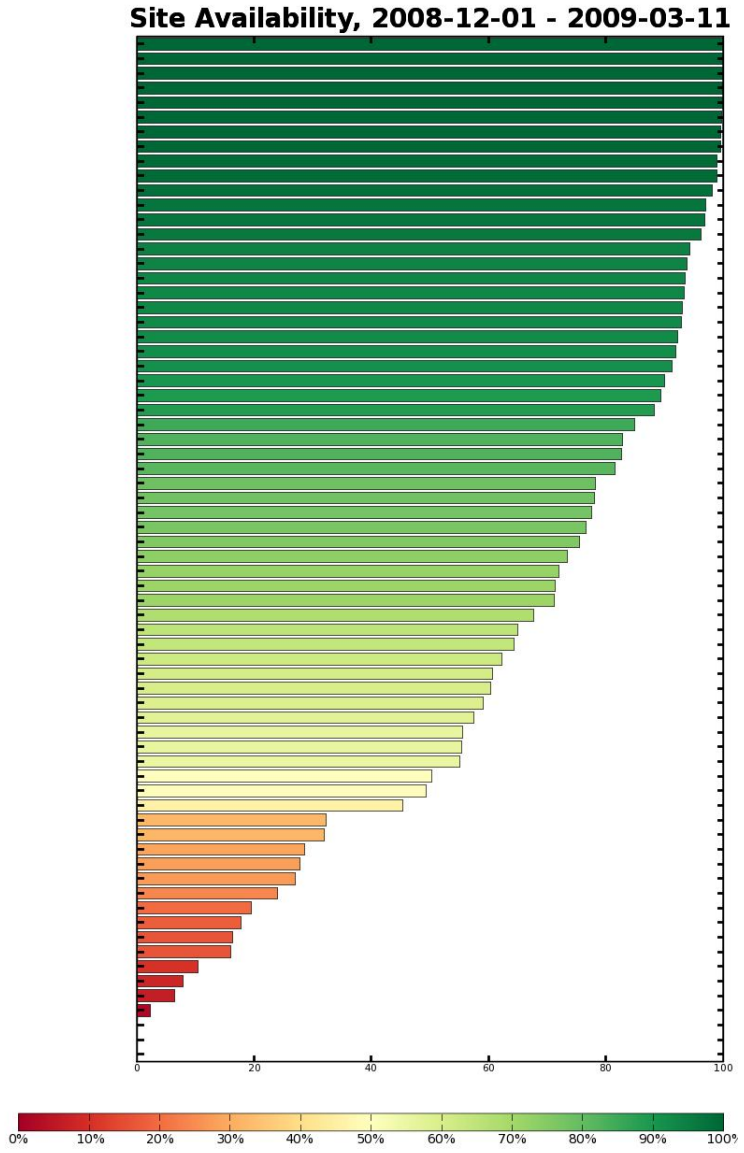


- The analysis tools need to avoid sites with failed tests:
  - For Ganga/EGEE users, feeding the results to the Ganga InfoSys accomplishes this.
  - For OSG and NG the sites are set offline manually by a shifter
  - In future, the results need to go to the planned central ATLAS InfoSys (AGIS)

- Results need to be relevant and clear so the problems can be fixed rapidly:
  - GangaRobot website has all the results… but causes information overload for non experts
  - SAM is more friendly and better integrated, but doesn't present the whole picture (and A.T.M. includes only EGEE).

Overall Statistics with GangaRobot





Plots from SAM dashboard http://dashb-atlas-sam.cern.ch/ of daily and percentage availability of EGEE sites over the past 3.5 months.

WARNING:
Don't automatically blame the sites! The fault could lie anywhere in the systems.

