LHCb distributed analysis and ARDA

A.Tsaregorodtsev,
CPPM, Marseille

ARDA Workshop, 21 January 2004, CERN

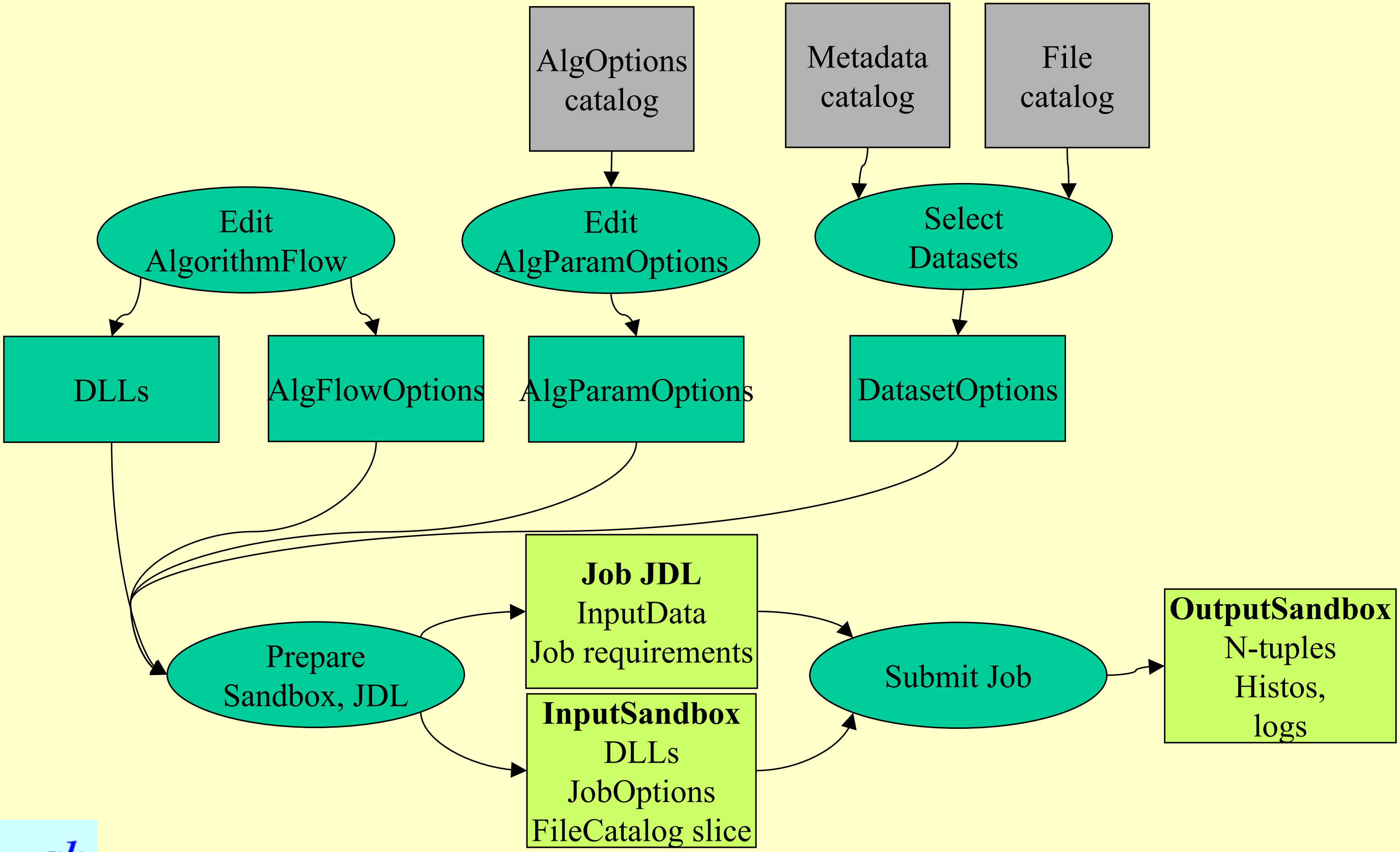
Outline

- LHCb distributed analysis tasks
- Distributed analysis tools
- Analysis of the DC2004 data
- What we expect from ARDA
- Conclusions

Analysis tasks

- LHCb distributed analysis tasks have (almost) no specific features compared to other LHC experiments:
  - As formulated in the HEPCAL II document;
- In the following the batch analysis tasks are discussed mostly; the interactive analysis is currently limited to PAW/ROOT sessions.

