Future Developments in EDG

The European DataGrid Project Team

http://www.eu-datagrid.org

Overview

- Where is the DataGRID project going?
  - Future developments in the EDG middleware
    - What you can expect to see in future releases of the software
  - WebServices and Open Grid Services Architecture
    - Where Grid computing is heading in the coming years

Workload Management (WMS)

- Architecture has been revised
  - Increase reliability and flexibility
  - Simplification (e.g. minimize duplication of persistent information)
  - Easier to plug-in new components that implement new functionality
  - Address some of the shortcomings that emerged in the early releases
  - Better integration with other Grid services (e.g. data management optimization facilities)
  - Favour interoperability with other Grid frameworks
- Advanced Functionality
- A coordination between EDG and PPDG (US project) has been established to define a common approach

WMS: New Functionality

- **Interactive Jobs**
  - jobs with continuous feedback of standard streams (stdin, stdout, and stderr) on the UI (submitting) machine
- **Job Check-pointing**
  - save the job state via "trivial" check-pointing API, so execution can be suspended and resumed
- **Job Partitioning**
  - decompose the job into smaller sub-jobs (executed in parallel) to reduce the elapsed processing time and optimize the usage of Grid resources
- **Job Dependencies**
  - program Y cannot start before program X has successfully finished
  - based on Condor DAGMan http://www.cs.wisc.edu/condor/dagman
- **C++ and Java API, and GUI**
  - API provides similar functions to command line
  - GUI does guided-creation of JDL & monitoring the status of jobs over the whole life cycle

New features: Deployment of Accounting infrastructure over Testbed

The DGAS (Data Grid Accounting System) will be based upon the following assumptions:

A computational economy model will implement an economic-brokering mechanism where Grid-resources are chosen by the Broker according to a cost assigned to them.

Users pay in order to execute their jobs on the resources, and the owner of the resources earns credits by executing the user jobs.

There are three reasons for this:

1. To have a nearly stable equilibrium able to satisfy the needs of both resource providers and consumers
2. To credit job resources usage to the resource owner(s) after execution
3. To avoid abuse of grid resources

New features: Advance reservation API and Co-allocation API

Advance reservation of resources to realize end-to-end quality of service (QoS) and reduce competition for resources.

The approach is based on concepts discussed in the Global Grid Forum.

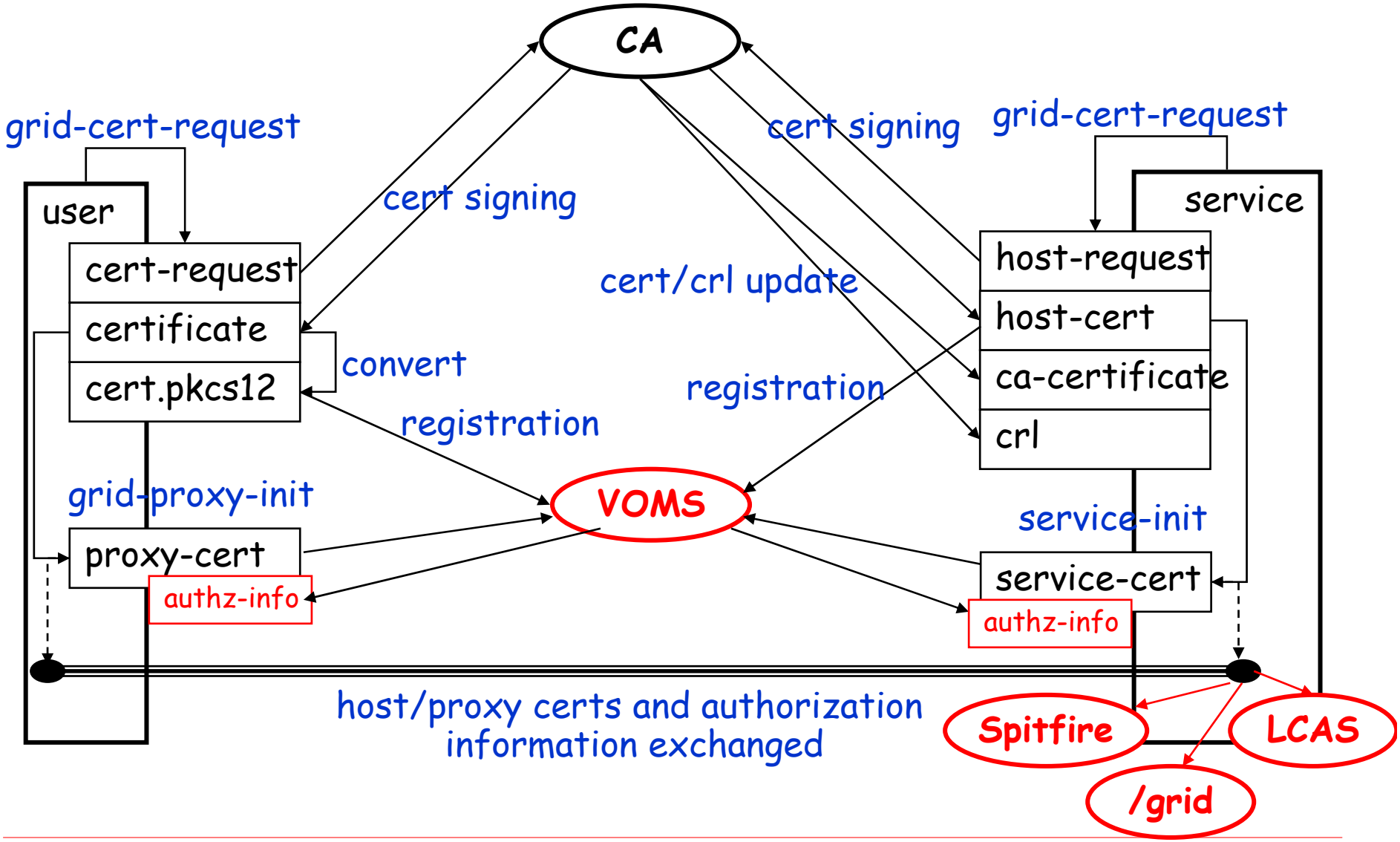
A reservation is a promise from the system that an application will receive a certain level of service from a resource (e.g. a reservation may promise a given percentage of a CPU).

