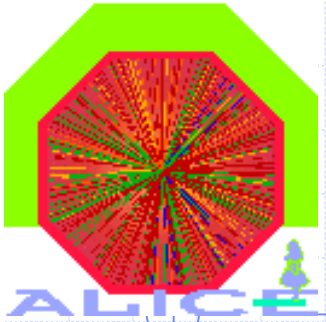


AliEn

@GRID

P.Saiz, J-E. Revsbech, P.Buncic

@GRID

Introduction

- ALIEN stands for ALIce production ENvironment
- It should provide everything that AliRoot framework requires to carry out
  - Simulation
  - Reconstruction
  - Event mixing
- It is meant to be a baseline solution for simulation effort required for Physics Performance Report (due next year)
- Must be ready early next year for ADC4

Boundary conditions

- Must be able to handle
  - Simulation and simulated events
    - Event size 2GB
    - possibly split into several physical files
    - 24h (800 MHz CPU) needed to create digits
    - 4h reconstruction
    - 20000 background events required
  - Huge number of raw data events data
    - size 40MB (Pb+Pb), 1MB (p+p)
    - 2PB/year
    - 1.2 GB/s
    - 10^9 files/year (x n, n>2)

@GRID

Use Cases

- Data Challenges
  - Monitoring and event sampling from remote sites
- Simulation
  - Centrally managed (configuration, monitoring) production of background events
  - Distributed processing and event storage
- Event mixing
  - Once background events exist, subsequent requests for event mixing must be routed to the location which holds required input data data
  - Not necessarily centrally managed

Alice Computing Model

- Distributed computing model along GRID lines
  - Currently 13 listed, possibly 20+ remote sites
- Two prong approach
  - AliGrid: try to use GRID tools as they are
  - AliEn: a lightweight, simplified but functionally equivalent alternative

What is required?

- File catalogue – global file system on top of relational database
  - Scalable & distributed
- Secure authentication service independent of underlying database
  - Unix like groups and permissions
- Central task queue
  - Commands (job specification) as well as job input and output stored in the catalogue
  - "Pull" rather than "push"architecture
- Flexible and extensible services
  - Accounting, monitoring…

@GRID

Implementation

- Perl5
  - Already existing pieces of code (NA49)
  - Good interface to different databases
  - Easy Web integration
- SOAP
  - Simple Object Access protocols (also known as Service Oriented Access Protocol)
  - Good Perl implementation (SOAP::Lite) on client and server side
  - Possibility to provide client access from many different platforms and languages (Java,C,C++…)

@GRID

Why Perl?

- Open Source
- Supported under Unix, Macintosh, Microsoft
- There are over 400 third party modules available from the Comprehensive Perl Archive Network (CPAN).
- Works with third-party databases like Oracle, Sybase, Postgres, and many others through the abstract database interface called DBI.
- Supports both procedural and object-oriented programming.
- Can interface with external C/C++ libraries be embedded into other systems