The good: Many sites with >90% efficiency

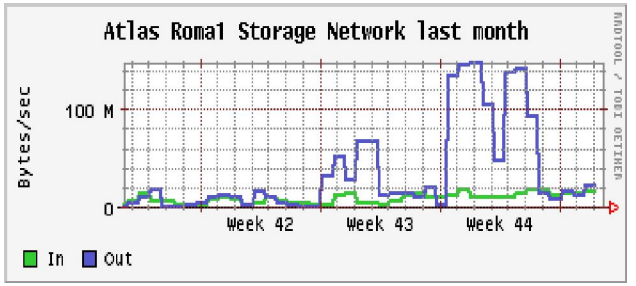
The bad: Less than 50% of sites have >80% uptime

The expected: Many transient errors, 1-2 day downtimes. A few sites are permanently failing.

Distributed Analysis Stress Testing

- The need for DA stress testing:
  - Example I/O rates from a classic Athena AOD analysis:
    - A fully loaded CPU can read events at ~20Hz (i.e. at this rate, the CPU, not the file I/O, is the bottleneck)
    - 20Hz * 0.2MB per event = 4 MB/s per CPU
    - A site with 200 CPUs could consume data at 800MBytes per second
    - This requires a 10Gbps network, and a storage system that can handle such a load.
  - Alternatively, this means that 200 CPU cluster with a 1Gbps network will result in ~3Hz per analysis job

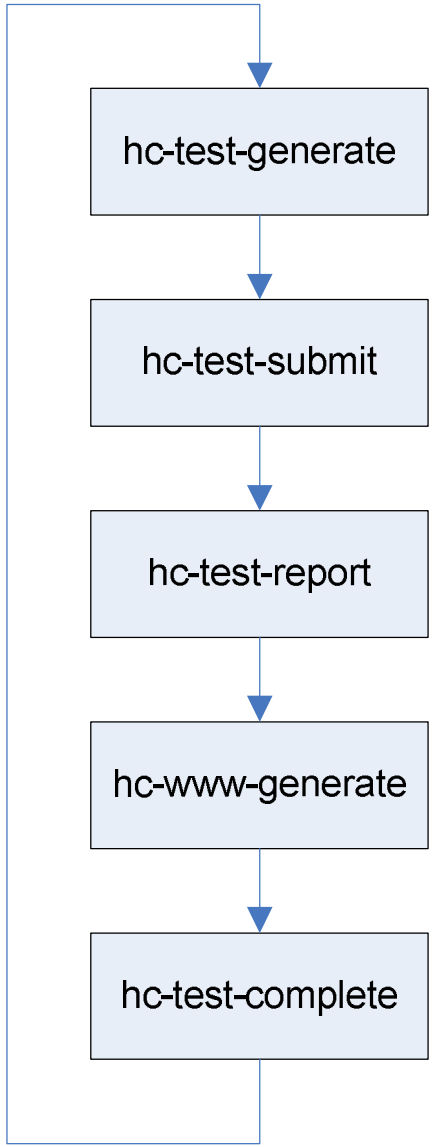
- In fall 2008, clouds started getting interested in testing the Tier 2s under load
  - The first tests were in Italy, and were manual:
    - 2-5 users submitting ~200 jobs each at the same time
    - Results merged and analyzed 24-48 hours later
    - The IT tests saturated 1Gbps networks at the T2 sites, resulting in <3Hz per job.



- From these early, we saw then need for an automated stress testing system to be able to simultaneously test all clouds: hence, we developed **HammerCloud**

HammerCloud: How it works?

- What does HammerCloud (HC) do?
  - An operator defines the tests:
    - What: a ganga job template, specifying input datasets and including an input sandbox tar.gz (athena analysis code)
    - Where: list of sites to test, number of jobs
    - When: start and end times
    - How: input data I/O (posix I/O, copy locally, or FileStager)
  - Each job runs athena over an entire input dataset. The test is defined with a dataset pattern (e.g. mc08.*.AOD.*), and HC generates one job per dataset.
    - Try to run with the same datasets at all sites, but there are not always enough replicas.
  - HammerCloud runs the tests:
    1. Generate appropriate jobs for each site
    2. Submit the jobs (LCG and NG now; Panda and Batch coming)
    3. Poll their statuses, writing incremental results in HC DB
    4. Read HC DB to plot results on web.
    5. Cleanup leftovers; kill jobs still incomplete
  - When running many tests, each stage handles each test sequentially (e.g. gen A, gen B, sub A, sub B,…)
    - This limits the number of tests that can run at once.