Co-allocation allows the concurrent allocation of multiple resources.

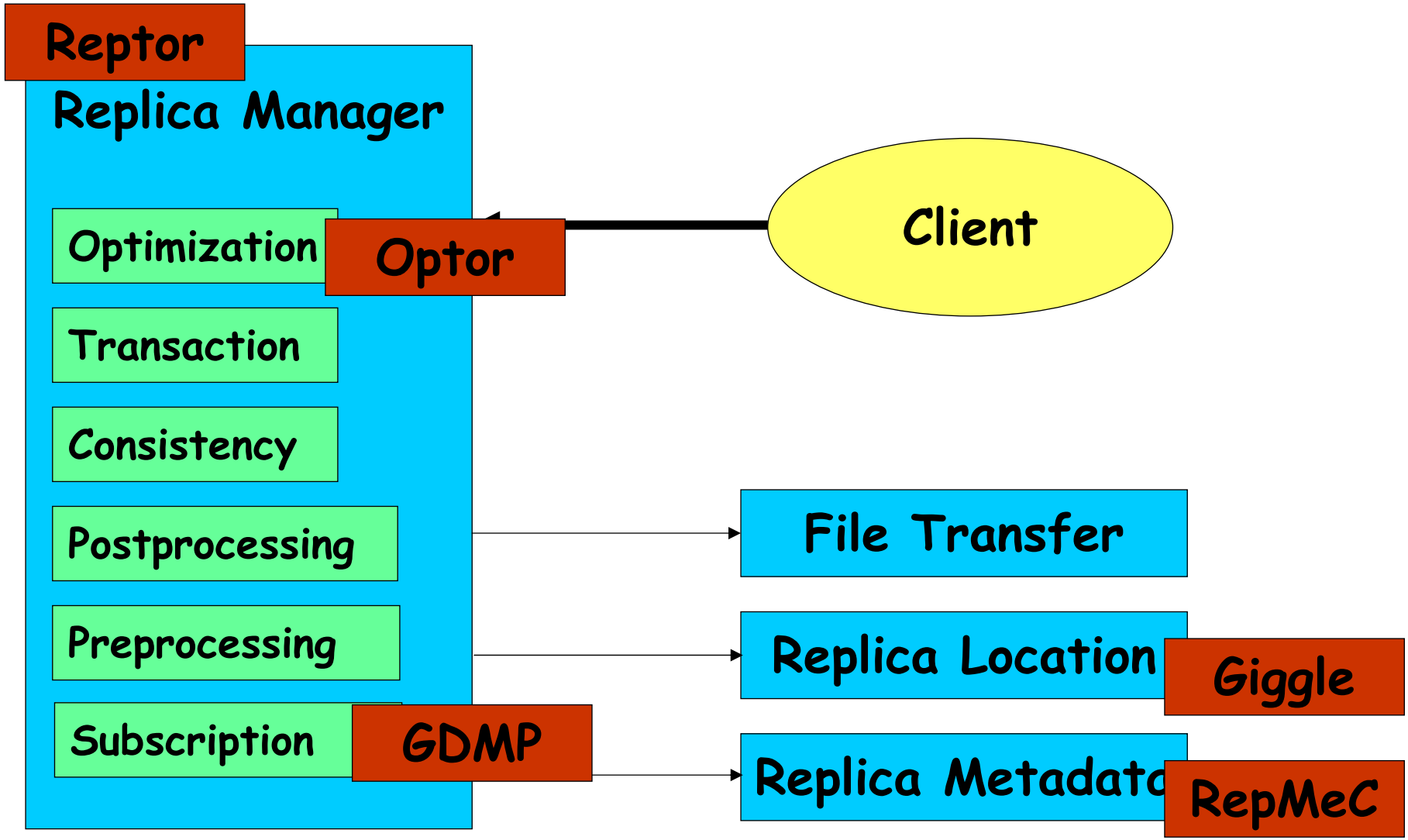
These resources can be homogeneous or heterogeneous.