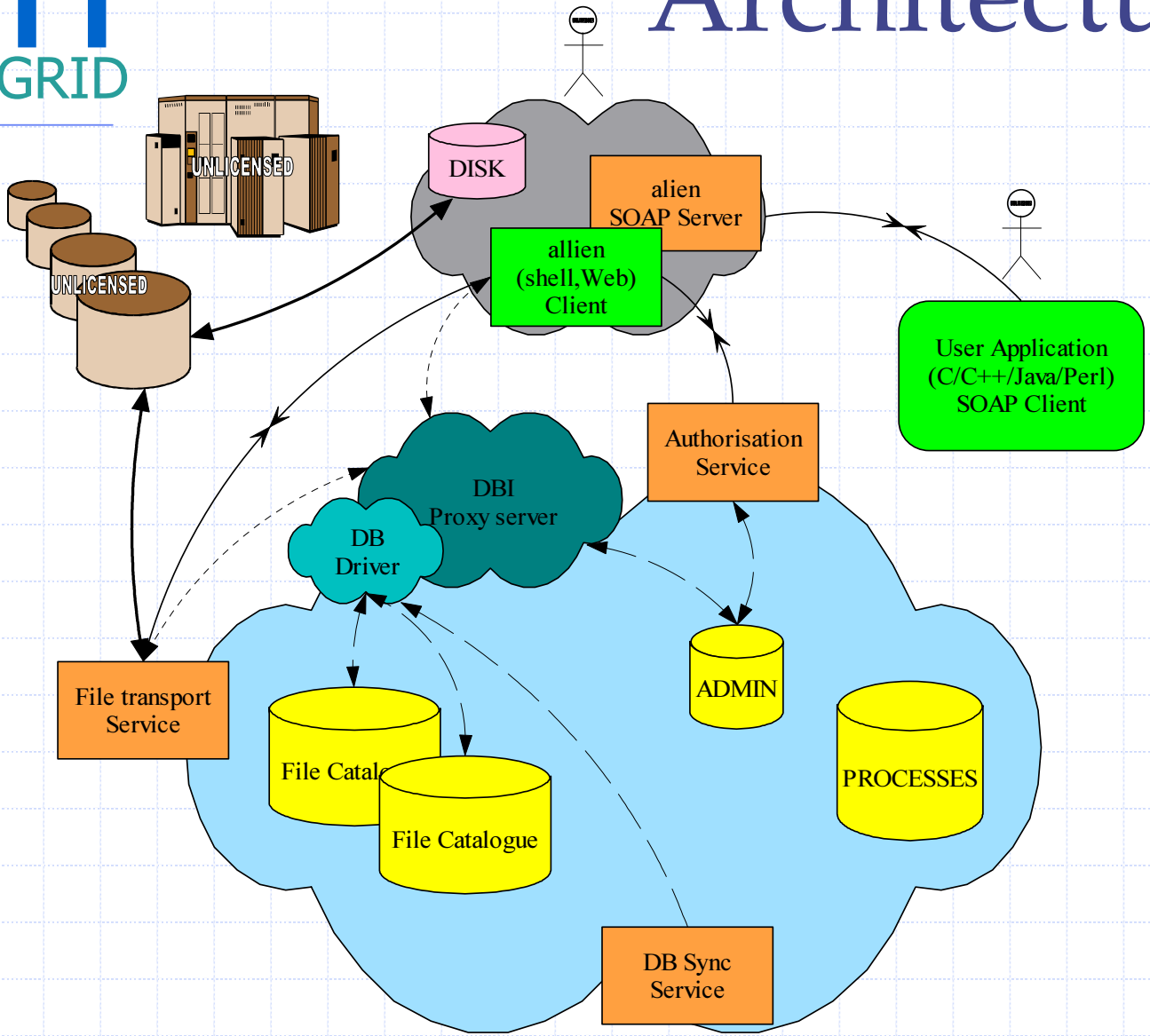
@GRID

SOAP Usage

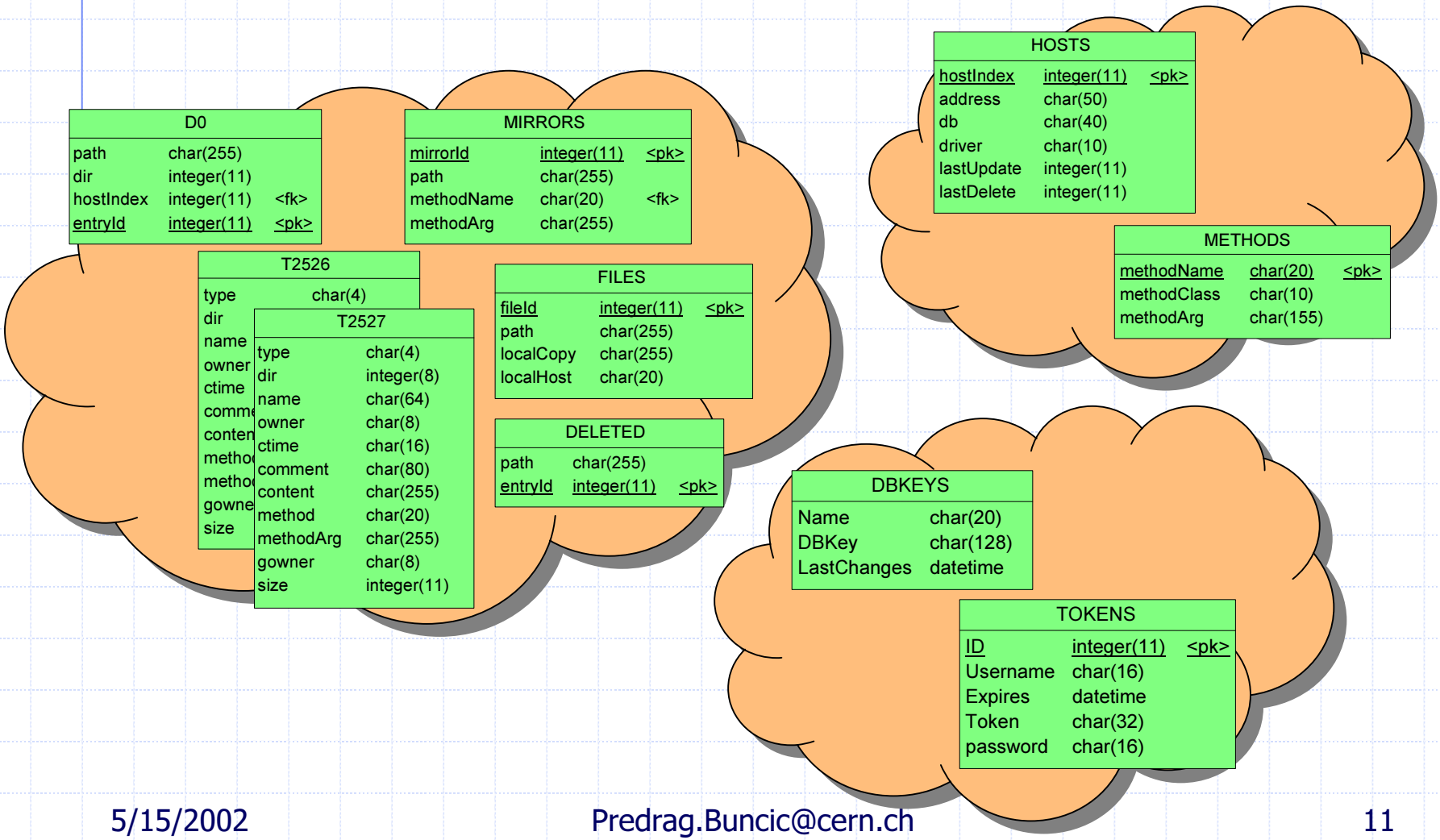
```
use DBBrowser::ObjectServer;

use SOAP::Transport::TCP;

my $daemon = SOAP::Transport::TCP::Server
    -> new( LocalAddr => "$host", LocalPort => $port, Listen => 5, Reuse => 1 )
        -> dispatch_to( "DBBrowser::ObjectServer" )
        -> handle
;
```

```
use SOAP::Lite;

my $soap=SOAP::Lite
    -> uri( 'DBBrowser/ObjectServer' )
    -> proxy("tcp:$host:$port" );

my $interface=$soap
       ->call('new' =>{'user'=>'alice'}), 'password', 'xxxxx'})->result;

my $dir= $soap->execute( $interface, 'cd')->result;
$dir= $soap->execute($interface, 'mkdir', 'otro')->result;
$dir= $soap->execute($interface, 'ls', '-l')->result;
$dir= $soap->execute($interface, 'tree')->result;
```