HammerCloud: What are the tests?

- HammerCloud tests real analyses:
  - AOD analysis, based on Athena UserAnalysis pkg, analyzing mainly muons:
    - Input data: muon AOD datasets, or other AODs if muons are not available
    - In principal, the results would be similar to any analysis where the file I/O is the bottleneck.
  - Reprocessed DPD analysis:
    - Intended to test the remote conditions database (at local Tier 1)

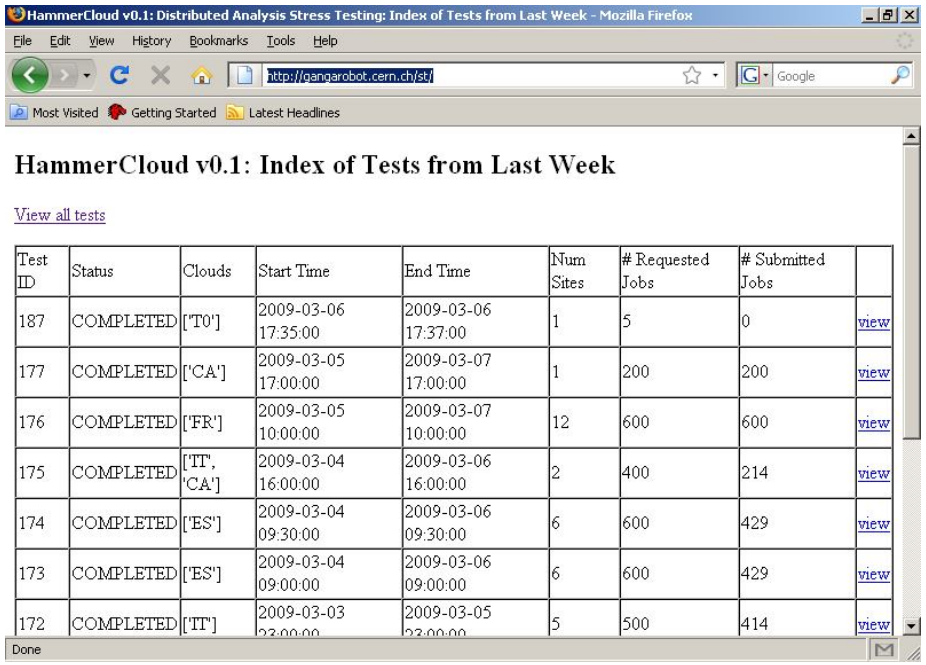
- What metrics does HammerCloud measure?
  - Exit status and log files
  - CPU/Wallclock ratio, events per second
  - Job timing:
    - Queue, Input sandbox stage-in, Athena/CMT setup, LFC lookup, Athena exec, Output storage
  - Number of events and files processed (versus what was expected)
  - Some local statistics (e.g. network and storage rates) are only available at site level monitoring
    - Site contacts very important!

HammerCloud: What are the tests? (2)