Authorization Improvements

plans

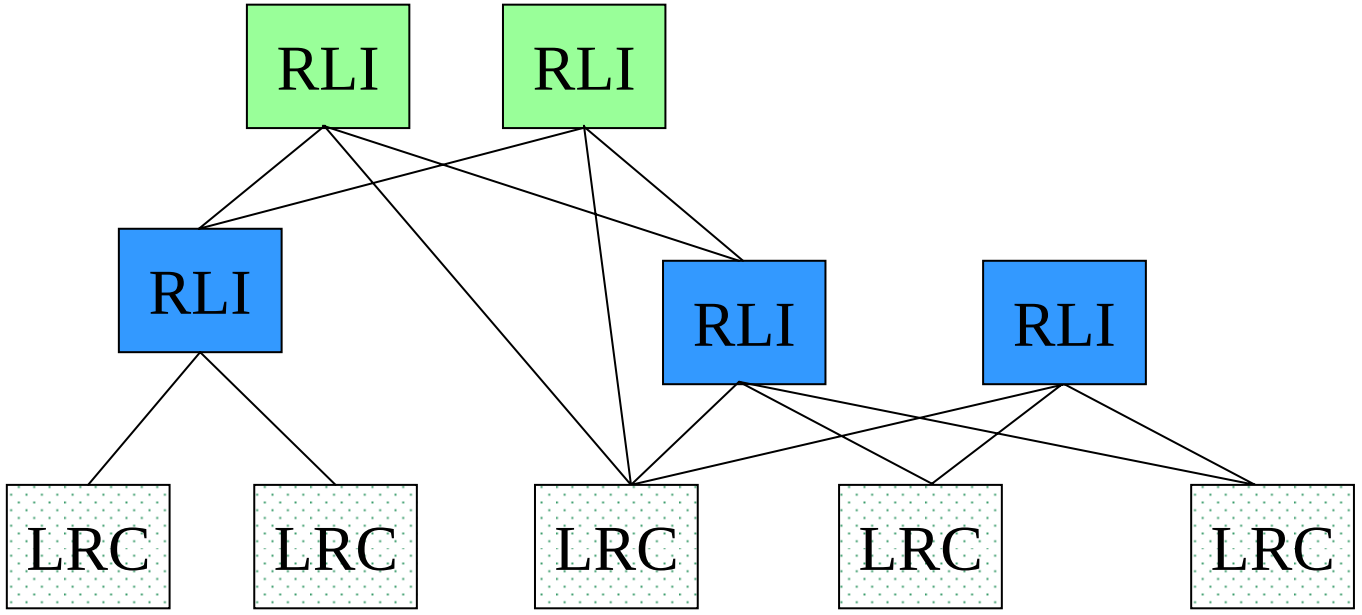


Data Management future : Reptor: The Next Generation Replica Manager



Replica Location Service

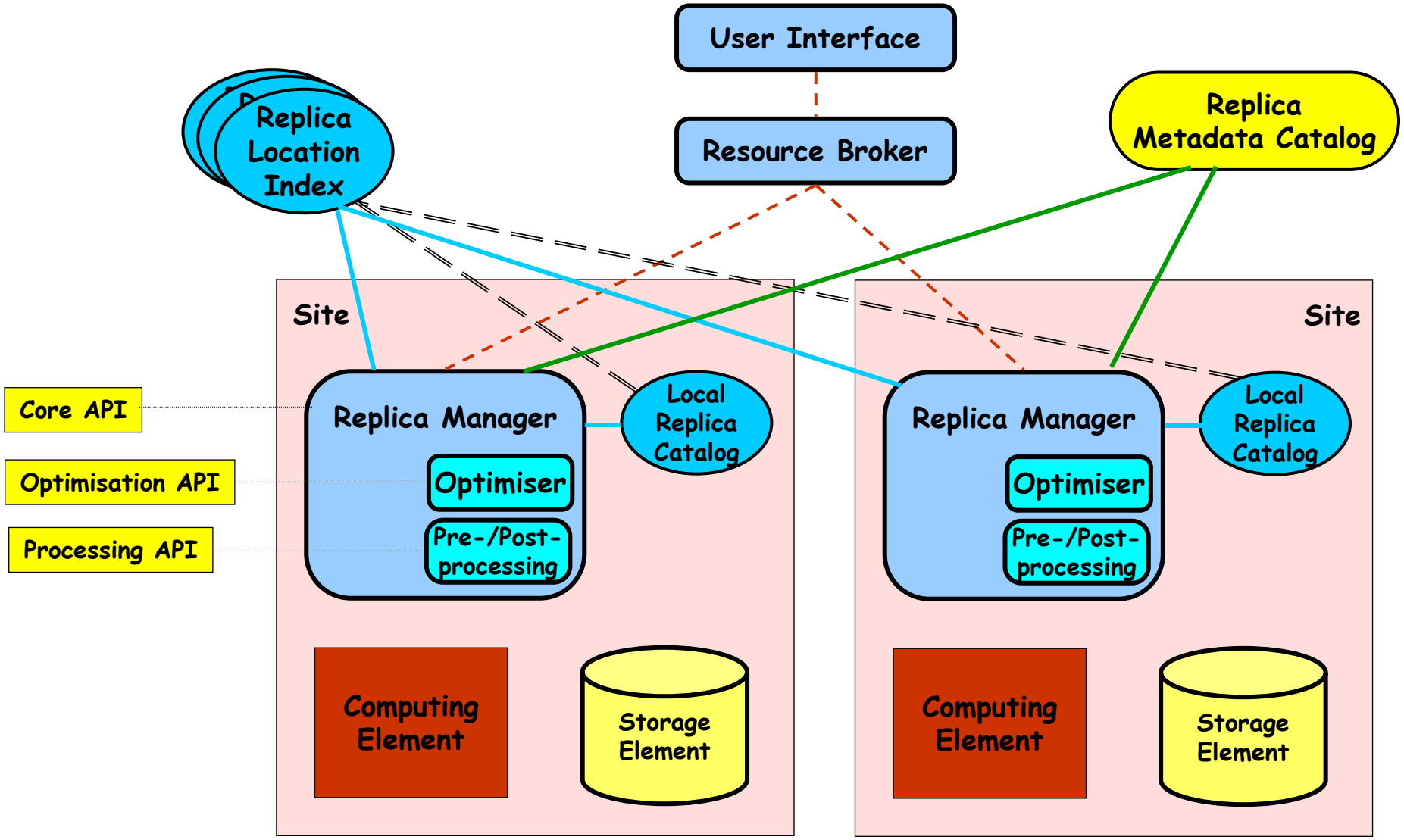
- Scalable, distributed replica location service (RLS)
- Local replica catalogs (LRCs) & collective replica location indeces (RLI)
- LRC holds reliable local state, RLI unreliable collective state with soft-state update



