

# The Phobos File Catalog and Distributed Disk

Maarten Ballintijn / MIT

4-Sep-2002

# Agenda

- Goals and Constraints
- Architecture
- Technologies
- Current Status
- Future Developments
- Related Technologies

# Goals and Constraints

- Goals for Distributed Disk
  - Distribute I/O
    - Limits in disk speed, network speed
  - Use cheap disk
  - No single point of failure
  - PhAT (batch), PROOF (interactive)

# Goals and Constraints

- Goals for File Catalog
  - Automate handling of a few  $10^5$  files
    - Provide a catalogue
    - Stageing (of collections)
  - Transparent use of multiple disk pools
  - Optimize disk usage
  - Implement policies
  - ‘‘Scatter’’ files

# Goals and Constraints

- Constraints
  - Environment
    - Secure Linux clusters in different domains
    - HPSS interface
    - International collaboration
  - Manpower restriction
    - Implement essential features only
    - Aim for low maintenance
  - Reuse vs. build

# Architecture

- HPSS
  - Use as backing for all files
  - Interact via FTP interface
- Disk Pools
  - Stores Instances (copies of Files in HPSS)
  - Collection of disk storage (in a cluster)
  - Managed as single entity, one File Daemon

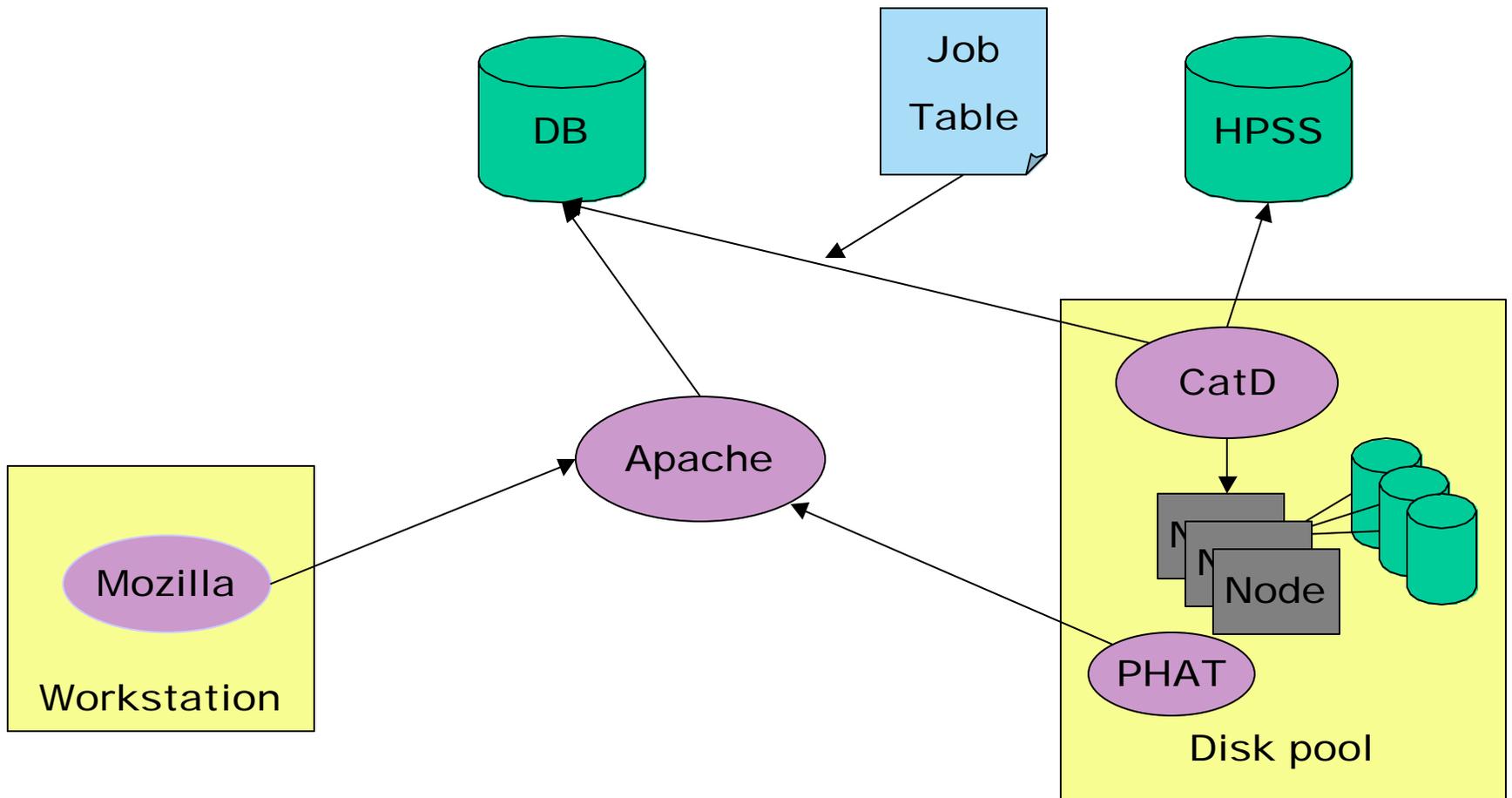
# Architecture

- Database
  - Persistent storage of File meta data
  - Configuration data
  - Interface between UI and File Daemons
- API
  - Translate Filename to physical location  
(“root://rcas4014//data1/PhoHit1234s12.root”)
  - To be extended with more functionality

# Architecture

- Web Application
  - User Interface
  - Policy decisions
- File Daemon
  - Schedules and Executes file manipulations and monitoring for a pool
  - Use plugin scripts for actual operations
  - E.g Plugins know to get a file from HPSS