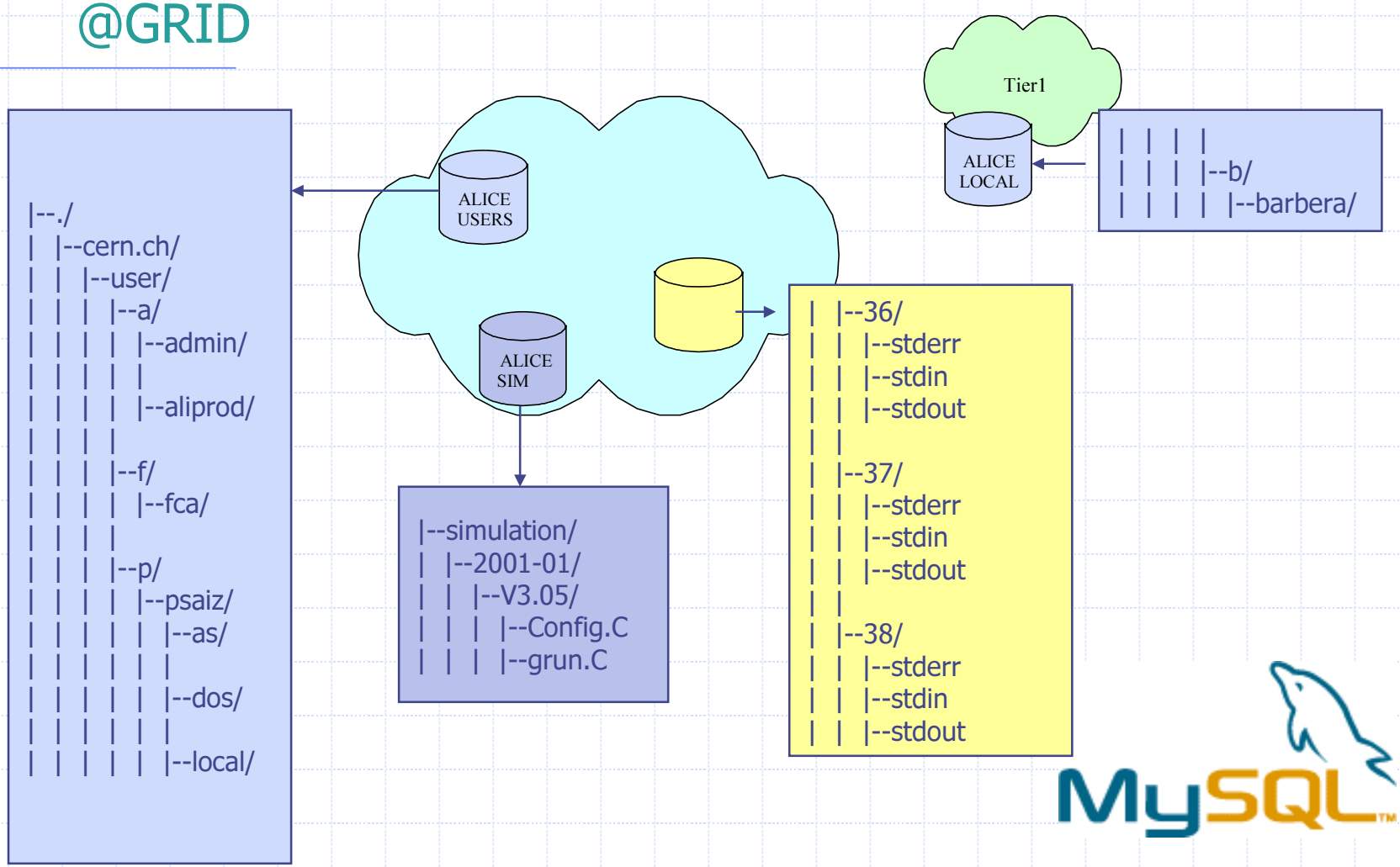
Architecture



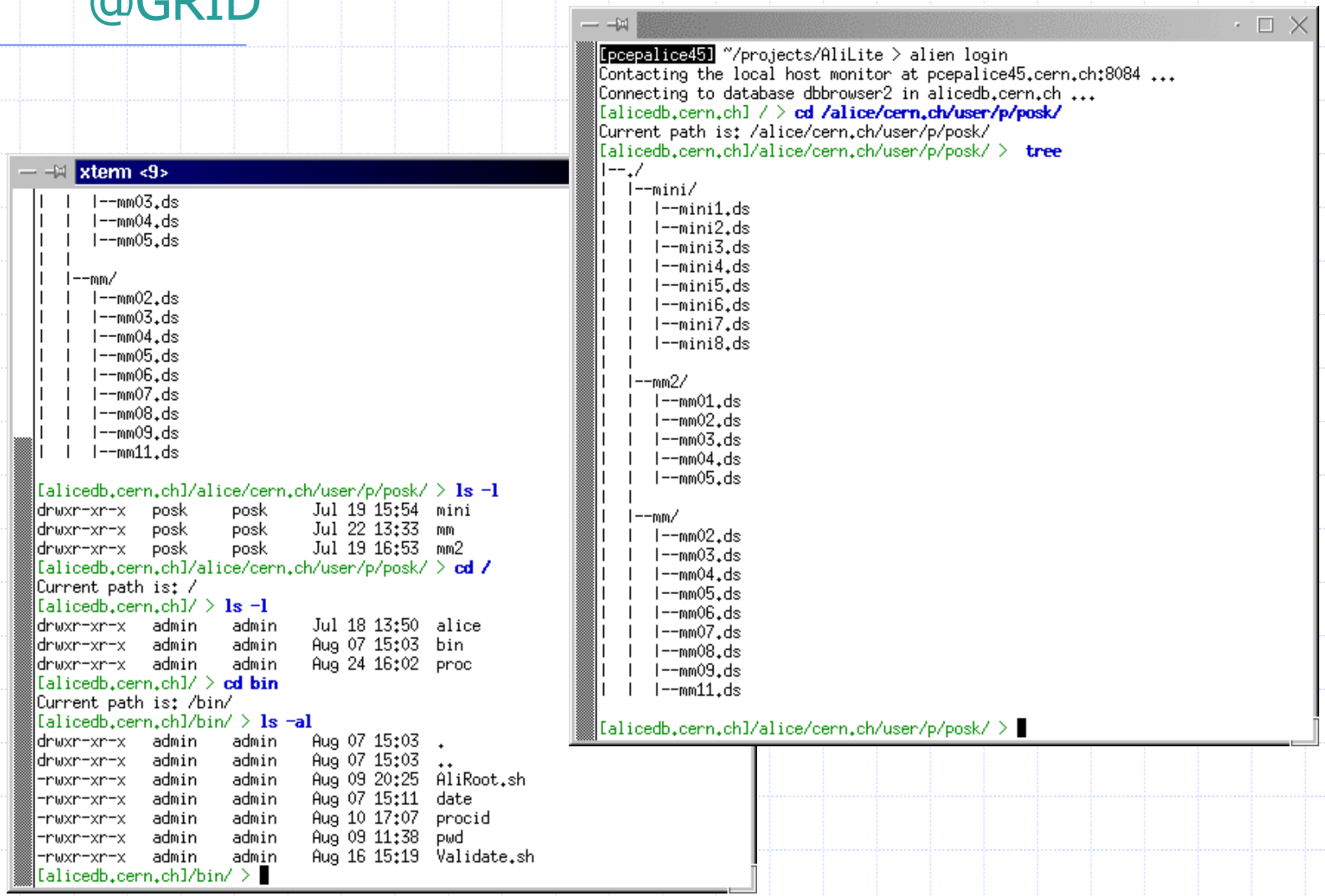
File catalogue



Distributed database



Cmd Line Interface



@GRID

LFN

- Logical file names
  - /alice/user/b/buncic/galice.root (command line interface)
  - Lfn://alien.cern.ch/alice/user/b/buncic/galice.root (via API)
  - Lfn://alien.cern.ch/alice/simulation/2001%/V3.05/%/galice.root?npart>1000#mytag (using tag table associated with directory table)