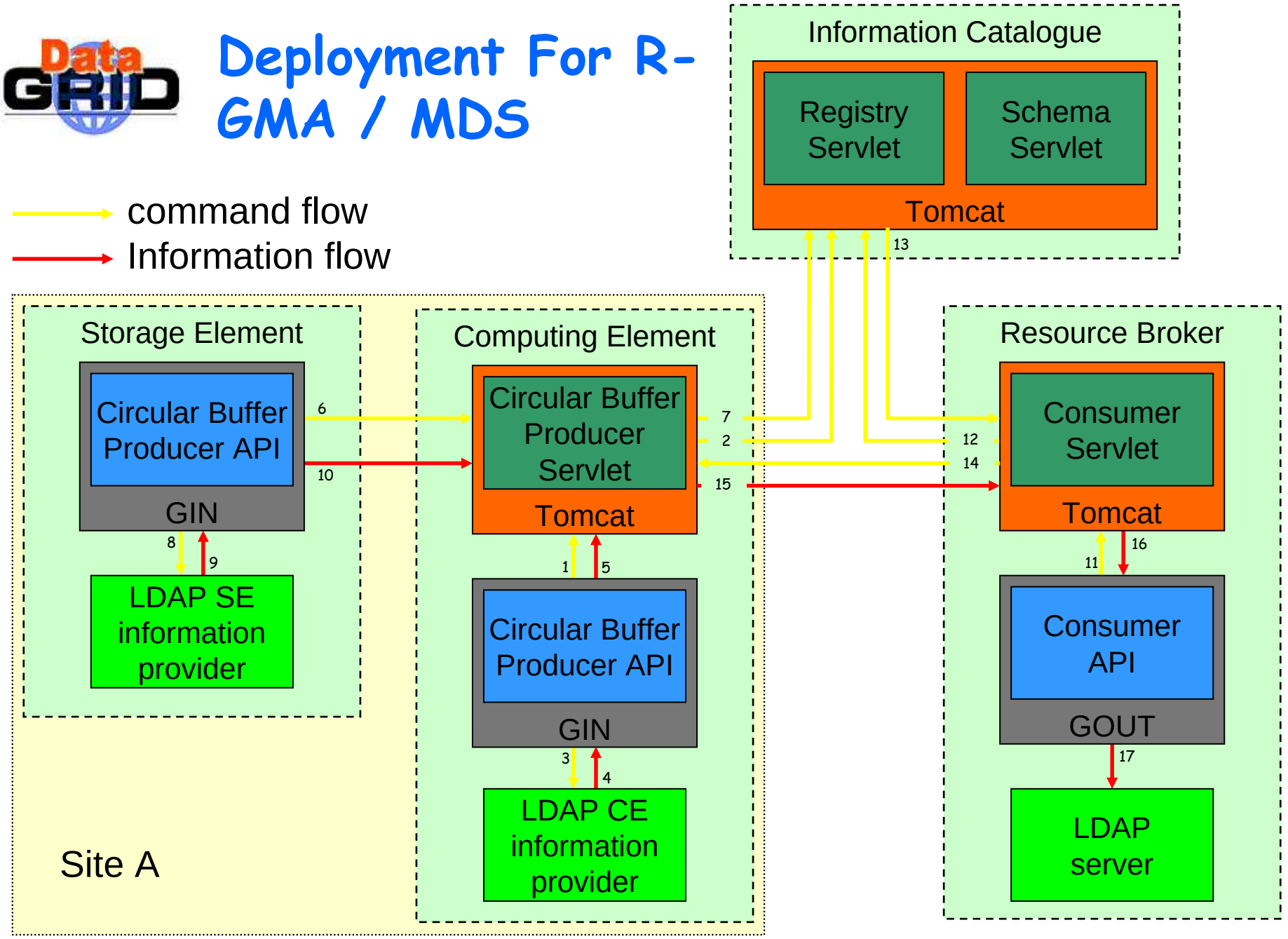
Replica Meta Data Catalog

- Code named *RepMeC*
- Built upon existing Spitfire infrastructure
- Stores all relevant information for the replication services to function
  - Metadata on data
    - Master copy location and locks
    - Versioning data
    - User information
    - Access information
  - Metadata on service internals
    - Transaction steps and locks
    - Consistency layers
    - Subscription based replication info