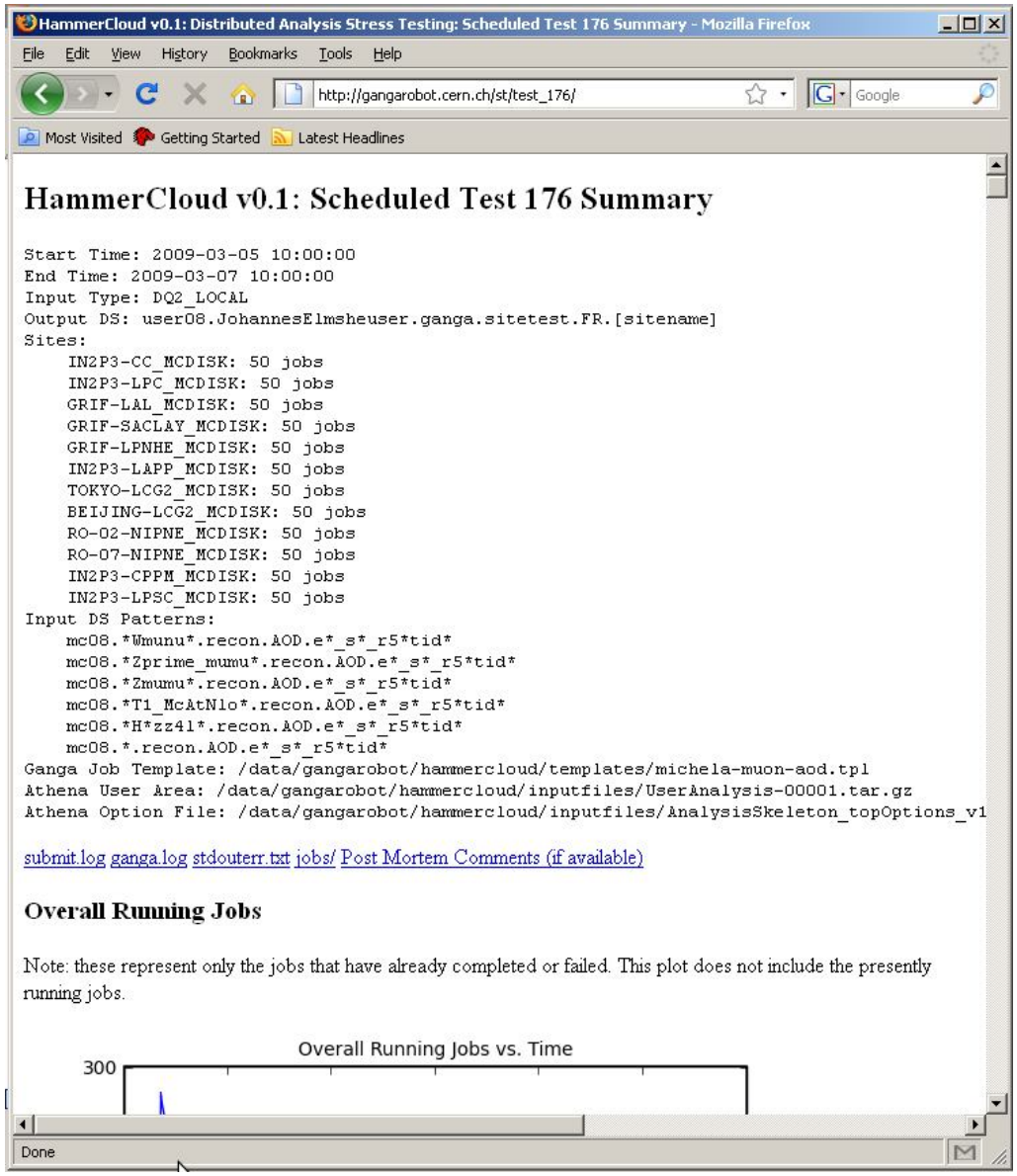
- Up until now, the key variable that HammerCloud is evaluating is the data access method:
  - Posix I/O with local protocol:
    - To tune rfio, dcap, gsidcap, storm, lustre, etc…
    - Testing with read-ahead buffers on or off; large, small or tweaked.
  - Copy the files locally before running
    - But disk space is limited, and restarting athena causes overhead
  - Athena FileStager plugin:
    - Uses a background thread to JIT copy the input files from storage
    - Startup – Copy f1 – Process f1 & copy f2 – Process f2 & copy f3 – etc…

