Imperial College
London

LHCb plans for distributed analysis and the rôle of GANGA

22 March 2004

LHCb plans for distributed analysis

- LHCb analysis is based on the Gaudi framework.
- Users *do* their analysis by:
  - writing new Gaudi algorithms;
    - either C++ or plain Python
  - modifying exsisting algorithms;
  - modifying job options of existing/new algorithms.
- LHCb real and simulated data will be distributed mainly at Tier 1 centres in different countries.
  - In addtion model should support local data as well as data at Tier 2 centres.
- The input data will be data in POOL format.
- Output from distributed analysis will be a combination of data in POOL format, ROOT/HBOOK files and stdout/stderr.

Distributed analysis goals for 2004

- The goals for 2004 were defined at workshop in October 2003.
  - Only minor changes in plans since then.
- The idea is to demonstrate a working model of distributed analysis.
- Demonstration requires:
  - Many users of system.
  - Scope set correctly to have a product in place.
- Distinct data will be kept at Tier 1 centres (and not Tier 2).
  - Not strictly required but gives several advantages:
  - Force a distributed analysis by having everything nowhere.
  - Will minimise need of support as only 8 specified centres involved.
  - Model will allow short cuts (like releases being pre-installed and not automatically by analysis jobs arriving).

Where GANGA fits in

- GANGA is in LHCb seen as **the** user front end for distributed analysis.
  - No system for distributed analysis directly from C++ (like the ROOT prompt) foreseen.
  - The distributed analysis should be the default way for a user to do analysis. Not just a demonstration system for a few experts.
- The distributed analysis system for LHCb and not necessarily the production system.
  - A production system (mainly large scale, complicated workflow, expert user) and an analysis system (mainly small scale, single task, novice) share requirements they have many differences.

Requirements on GANGA

- Look and feel
  - The user should see GANGA as a single application.
  - Only a need to learn one part of GANGA to get a feeling for the rest
  - Scripting (in Python) should have one-to-one correspondence with buttons pressed in GUI.
- The user client part of GANGA should be local.
  - Both a system and an unprivileged user installation should be supported.
  - Should run on multiple linux flavours and Windows.
- Configuration
  - The configuration required to run GANGA should be relatively simple.
- Support
  - Again GANGA should be seen as a single application. Not an ensemble of individually supported applications.

Requirements on GANGA

- For GANGA to become a success we need
  - Stable application.
  - Clear error messages.
  - Good response time.
  - Limited number of dependencies.

Job splitting and merging.

- Job splitting should be implemented as a *job splitter* such that the user only specifies a given logical dataset to run on.
  - Job splitter will for 2004 not be required to deal with situation where logical data is spread across several Tier 1 centres.
- Some generic merging (or post processing) algorithms should be written.
  - Should be user driven (ie happen when a user request it)
  - Needs to be highly configurable
  - Should provide for:
    - A simple list pointing to all output data files.
    - Merge histogram files.
    - Create a macro that chains together output N-tuples.
    - Pick selected lines from ASCII output and merge them.

Access to data at worker node

- When the input data is selected in GANGA it will be based completely on logical file names (LFN).
- When Gaudi job is running on WN it will also simply open a LFN.
  - For DC04 the conversion from LFN to physical file name at WN location will use an XML slice of the file catalogue.
  - A service is required to create this slice given a set of LFNs.

Persistency and provenance

- A user should be able to start/stop GANGA session at any given time.
  - State should be preserved between sessions.
  - State should be preserved when starting GANGA on different machine.
    - Not the highest priority for 2004.
- Exchange of jobs between users
  - Analysis is often carried out in small teams where jobs and output are shared/exchanged.
  - GANGA should support this kind of collaboration at the job level.
    - Job import/export or definiton of shared jobs?
- Provenance
  - The history of jobs needs preserving from selected input data to the merged output.
  - Important not to costrain users by too rigid system.

Job configuration