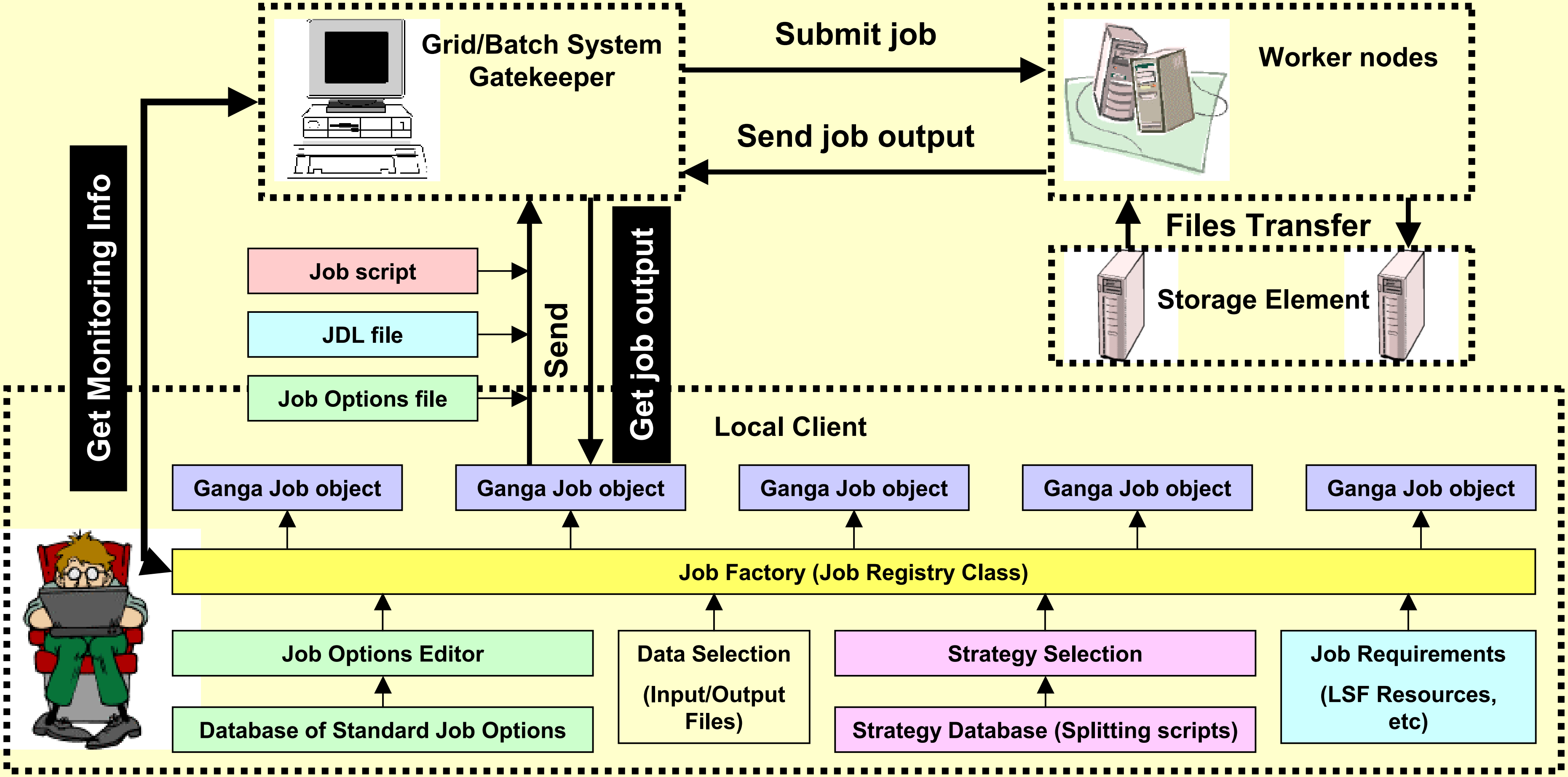
Analysis task flow



Analysis tools

- GANGA User interface;
- Metadata Catalog;
- File Catalog;
- Data Management;
- Workload management;
- Various information services:
  - Configuration service;
  - Job and System monitoring;
  - Software package manager;
  - etc