HammerCloud Website

http://gangarobot.cern.ch/st/

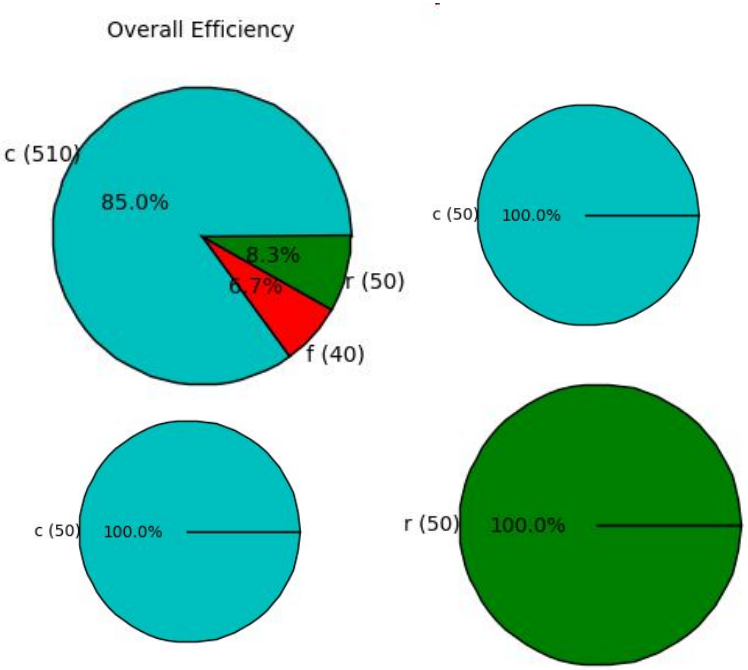


Results from all tests are kept indefinitely.



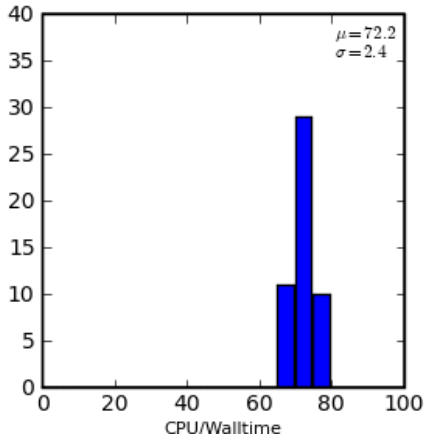
Example HammerCloud Test Results

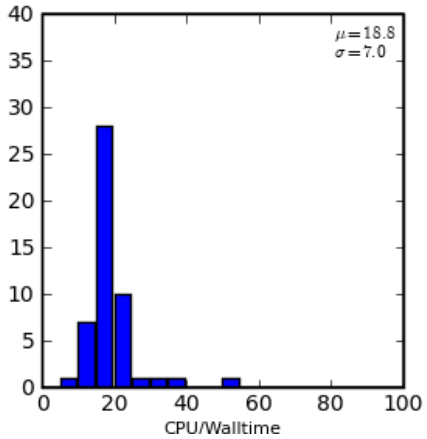
600 jobs across 12 sites
~50 million events, ~20000 files



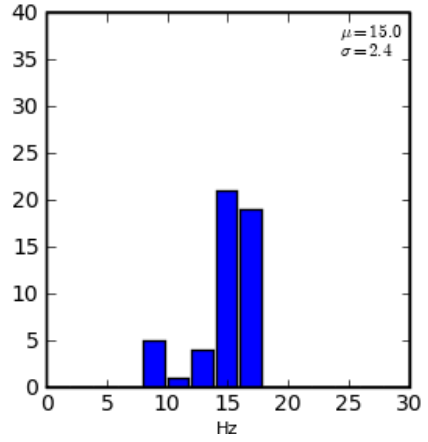
7 sites had no errors

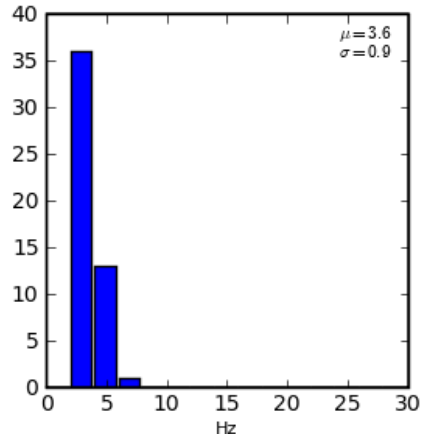
% CPU Used





Events/second





But, beware hidden failures! Did the job actually process the files it was supposed to? No, only 92% of the files that should have processed were… the other 8%? See later.