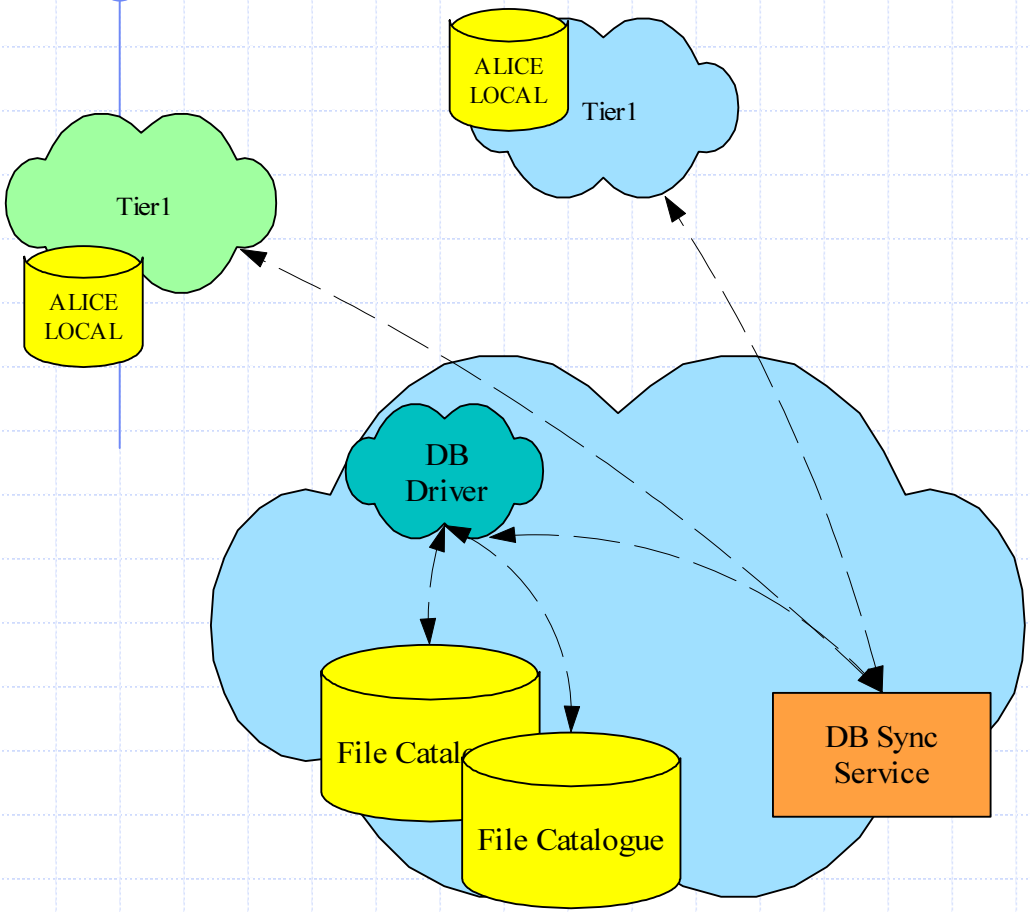
TFN,PFN

File Transport Methods

METHODNAME	METHODCLASS
RFIO	RFIO
castor	Castor
TAPE	TAPE
FTP	FTP
RCP	RCP
FTP2	FTP
ALIDB	DB
File	File

- Transport and physical file names are not clearly separated
- The entry in FILES table contains a reference to METHOD in URL like notation
- After URL schema, follows the hash list of arguments required to execute selected method
- This will be revised and all files will be referenced via standard URI::URL interface

Synchronization



- Some tables must be synchronized between database servers (D0)
- D0 table provides an instant lookup between file path and corresponding directory (table)
- This is a task of DB Sync Service

@GRID

Mirroring

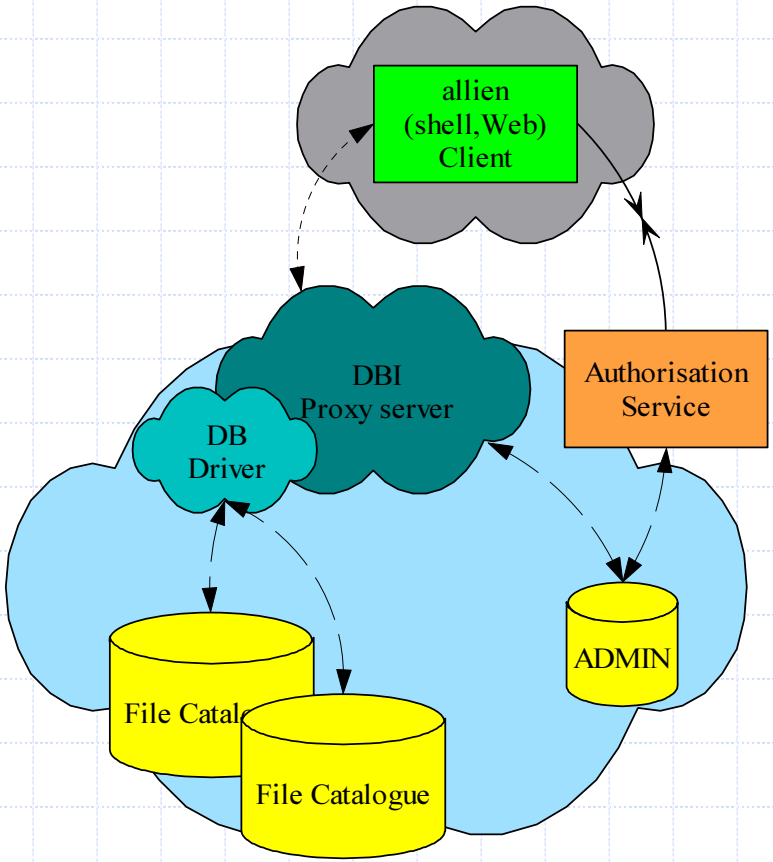
- User (or another service) can request LFN to be mirrored on its site
- In this case, system updates MIRRORS table on a local site
- This table is searched every time user request LFN for which D0 table indicates that original file is on remote site

@GRID

Cache

- Every time when user requests LFN, the client tries to copy corresponding file to a local file system cache and registers copy in local FILES table
- On subsequent request for the same LFN, the cached copy is returned (if exists), otherwise, client tries again to reduce TFN to PFN
- If it fails, it asks File Transport Service to reduce LFN to URL it can handle
- File Transport Service from one site can contact another site and forward the request

Authentication



- User (via SOAP) client connects to Authentication Service using AFS password
- Password is verified against local PAM
- If user does not exist, new account entry in ADMIN database is created
- New token and new user password are created
- Password is stored in local secure database table
- Token is sent back to client and encrypted with secret key
- On subsequent request, user supplies token which is verified by the proxy server
- User never gets real password to access database
- The password is changed after 24h and that invalidates user token as well

Bookkeeping

- As complicate as it might look, file catalogue still does not know anything about physics
- Adding an additional information to quantify event (file) properties in terms of physics observables transforms file catalogue into bookkeeping database
- In AliEn environment this can be achieved by attaching an arbitrary table(s) to corresponding directory table
- Example:
  - Lfn://alien.cern.ch/alice/simulation/2001%/V3.05/%/galice.root?npart>1000#mytag
  - In this example, the search will first select all tables on the basis of the file name selection and then locate all tables that correspond to "mytag" definition, apply selection and finally return only the list of files for which the attribute search has been successful