GANGA User interface



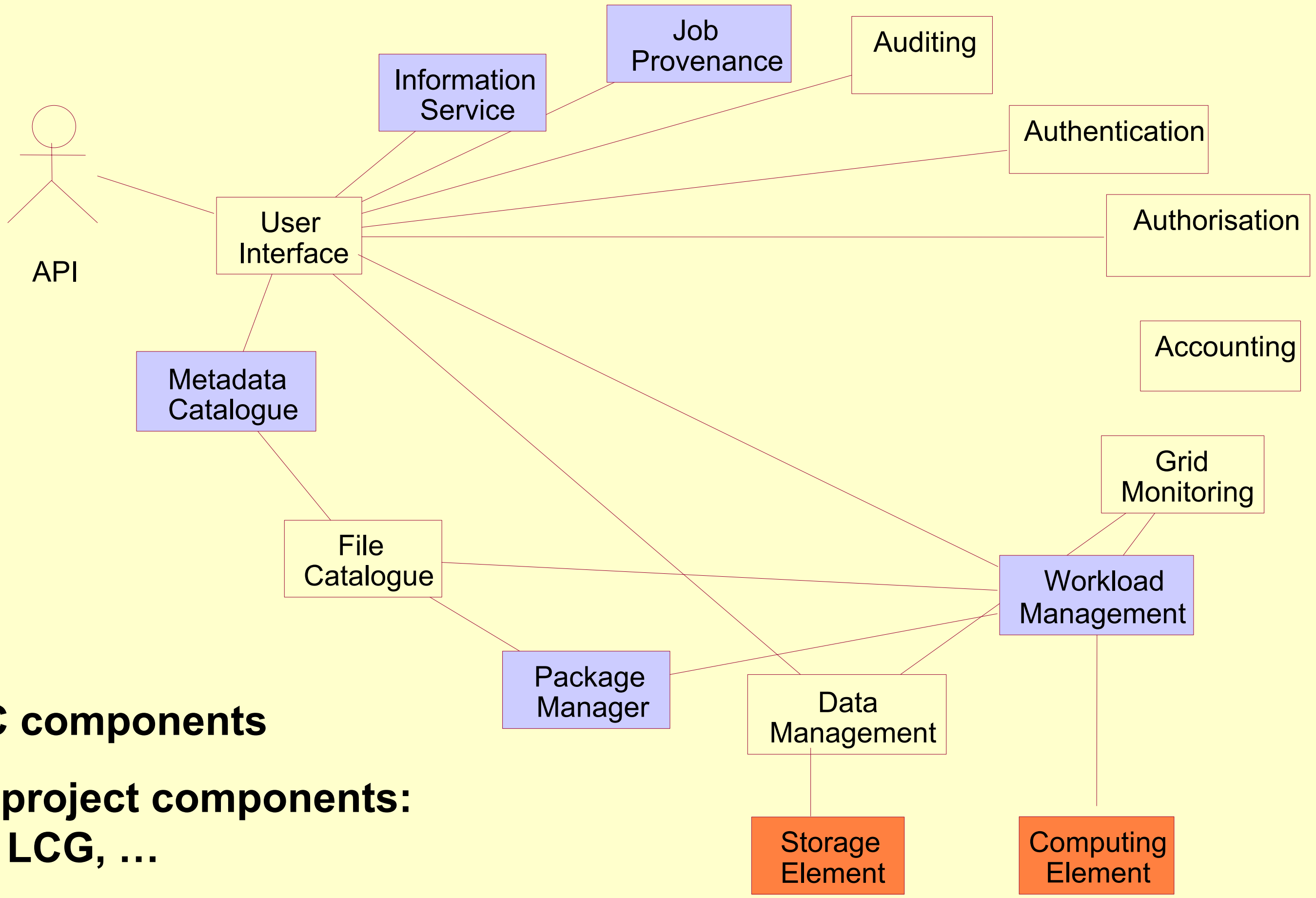
GANGA User Interface

- GANGA will allow user to perform standard analysis tasks:
  - Selecting data making queries;
  - Configuring jobs, defining the job splitting/merging strategy;
  - Submitting jobs to the chosen grid resources;
  - Following the job progress;
  - Retrieving the job output;
  - Job bookkeeping.

Services based architecture

- The UI provides user with an access to various grid services:
  - Catalogs;
  - Job execution;
  - Information;
  - Monitoring;
- These services are either developed in ***DIRAC*** – the LHCb distributed production system
- Or imported into ***DIRAC*** from other projects
- Or provided by LCG for tasks executed on the LCG2 platform.

DIRAC services architecture



🟪 **DIRAC components**

⬜ **Other project components: AliEn, LCG, …**

🟧 **Resources: LCG, LHCb production sites**

Bookkeeping database

- Corresponds to the ARDA Job Provenance DB and MetaData Catalog;
- ORACLE with XML-RPC service interface;
- Very flexible basic structure with regularly built views to enhance standard queries efficiency.

File Catalog

- Used to be part of the Bookkeeping database, we never had the intention to develop our own FC;
- Starting from DC2004 we are going to use AliEn File Catalog wrapped as web service with XML-RPC (or SOAP) interface;
  - Use ARDA File Catalog as it will become available;
  - Making POOL interfaced to the ARDA Catalog is necessary.
  - Using File Catalog slices as POOL XML catalogs to allow execution of applications without access to remote catalog.
- For data produced in LCG we will (try to) use RLS:
  - Synchronization of the 2 catalogs is an issue.
  - Not all the MSS's are ready to be used now !

Data management tools

- Using bbftpd server as a SE:
  - Too limited the functionality;
  - Using passwords or globus certificates for authentication;
- Considering other Data Management tools candidates:
  - AliEn Data Management;
  - LCG replica manager.

Information services

- Configuration service:
  - MySQL with an XML-RPC interface to provide DIRAC configuration parameters to other services and agents;
- PackageManager service:
  - MySQL with an XML-RPC interface to provide information on software packages availability, dependencies and status;
- Job Monitoring and Accounting;
  - Provides information via XML-RPC interface to job monitoring and accounting report generation applications.
- System Monitoring service:
  - Monitoring the availability of the services and agents;
  - Looking at Instant Messaging (Jabber) mechanism of services presence detection.