# Architecture



# Technologies

- Perl
  - Well suited for systems programming
  - Many available modules (mod\_apache, DBI)
- Mason
  - Template system for web applications
  - Provides caching
- DBI
  - Database independence
  - Apache persistent DB connection support

# Technologies

- Apache
  - The standard solution
  - Integrates well with the other components
- Oracle
  - Reliable, versatile, fast enough
  - Leverage existing knowledge and infrastructure
- SSH
  - Provide secure communications
  - Leverage existing infrastructure

# Current Status

- In production since April 2002
- Currently Implemented:
  - Authenticated access, authorization levels
  - Querying and browsing of Files, Instances, Pools
  - Stageing of Files from HPSS or other pools
  - Deleting Instances
  - API for instance lookup

# Current Status

- Issues and missing features
  - HPSS synchronization
  - Oracle connectivity
  - Help system
  - Job scheduling

# Current Status

## Files registered in the Catalog

<b>Type</b>	<b># Files</b>	<b>Size (Gb)</b>
PhoRaw	25077	20133
PhoHit	47078	31133
PhoTrkDST	34758	2526
PhoSmMCHits	12785	1397
<i>Total</i>	<i>119698</i>	<i>55189</i>

# Current Status

- 3 Pools defined
  - RCF: 153 disks / 17 Tb , 79% full
  - Pharm: 18 disks / 983 Gb, 82% full
  - Pdev: 12 disks / 1 Tb
- Instances
  - 35,000 Instances
  - 7,000 not yet used
  - 130,000+ TPhDST requests (avg 900 / day)
  - 1 Instance was requested 202 times!

# Future developments

- Improve HPSS synchronization
- New API
  - TPhGrid based on TGrid virtual interface
  - Web services (HTTP, XML-RPC, SOAP)
- Improve UI and extend functionality
- Automatic File migration, space reclaim

As required by users and data volume ...

# Related Technologies

- PROOF – Parallel Root Facility
  - Bring the KB to the PB not the PB to the KB*
  - (more on the next slides)
- GRID Computing
  - Authentication, Authorization, Single Sign-on
  - File Catalogues, Replication services
  - Resource brokers

Migrate to fully GRID based infrastructure over time

# PROOF – Parallel ROOT Facility

- Collaboration between core ROOT group at CERN and MIT Heavy Ion Group
  - Rene Brun
  - Gunther Roland
  - Fons Rademakers
  - Maarten Ballintijn
- Part of and based on ROOT framework
- Currently no external Technologies
- ROOT since 1995, PROOF started 2001
- A wealth of info at <http://root.cern.ch>

# PROOF Architecture

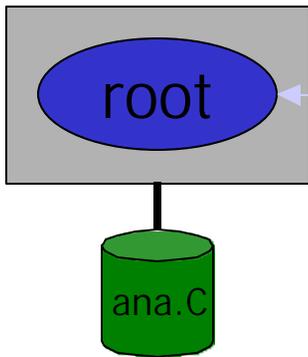
- Data Access Strategies
  - Local data first, also **rootd**, SAN/NAS
- Transparency
  - Input Objects copied from Client
  - Output Objects merged, returned to Client
- Scalability and Adaptability
  - Dynamic packet size (specific workload, slave performance, dynamic load)
  - Heterogeneous Servers
- Migrate to multi site configurations

# Parallel Analysis of Remote Data

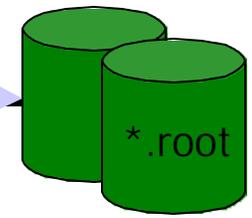
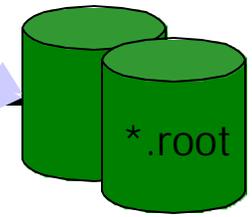
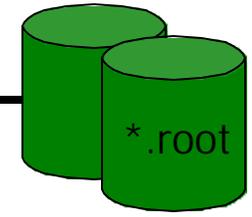
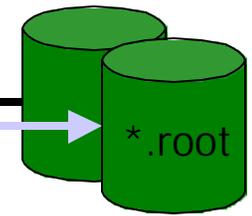
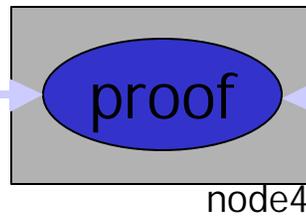
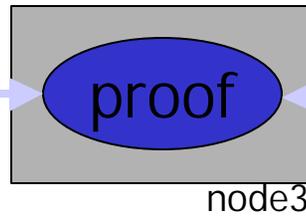
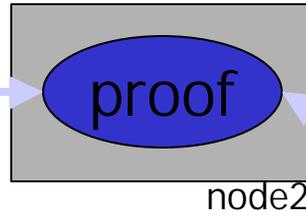
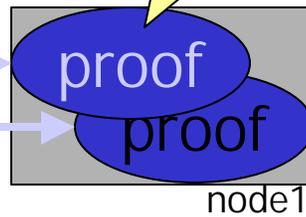
Local PC

Remote Data Cluster

```
#proof.conf  
slave node1  
slave node2  
slave node3  
slave node4
```



← stdout/dbj  
ana.C →



```
$ root  
root [0] tree.Process("ana.C")  
root [1] gROOT->Proof("remote")  
root [2] dset->Process("ana.C")
```

proof = master server  
proof = slave server

# PROOF Future

- Introduce GRID services
- Implement GRID driven configurations
- Scale to multi site, WAN, proof clusters
- Layer services on basic PROOF layer

This slide intentionally left blank