Replication Services Architecture

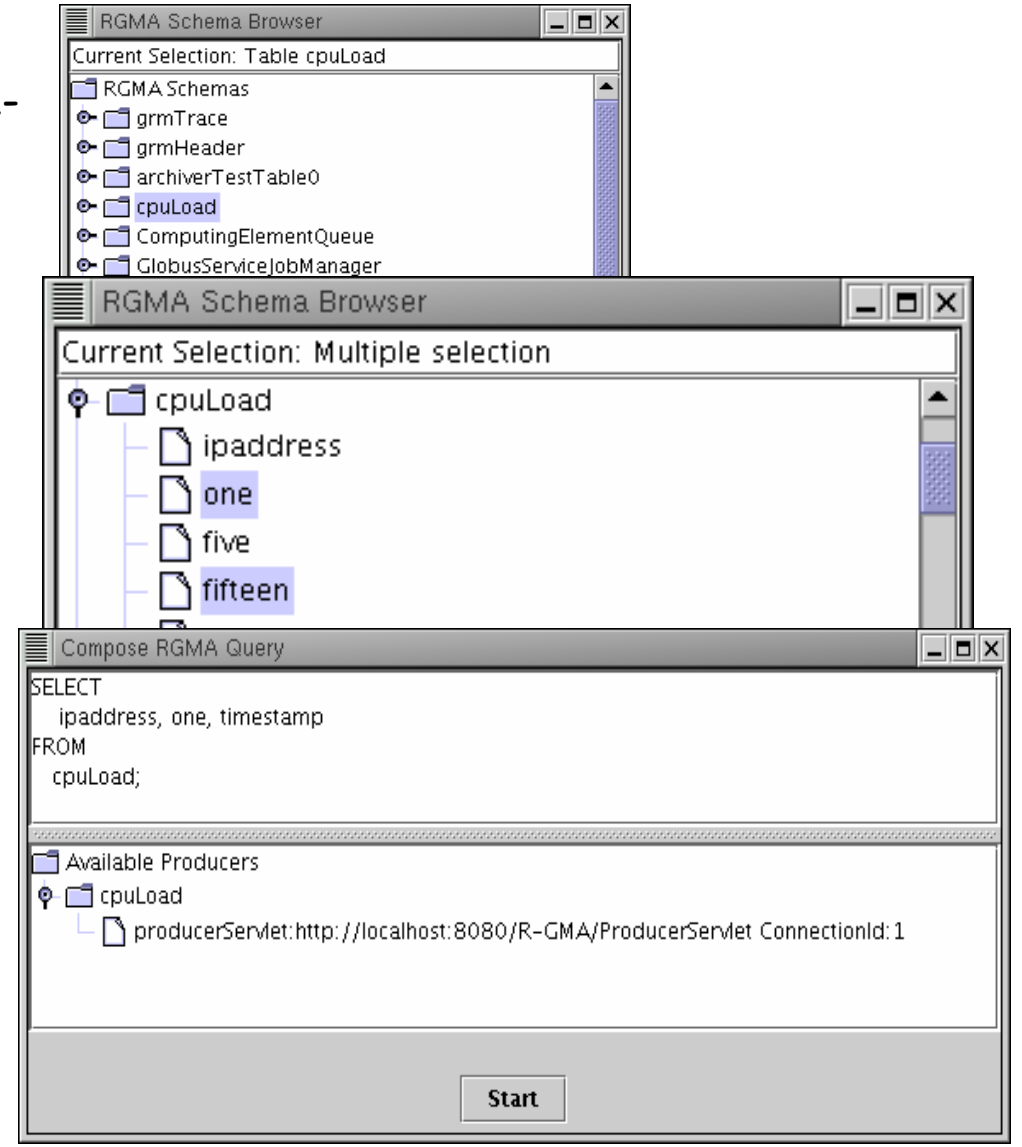


Deployment For R-GMA / MDS



Information Systems future : Pulse (in release 1.3)

- Pulse is a graphical interface to the R-GMA registry
- When a table is selected a Simple Query can be issued
- A Consumer is created within Pulse that retrieves information from R-GMA belonging to the selected table
- More complex queries can be defined by selecting specific attributes of tables
- In a composer window, an SQL selection statement corresponding to the actual selection can be further edited by hand
- The user may choose to have a single result set returned or to have data streamed back.