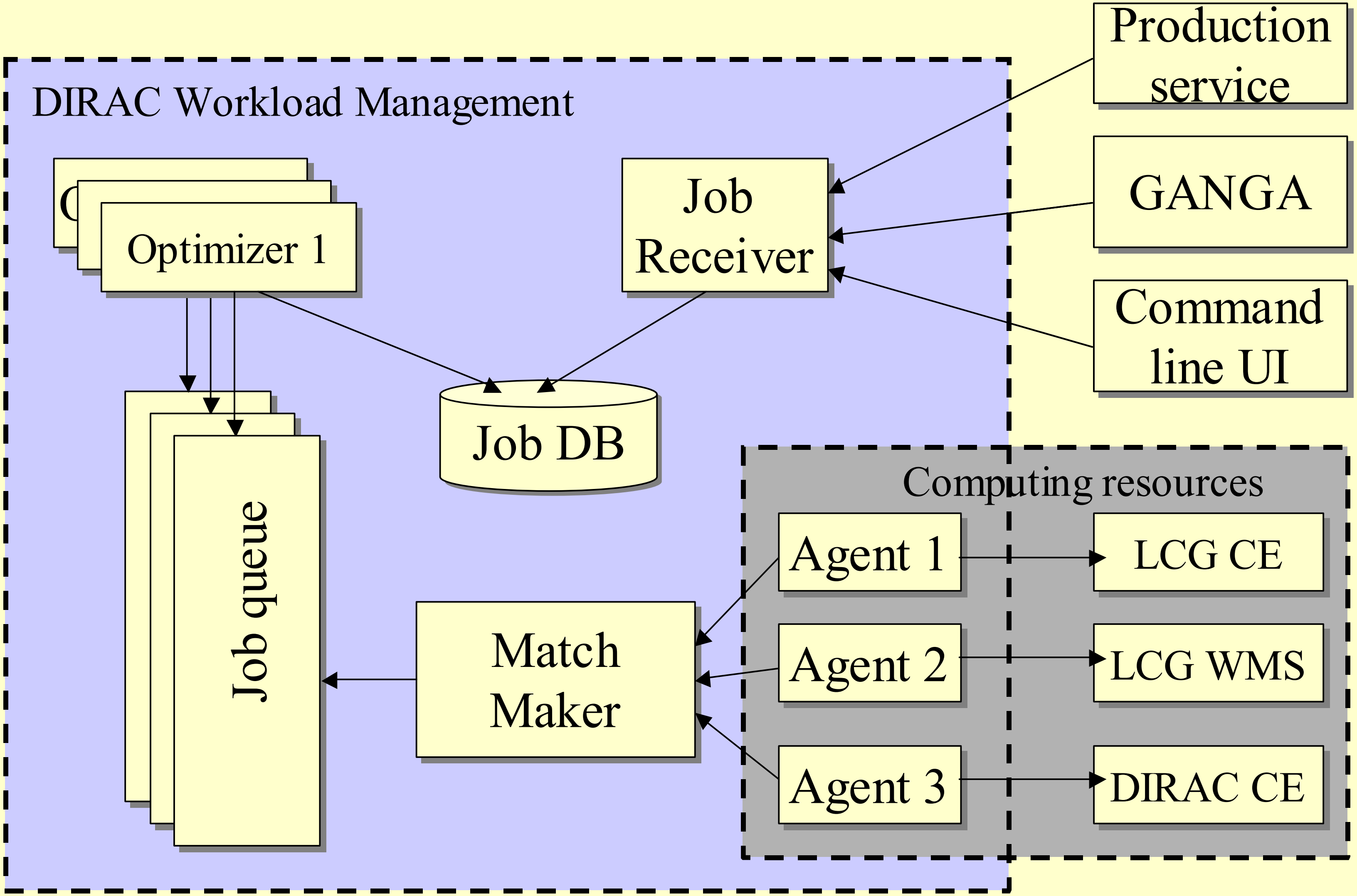
Software installation

- In DIRAC we install the production software on the fly by the running job:
  - Proved to be very efficient and useful;
  - Only the first job of the kind installs the new version on a DIRAC site;
- This worked as well on the EDG testbed:
  - Each job installing the software for itself;
- On the LCG we will use special jobs and info tags to install the production software on the LCG sites:
  - Following the procedure proposed by the LCG team:
    - Software distribution uploaded to one of the SE on the grid;
    - Installation/verification jobs by a specially defined user account
- User specific analysis algorithms are shipped as DLL's in the InputSandbox of a job.