Tags

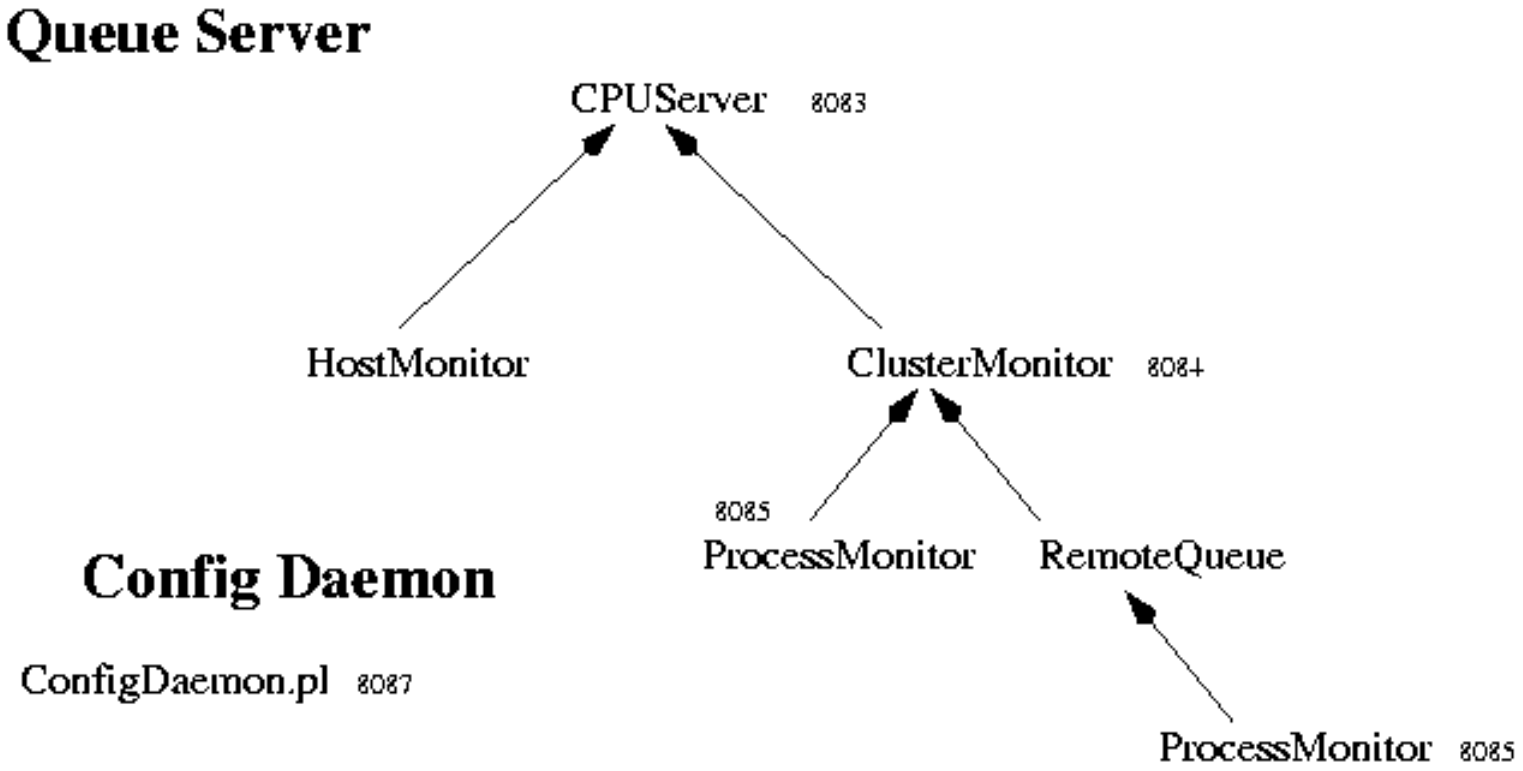
```
--./
|  |--r3418_01-01.ds
|  |--r3418_02-02.ds
|  |--r3418_03-03.ds
|  |--r3418_04-04.ds
|  |--r3418_05-05.ds
|  |--r3418_06-06.ds
|  |--r3418_07-07.ds
|  |--r3418_08-08.ds
|  |--r3418_09-09.ds
|  |--r3418_10-10.ds
|  |--r3418_11-11.ds
|  |--r3418_12-12.ds
|  |--r3418_13-13.ds
|  |--r3418_14-14.ds
|  |--r3418_15-15.ds
|  |--r3418_16-16.ds
|  |--r3418_17-17.ds
```

- Tags are stored in catalogue as well(in /alice/tags directory)
- They simply contain SQL definition of a table that will be crated when tag is associated with directory
- It is user's responsibility to populate tag table
- Only owner of directory can add new tag to it
- This scheme is completely open and flexible