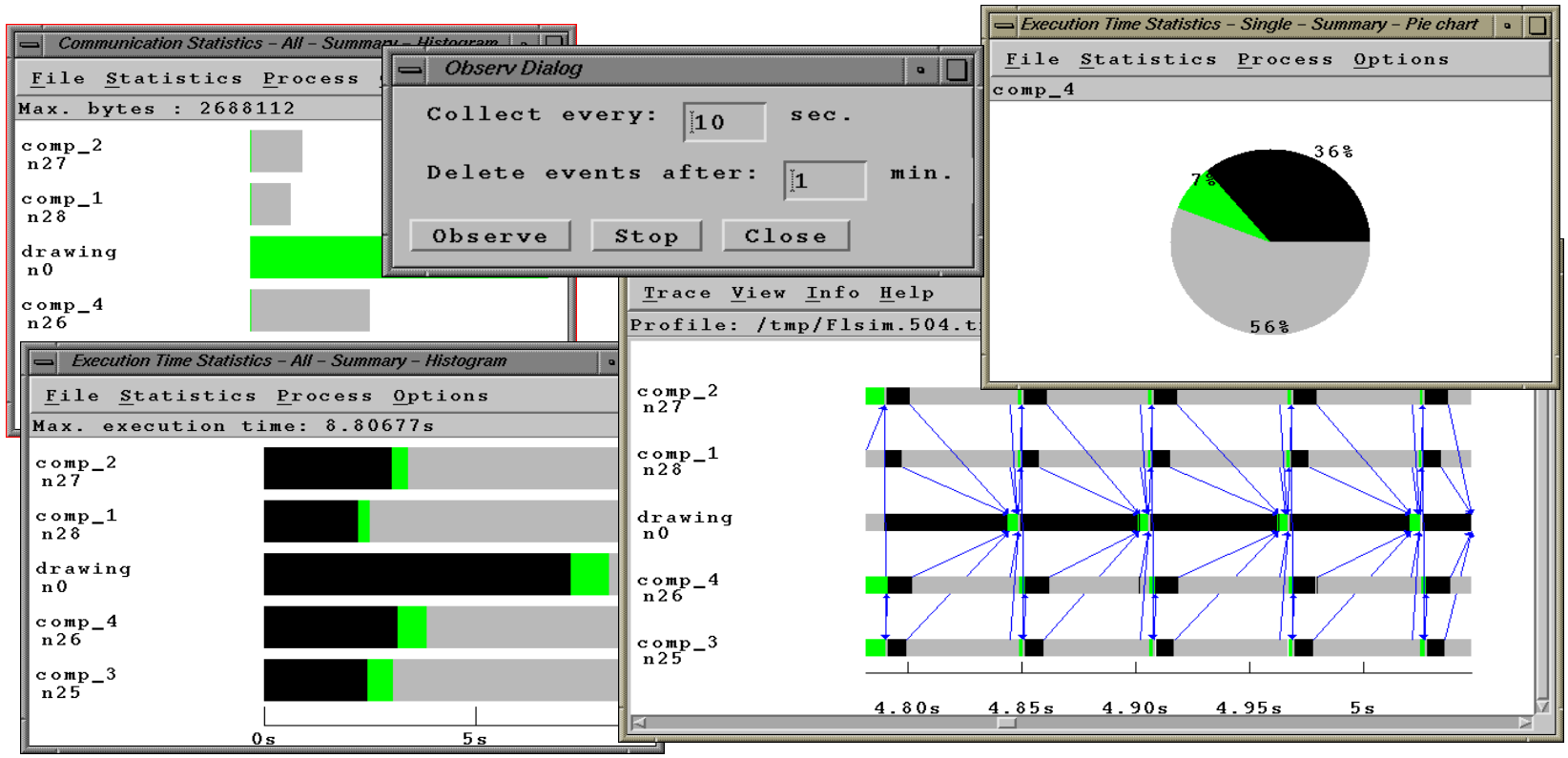
R-GMA & GRM/PROVE

- GRM – Provides C API for application monitoring
- PROVE - Visualisation tool

- GRM
  - Application is instrumented using the C library
  - Produces statistics about different processes and blocks of code which are written to a trace file
  - At present the trace file has to be passed back for interpretation at the end of the job
  - The future integration with R-GMA will allow the transfer of the trace information during run time

PROVE

- Statistics include
  - execution time for individual processes
  - execution time for all processes
  - amount of communications for all processes



Fabric Management

- Future Fabric Management developments:
  - Installation and configuration Management
  - Gridification
  - Monitoring and Fault Tolerance
  - Resource Management

Fabric Installation and Configuration Mgt

Next Release:

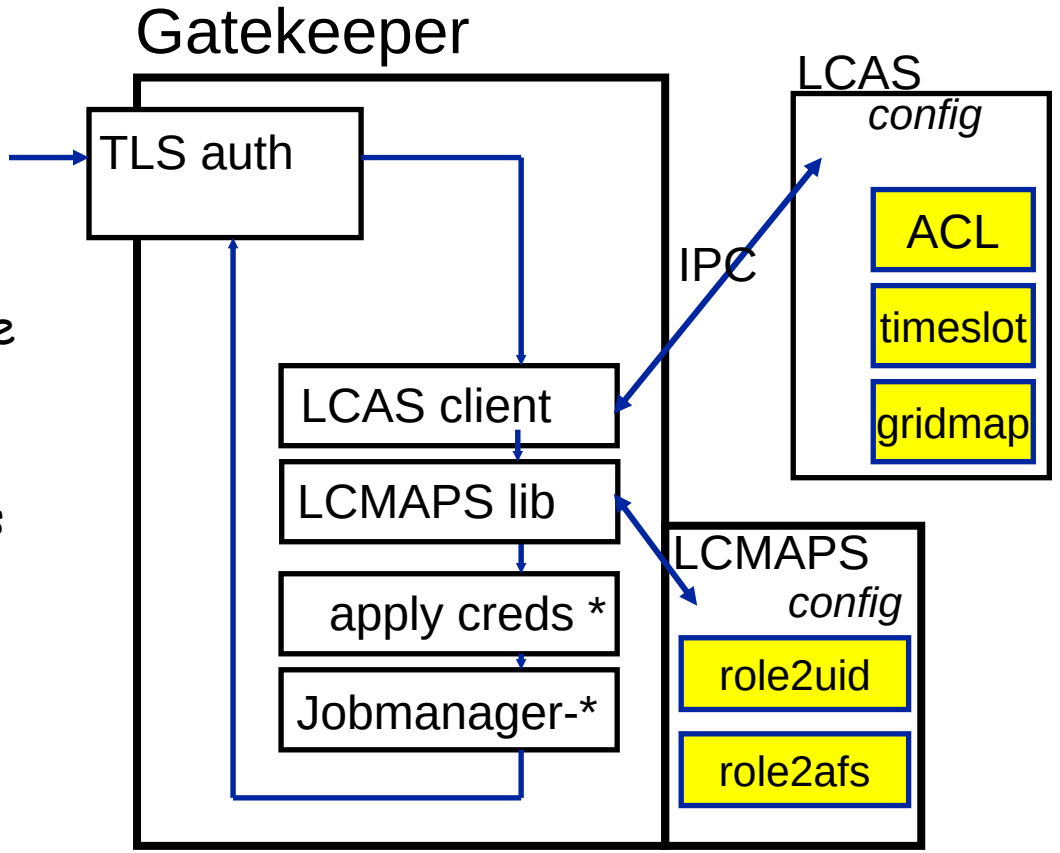
- LCFGng (new generation): Support for RedHat 7.2
- Integration with Monitoring

After:

