Comments about PROOF / ROOT evolution during coming years

G. Ganis
14 Dec 2006

PROOF

The current plan of work contains essential milestones and its completion will take at least next year:

- full control in case of failures (error handling)
- optimized responsiveness in multi-user environment (scheduling)
    - come-and-go functionality for workers

Beyond this, non-trivial items deserving resources are:

- Technical
    - dynamic parameter setting
    - development of a test suite
    - optimizations for local sessions
- General
    - full transparency

PROOF – technical issues

- dynamic parameter setting
    - input list is static, session-oriented
    - possibility to parse a list of arguments will be more user-friendly

- development of a test suite:
    - Difficult: complex system spread over network
    - … but essential (especially in production):
        - we often break things now because we do not have it

- optimizations for local sessions
    - no need of daemons or masters with local workers
    - can fork directly the worker processes
        - client can also act as master
    - UNIX sockets to optimize transfers between local processes
    - important in view of multi-core processors
    - can act as code-tester

PROOF – final goal (dream): full transparency

- Transparency means that there should be as little difference as possible between a local ROOT session and PROOF

- Not really the case now
  - e.g.:TSelector+TChain+SetProof() not guaranteed to work on PROOF

- By its nature, PROOF should be able to speed-up processing of any *job* consisting of uncorrelated steps
  - this is what many people hearing about PROOF expect

- Ideally, anything *parallelizable* would be automatically processed in parallel
  - e.g. can

```
root[] .x MyLoop.C(10000)
```

go automatically on the default PROOF session?

PROOF – final goal (dream): full transparency (cnt'd)

- Need for a more general framework incorporating TSelector+TChain as special case ?
- Based on TTask ?

- We should address first the obvious case of TChain (see above)
- Learn from user's feedback how far we should go

ROOT

- Big project, in healthy state
- One of its main goals was to provide an efficient tool for next generation (wrt LEP) experiments
  - serving correctly the needs of LHC is (will continue to be) a target
  - ongoing testing, validation by LHC experiments are of utmost importance
    - output will have an impact on the priorities on the project

- Difficult to say now what is missing (if any) outside what is planned
- As analysis progresses new tools will be developed and should be integrated
  - as it was for LEP, but there was no common, open framework where to plug-in new things

- Tendency to think that graphical tools are an option but when we come to final results everybody asks for more features (see PAW at LEP).

ROOT – technical issues

- Thread-safeness of CINT is very important

- Need to rationalize our view of data sets
    - In PROOF the apparent dualism TChain-TDSet creates confusion
    - TDSet is in an extension of TChain, not an alternative
        - good starting point for the discussion

- Packaging issues
    - larger (better) granularity for shared libraries
    - separate library for dictionaries?