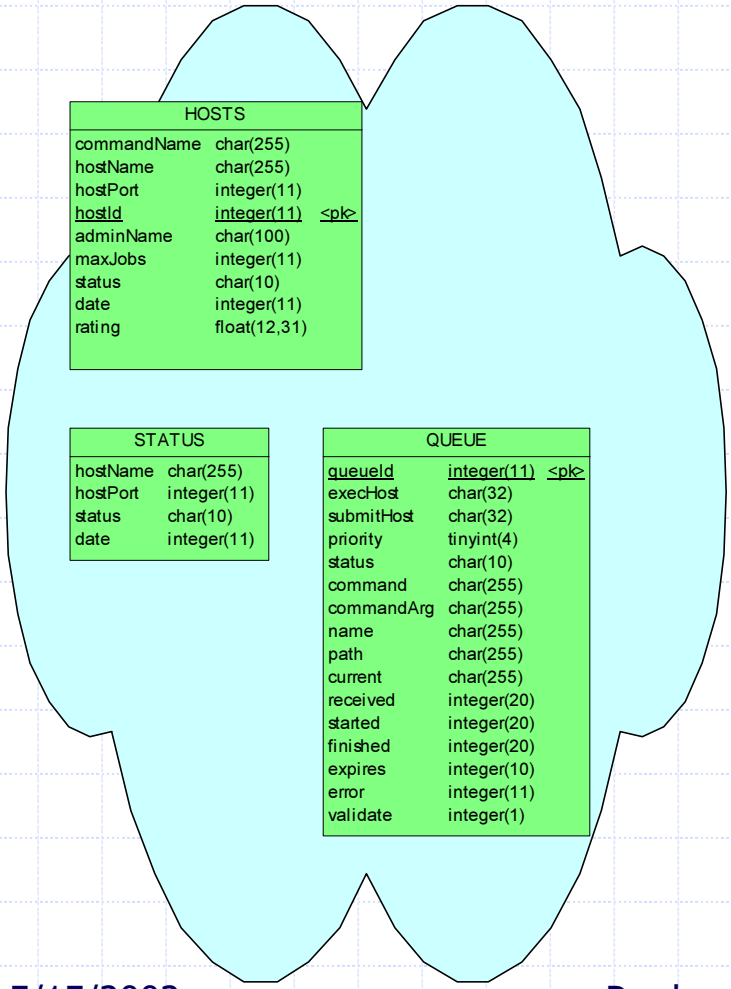
Calibration

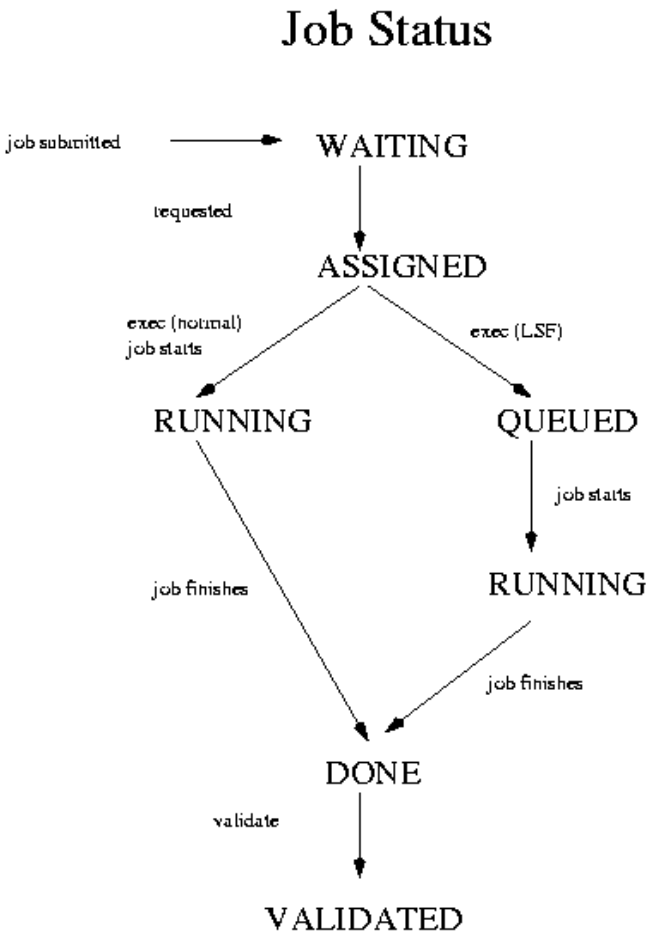
- This can be handled by defining a special kind of tag containing the key pairs like
  - First_run, Last_run
  - Start_time, Stop_time
- In addition, this tag table could have any other column (like version)
- The application would simply request calibraton object which corresponds to (run,time,version) and catalogue would return a file which contains requestd information

Task Queue - Services



Task queue


HOSTS
commandName char(255)
hostName char(255)
hostPort integer(11)
hostId integer(11) <pk>
adminName char(100)
maxJobs integer(11)
status char(10)
date integer(11)
rating float(12,31)
STATUS
hostName char(255)
hostPort integer(11)
status char(10)
date integer(11)
QUEUE
queueId integer(11) <pk>
execHost char(32)
submitHost char(32)
priority tinyint(4)
status char(10)
command char(255)
commandArg char(255)
name char(255)
path char(255)
current char(255)
received integer(20)
started integer(20)
finished integer(20)
expires integer(10)
error integer(11)
validate integer(1)


Job Status
job submitted
WAITING
requested
ASSIGNED
exec (normal)
job starts
exec (LSF)
RUNNING
QUEUED
job starts
job finishes
RUNNING
job finishes
DONE
validate
VALIDATED

@GRID

Sending a job…

Production

Edit

ROUND	TAG	COMMENT	STATUS	PATH
2001-01	V3.05	PPR simulation	TESTING	/alice/simulation/2001-01/V3.05