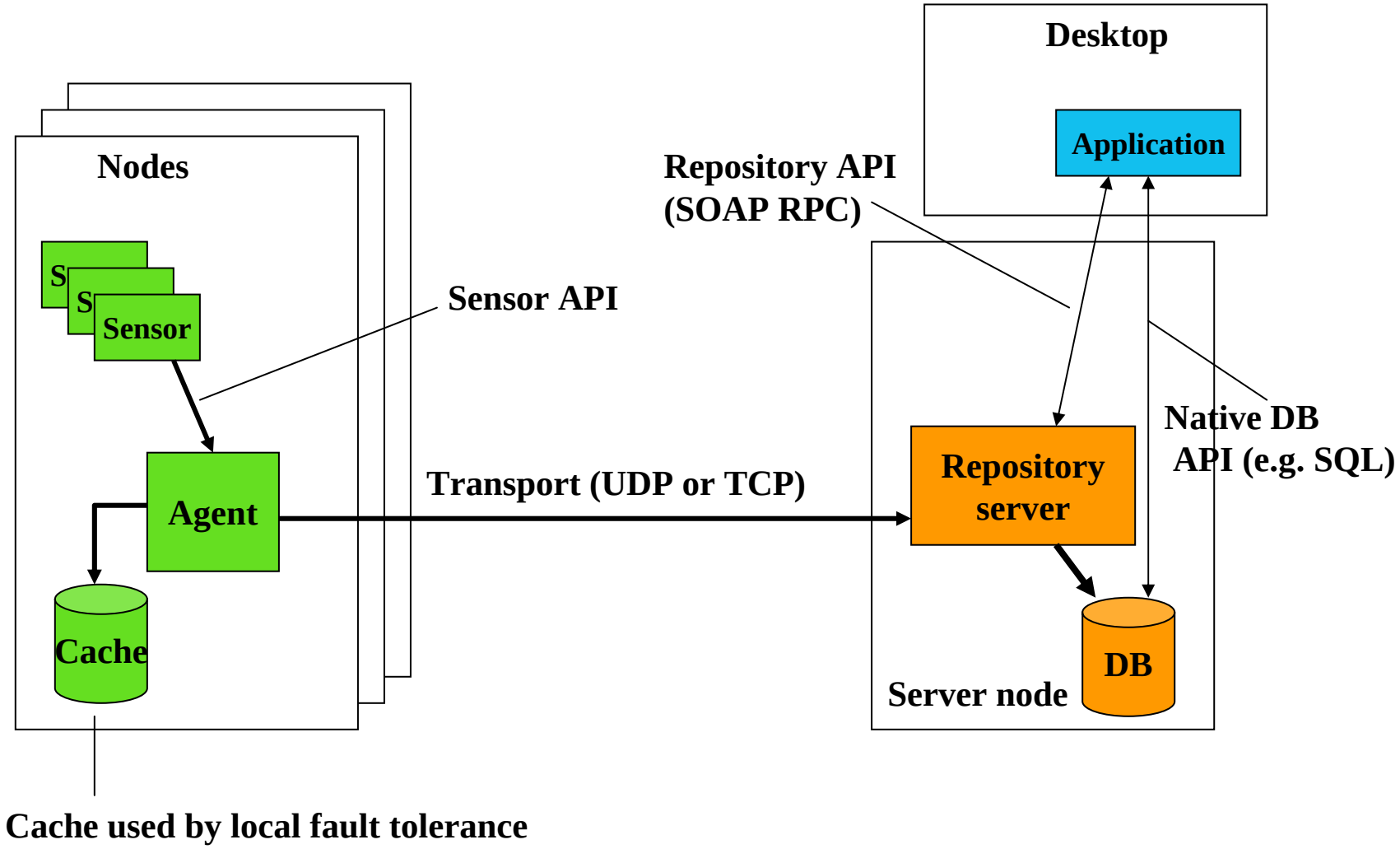
- The LCFG configuration language and compiler will be replaced by new EDG developments
  - This will require changes in existing components everywhere the configuration is accessed.
- Configuration can be made available across components in a global configuration tree, altogether with private namespaces (current LCFG approach)

Fabric Gridification

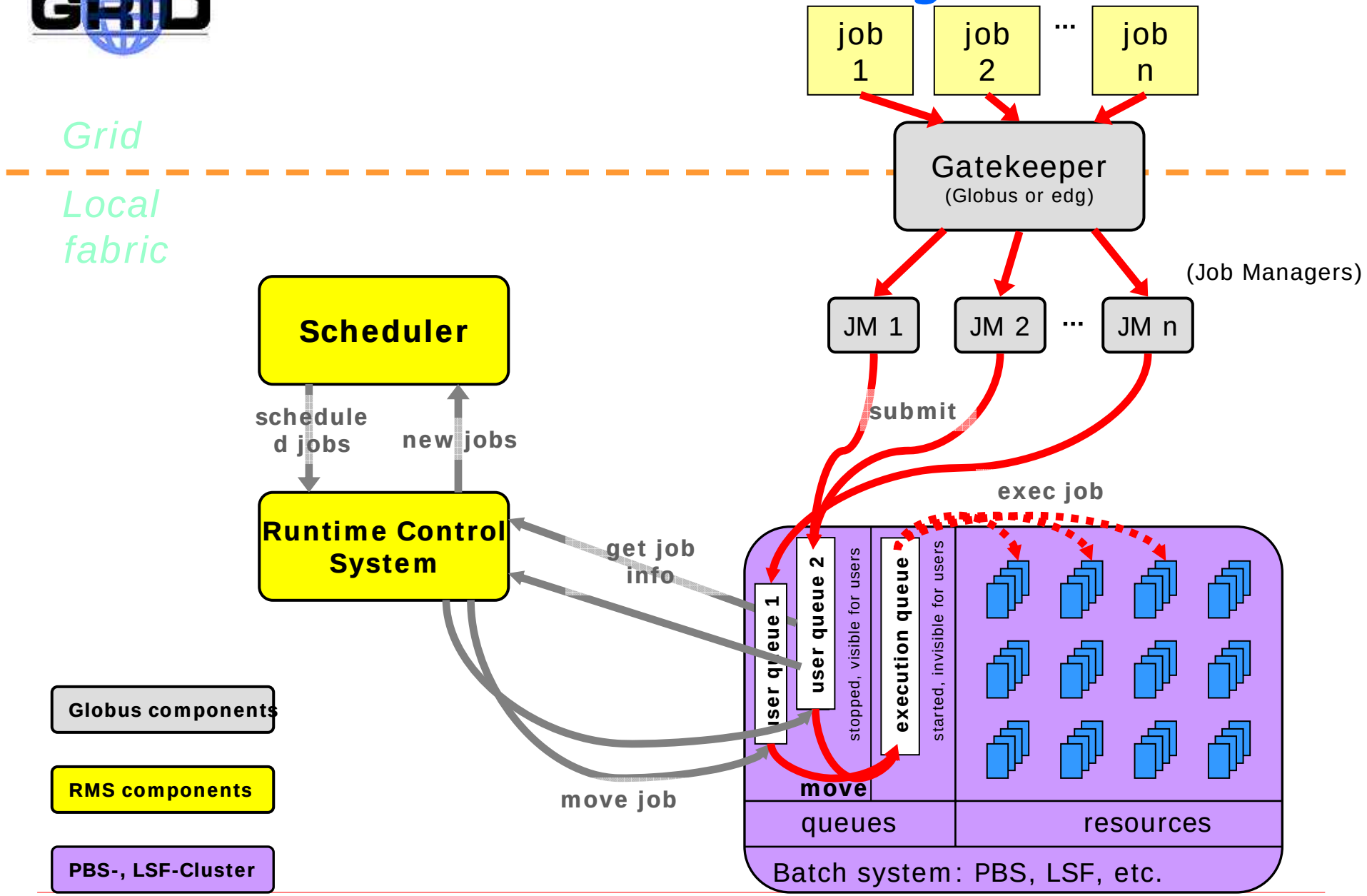
- LCAS: Local Centre AuthZ Service
  - Policy-based authorization
  - Plug-able framework
  - Separate daemon
- LCMAPS:
  Local Credential MAPping Service
  - Maps credentials and roles to local accounts and capabilities
  - Support for AFS, Kerberos tokens
  - Library implementation
  - Enhances gridmapdir
- Requires modified Gatekeeper
  - Improved error&status handling
  - Getting a useful message to the user
- Job repository, FLIDS, FABNAT > EDG 2.x



