Distributed storage works update

SuperB Collaboration Meeting
Pisa, Sept. 20th 2012

Paolo Franchini (CNAF)
for the distributed storage group

Distributed storage R&D

- Data access dedicated library development
- New generation mass data transfer system (FTS3 evaluation)
- New generation file catalog solutions
- Geographically distributed data center (design study)
- Data model definition study

WAN data access

- Experiment use cases:
  - interactive usage of SuperB data
  - analysis code writing and debugging
  - analysis tasks executed on non SuperB resources
  - job execution on small sites like Tier3s
  - safeness in case of storage failure
- Network protocols
  - xrootd and HTTP:
    - support posix-like calls
    - capabilities of work through routers and firewalls
    - caching and pre-fetching mechanism
    - supported by ROOT framework

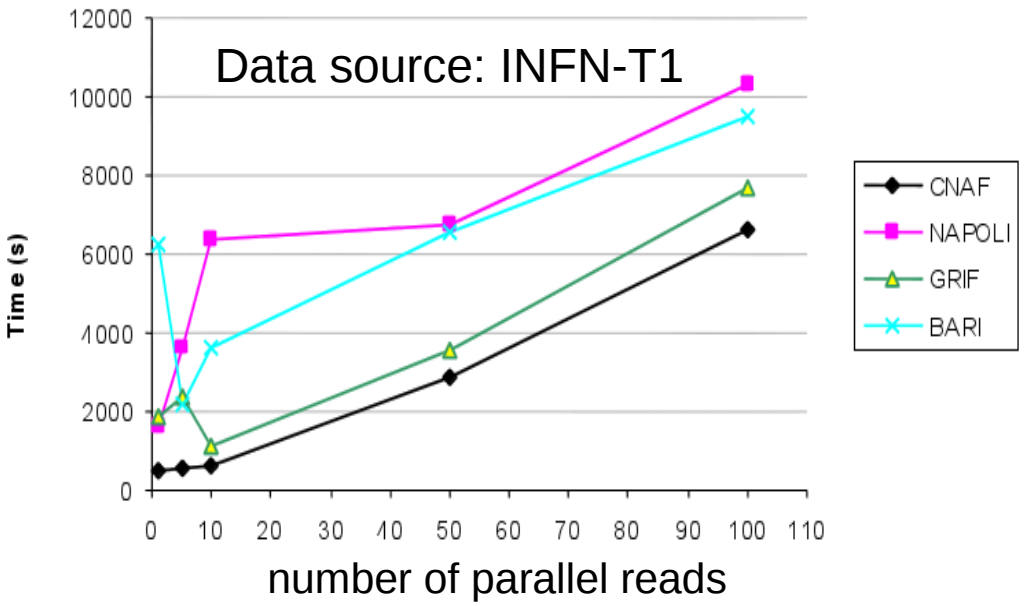
HTTP data access test

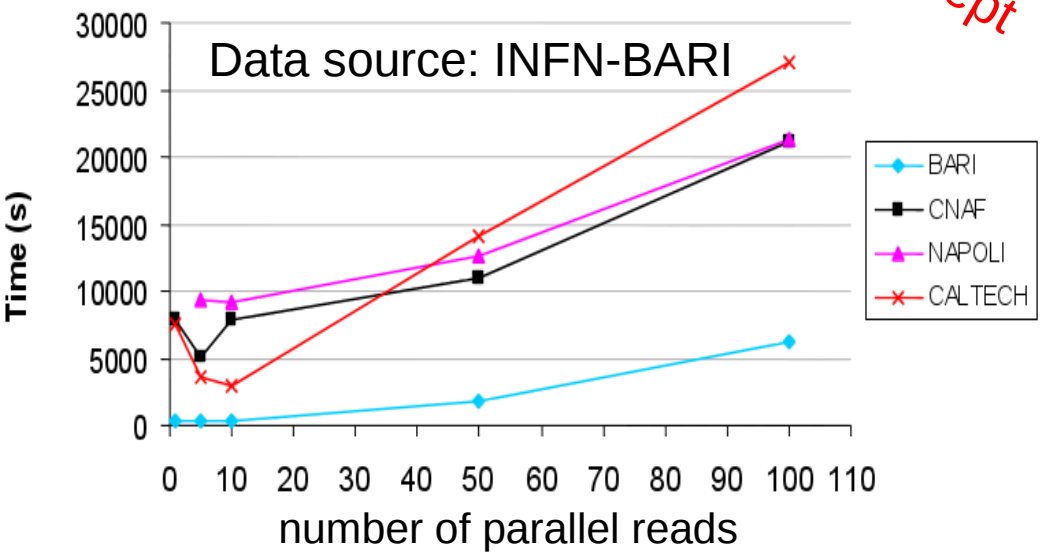
proof-of-concept

- Test goals:
  - measure the latency period due to the increase number of parallel read stream
  - measure the latency period due to the increase of round trip time elapsed between source and destination
  - support the development of a general, experiment wide, data access software layer
  - start the characterization of a concrete WAN scenario, including traffic impact, typical latency, network resource overloading
- Test layout definition:
  - 1, 5, 10, 50 and 100 parallel set of read streams
  - each stream reads a random files according to a trace file obtained from an analysis application
  - 250 compressed root files, 500 MB each
  - sources: INFN-T1 and INFN-Bari
  - destinations: INFN-T1, INFN-Bari, INFN-Napoli, GRIF, Caltech
  - measured the time of the cURL execution

HTTP data access test: results

proof-of-concept





- The network latency influences the read stream operations for all the routes
- The link congestions affect the case of single read-stream also on short routes
- The dips of the curves can be the effects of a specific link-to-link overload

Data access library

- R&D work for the development of a library permitting an optimized data access management
- Features:
  - intelligent pre-fetching and buffering algorithms
  - logical file name map with different physical storage URI
  - possibility of support to unsupported ROOT storage protocols
  - read-head buffer and caching mechanism in order to solve the overhead

Library approaches

- **High level**: library wrapper of ROOT data access/download methods
  - pros: simple sbROOT deployment, coherent lib insertion in experiment framework, user learning curve minimized in terms of simple code package linking
  - cons: users need to stop using ROOT primitives, old libraries/code should be updated to include new SuperB custom data access methods
- **Low level**: new file protocol developing a ROOT class