Runs

Edit

Action	Run	Todo	Done	Failed	STARTED	Config	Macro
Resume	2	10	0	0	Sat Aug 25 10:39	Config.C	grun.C

None
Submit
Pause
Resume
Stop

Submit Reset

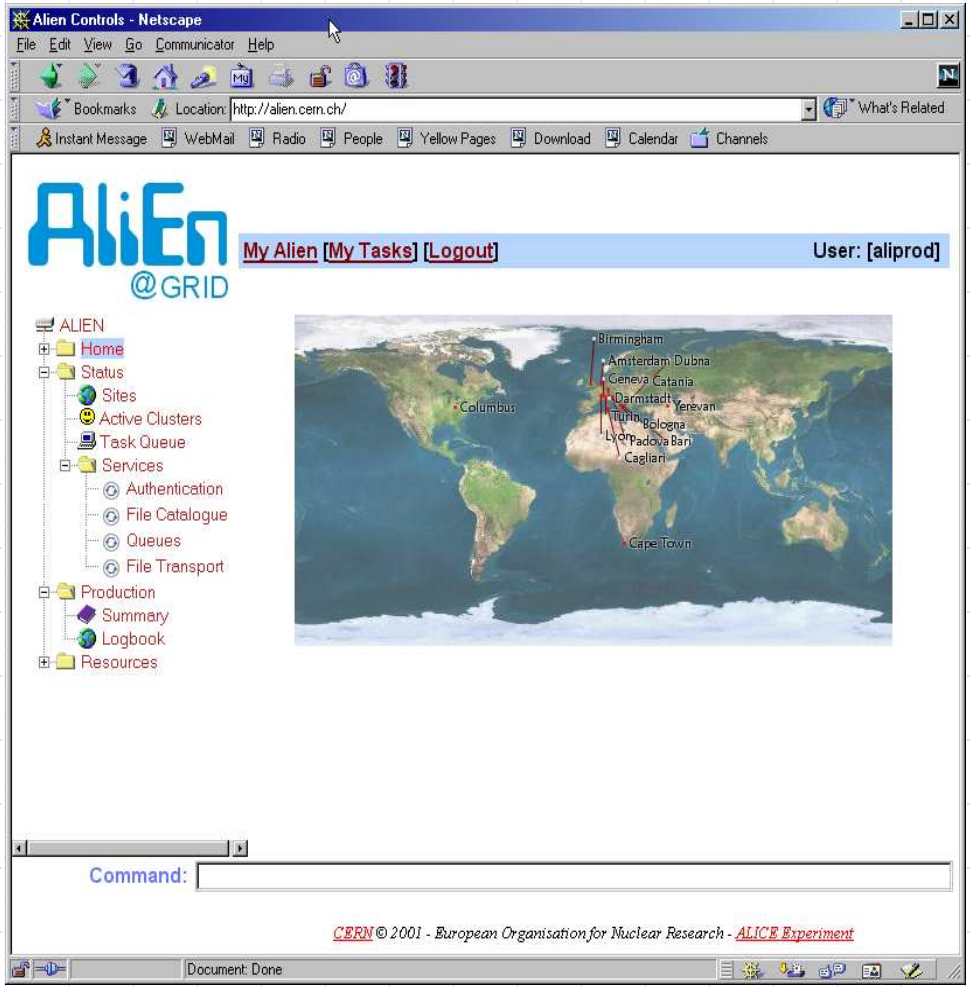
Task Queue

[ALL][EXPIRED][WAITING][DONE][KILLED][ERROR]

Task Queue

ID	USER	EXEC	STATUS	COMMAND	START	TIME
30	psaiz	pcepaip15.cern.ch	ERROR	AliRoot.sh	Fri Sep 14 10:21	447:38:42
31	psaiz		KILLED	AliRoot.sh		00:00:00
32	psaiz		KILLED	AliRoot.sh		00:00:00
33	psaiz	pcepaip15.cern.ch	ERROR	AliRoot.sh	Fri Sep 14 09:57	448:02:59
34	psaiz	pcepaip15.cern.ch	ERROR	AliRoot.sh	Fri Sep 14 10:00	447:59:12
35	psaiz	pcepaip15.cern.ch	KILLED	AliRoot.sh	Fri Sep 14 10:05	447:54:06
36	psaiz	pcepaip15.cern.ch	DONE	AliRoot.sh	Fri Sep 14 10:23	00:00:19
37	psaiz	pcepaip15.cern.ch	DONE	AliRoot.sh	Fri Sep 14 10:27	00:00:49
38	psaiz	pcepaip15.cern.ch	DONE	AliRoot.sh	Fri Sep 14 11:22	00:00:15
39	psaiz	pcepaip15.cern.ch	KILLED	AliRoot.sh	Fri Sep 14 10:18	447:41:16
40	psaiz	pcepaip15.cern.ch	DONE	AliRoot.sh	Fri Sep 14 11:25	00:00:18

The Web Interface



AliEn & GRID

- AliEn is not supposed to be replacement for GRID on long term
- It is our working environment that can gradually incorporate GRID elements as they become available
- Working on it and using it, we can gain much better ideas on what is really needed from GRID
- The bottom line: take it as elaborate case of user requirements document from user who prefers writing code to writing documentation

@GRID

Todo

Info & Installation

- Web site: http://alien.cern.ch
- Installation instructions (including all perl module dependencies): See Roberto's talk