DIRAC use of computing resources

- DIRAC design goals to facilitate operation in various environments:
  - Scheduling jobs to "any grid" computing resources:
    - "native" sites, running DIRAC Agents;
    - EDG/LCG grid as a whole, passing through the RB;
    - EDG/LCG CE's and SE's as DIRAC resources.
  - Using "any grid" storage resources:
    - Data Management component to be able to replicate data between LCG and DIRAC storage elements;

DIRAC WMS architecture



DIRAC WMS components

- Job Receiver:
  - Connects to the Job Data Base
  - Parses the JDL
  - Inserts the JDL into the appropriate tables
  - Notifies Optimizers that a new job has arrived via the Jabber Instant Messaging system
- Optimizers
  - Extracts new job from DB
  - Inserts it into Job Queue with a particular rank
  - Sort Jobs
- Matcher
  - Does match making between Job Jdl and resource jdl
- Agent
  - Asks job to Matcher with its specific resource JDL
  - Send job to CE

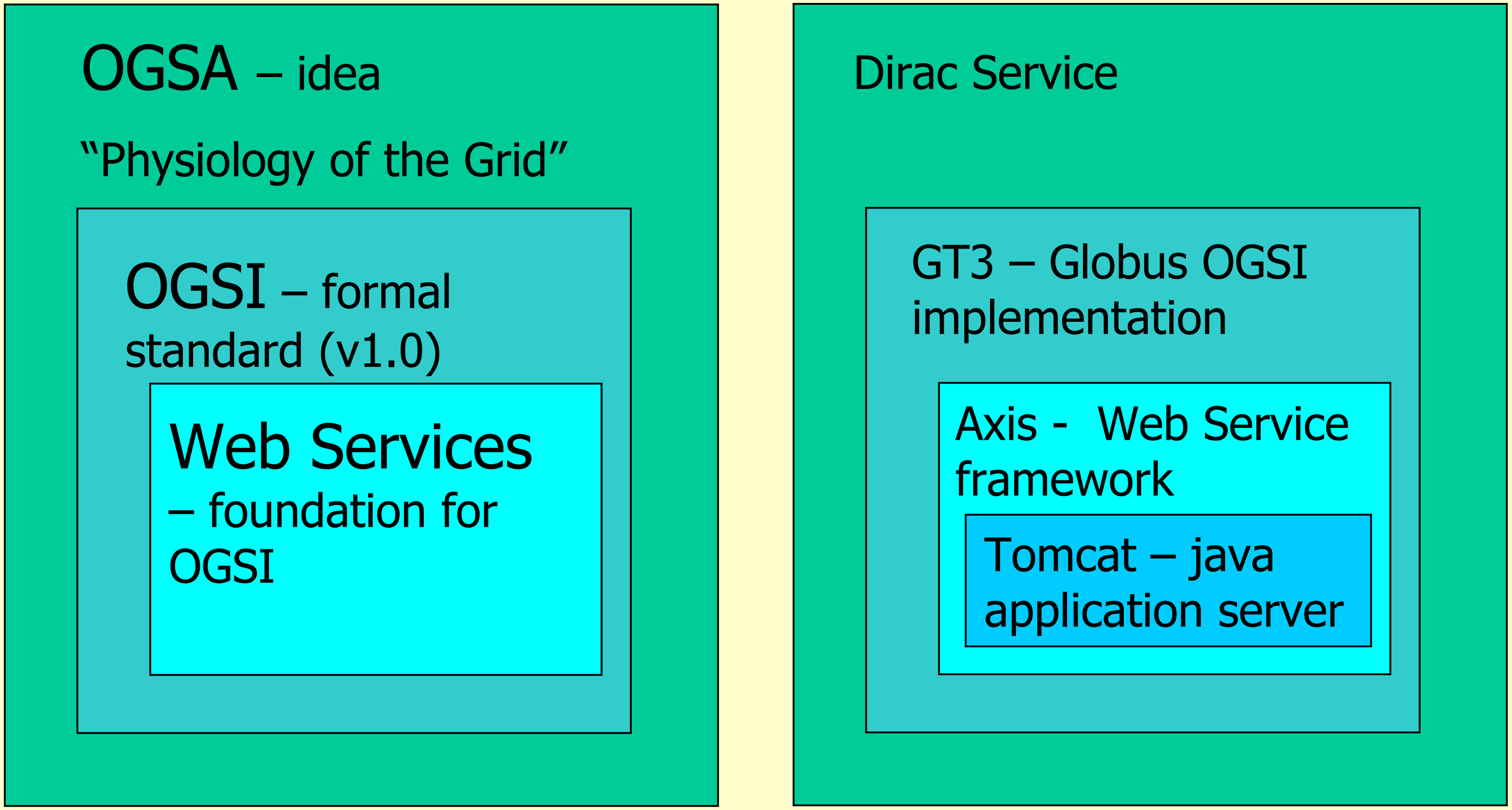
DIRAC WMS technologies

- JDL Job description ;
- Python as development language ;
  - Sharing developments with GANGA, maybe reuse some modules;
- MySQL for Job DB and Job Queues ;
- Condor Classad library for matchmaking (wrapped in python with SWIG) ;
- Internal interfaces by instant messaging technologies ;
  - Python/Java Jabber ;
  - Advantages : Asynchronous message, distributed service, XML based, scalable, robustness, and so on…
- Various batch system back-ends (CE's):
  - LSF, PBS, Condor, EDG/LCG, fork.
- Running jobs communicating with various services:
  - WN connectivity is mandatory ! (at least Outbound).

OGSI compliant prototype

- Development the DIRAC WMS interface as an OGSI compliant service:
  - Using GT3 toolkit
- First experience:
  - The GT3 based services are difficult to develop, deploy and configure;
  - The service efficiency is low:
    - 1 minute per job submission instead of 1 second with the “standard” DIRAC service;
- Looking for other OGSI implementations:
  - Lighter and easier to use, may be with somewhat limited fuctionality;
  - E.g. PyGridWare package.

OGSI philosophy vs GT3 implementation



Concepts

OGSA – idea

“Physiology of the Grid”

OGSI – formal standard (v1.0)

Web Services – foundation for OGSI