  - pros: code and development users practices do not need to be modified
  - cons: need an ad-hoc ROOT implementation, with the risk of including functionalities out of the scope of a protocol implementation
- **Configuration driven**: need a ROOT configuration interface in order to change the data access according to a set of parameters.

Library approaches

- Philippe Canal, from the ROOT team, suggested to implement the low level approach with a new protocol
  - In this case the library code has to be distributed external from a ROOT distribution
  - Users have to load the library and a configuration file
- The initial idea was to implement the library within the software framework.

Library state-of-art: `libSbNet`

- The library input is the catalog name (`lfn://`) of the file that must be used in the analysis
- The library output is the local file name (`file://`) that ROOT can use in the analysis
- In order to obtain the output the library first checks the default storage element, defined in the environment variable `VO_SUPERBVO_ORG_DEFAULT_SE`.
- If this SE returns a valid `file://` TURL the work is done…

Library state-of-art: `libSbNet`

- If the default SE has some problem the library uses the lcg API (`lcg_lr`) to obtain the list of the file replicas, sorts them according to route distance, then asks every SRM a valid `file://` TURL
- If the `file://` protocol is not available, then falls down to slower protocols, first `http://` (not yet implemented) and eventually `gsiftp://`
- In the `gsiftp://` case the lib copies the file locally and returns its full local path at the caller.
- It’s up to the lib caller to delete this temporary file at the end of its use.

Hadoop testing: activity status

- Developed a policy to distribute data among different farm
  - Under test on a geographically distributed cluster between INFN-Bari and INFN-Naples
  - Using Cloudera HADOOP (HDFS-2.0)
- Testing of the new version of HADOOP 2.0
- Testing of the Kerberos security on HADOOP
- Monitoring:
  - Configuration of Ganglia plugin
  - Development of ad-hoc script for monitoring the data location
- Developing automatic script for installing and configuring HADOOP

Hadoop testing: next steps

- Adding another site to the geographical testbed
- Performance and stress test within a single farm up to 250 worker nodes
- Performance and stress test among geographical sites
- Testing the reliability of geographical automatic replica

Future plans

- Library development: implement the protocol identification, data access optimizations, integration in ROOT system
- Wan data access: setup of a new testbed using the library with more statistic and more control, using HTTP and xrootd.
- HadoopFS on WAN: perform a complete performance and functionality test → full report for the December CM
- FTS3 test with RAL service setup