Overall HammerCloud Statistics

- Throughout the history of HammerCloud:
  - 74 sites tested; nearly 200 tests; top sites tested >25 times
  - ~50000 jobs total with average runtime of 2.2 hours.
  - Processed 2.7 billion events in 10.5 million files
- Success rate:
  - 29 sites have >80% success rate; 9 sites >90%
- Across all tests:
  - CPU Utilisation: 27 sites >50% CPU; 8 sites >70%
  - Event rate: 19 sites > 10Hz; 7 sites >15Hz
- With FileStager data access mode:
  - CPU Utilisation: 36 sites >50%; 24 sites >70%
  - Event rate: 33 sites > 10Hz; 20 sites > 15Hz; 4 sites >20Hz

NOTE: These are overall summaries without a quality cut; i.e. the numbers include old tests without tuned data access.

- Full statistics available at: http://gangarobot.cern.ch/st/summary.html

What have we learned so far?

- The expected benefits:
  - We have found that most sites are not optimized to start out, and HC can find the weaknesses.
    - The sites rely on large quantities of jobs to tune their networks and storage
  - HammerCloud is a benchmark for the sites:
    - Site admins can change their configuration, and then request a test to see how it affects performance
  - We are building a knowledge base of optimal data access modes at the sites:
    - There is no magic solution w.r.t. Posix I/O vs. FileStager.
    - It is essential for the DA tools to employ this information about the sites.

What have we learned so far? (2)

- The unexpected benefits:
  - Unexpected storage bottlenecks (hot dataset problem):
    - In many tests, we found that the data was not well distributed across all storage pools, resulting in one pool being overloaded while the others sat idle.
    - Need to understand how to balance the pools
  - Misunderstood behaviour of distributed data management tools:
    - The DB access jobs require a large sqlite database to be dq2-get'd before starting. It was not known that the design of dq2-get did not retrieve from a close site.
    - A large test could have brought systems down (but this was caught before the test thanks to a friendly user).
    - Ganga's download of the sq2lite DB was changed (as was dq2-get's behaviour).
  - Found athena I/O bug/misunderstanding:
    - HC found discrepancies in the number of files intended to be and actually processed.
    - We found that athena, in the case that file open() times out, would exit with error status 0 and "success".
    - Behaviour was changed for Athena 15.

Next Steps

- GangaRobot Functional Testing TODO list:
  - Technical improvements:
    - More tests: enumerate the workflows and test them all
    - Better integration with SAM/dashboards/AGIS: add non-EGEE sites !!
  - Procedural improvements:
    - Need more effort to report to and fix the broken sites

- HammerCloud Stress Testing TODO list:
  - V0.2 is ready, pending verification:
    - New testing model (continuous parallel tests) that will allow upward scaling
    - Advanced booking of repeated (e.g. daily/weekly) tests
  - Implement testing on Panda & Batch backends:
    - Testing on Panda is the top priority.
  - More metrics, improved presentation, correlation of results
    - We have more than 60GB of logfiles… any data miners interested?
  - Make it more generic with support for other VOs:
    - LHCb testing would be rather simple

Conclusions

- Validating the grid for user analysis is a top priority for ATLAS Distributed Computing
  - The functionalities available to users are rather complete, now we are testing to see what breaks under full load.
- GangaRobot is an effective tool for functional testing:
  - Daily tests of the common use cases are essential if we want to keep sites working.
- HammerCloud is a relatively new tool; there is a lot of work to do.
  - Many sites have improved their networks and storage configurations
  - ATLAS-wide application of these tests are the top development priority.