Implementations

Dirac Service

GT3 – Globus OGSI implementation

Axis - Web Service framework

Tomcat – java application server

Distributed analysis on the LCG

- The analysis will use LCG as it is:
  - Task preparation in GANGA;
  - Submission to the RB of the LCG;
  - Using LCG SE/RLS/RM for data manipulation;
- Making data produced outside LCG available for the LCG jobs is an issue:
  - Copying to LCG SE's or just registering in the RLS ?
- During the EDG tests, we studied the mode of operation when jobs were submitted directly to CE's bypassing the RB
  - Hope this will be not necessary with more stable RB of the LCG;
  - Keeping this possibility in our toolkit.

Batch versus Interactive analysis

- All said so far concerns Batch Analysis:
  - Submitting jobs to batch system;
  - No direct interaction with the jobs.
- We are interested though in the possibility to run interactive jobs on the grid:
  - Many issues to be solved:
    - Job efficient parallelization;
    - Resources reservation;
    - WN connectivity problems;
    - Etc, etc
  - The PROOF system is very promissing:
    - Allthough not applicable to LHCb straight away

What we expect from ARDA

- Lacking components, most notably:
  - File Catalog;
  - Data management;
  - Grid monitoring;
  - Authentication/authorization;
- The architecture where we can choose the most suitable and best performing component implementations.

What we expect from ARDA

- More efficient development process:
  - Rapid development cycles;
  - Keeping the functional core and adding functionality incrementally;
  - Emphasis on intensive testing while the development.
- Concurrent development of components to try out different ideas and to enhance the quality by competition.

How we can contribute to ARDA

- Participate to the definition of the services interfaces, testing and feedback;
- Prototyping ARDA components using OGSI compliant implementations;
- Developing the DIRAC WMS into an ARDA compliant service

Conclusions

- The first tests of the distributed analysis will be done during the DC2004 (May) using LHCb developed and LCG tools;
- We see the further evolution of the LHCb distributed analysis tools within the context of the ARDA architecture and the proposed development process.