Fabric Monitoring: FMON



Fabric Resource Management



OGSA: A major development in distributing computing resources and services

- A new conceptual framework to distribute computing and services bringing together aspects of web services and grid computing
- The Open Grid Services Architecture is based on the definition of a **Web Service** as **a set of related application functions that can be programmatically invoked over the Internet.**
- To invoke a Web service, applications make use of the service definition information in a Web Services Description Language (**WSDL**) document
- Work on the impact and the possible implementation of an OGSA-based GRID is being carried out (to define possible architectural frameworks and agree on standards) within GGF

The Web Service architecture

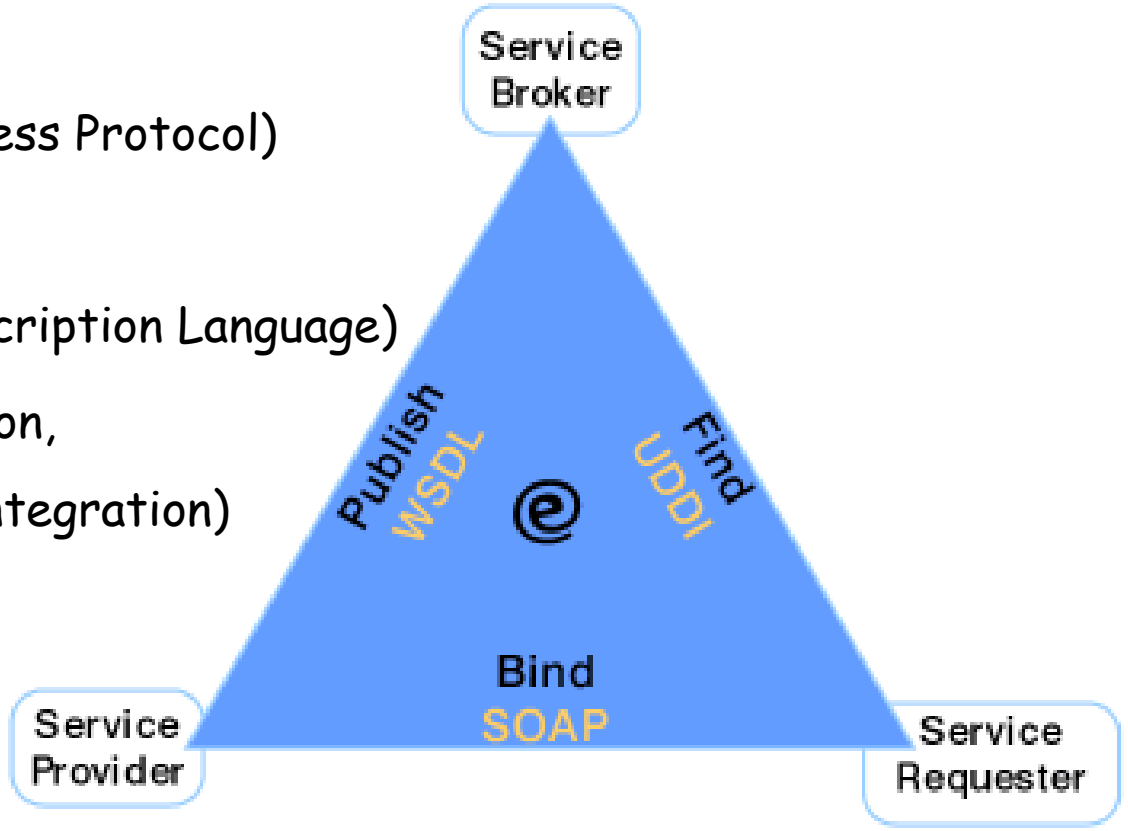
- Three primary players , pillars
  - 1. Providers of the services
  - 2. Directory functions, i.e. Service Broker
  - 3. Service Requesters

SOAP (Simple Object Access Protocol)

interconnects 1,2,3

WSDL (Web Services Description Language)

UDDI (Universal Description,
Discovery & Integration)



OGSA Features vs. Web Services

- Web Services is a conceptual framework to access services to build dynamic applications over the internet, have them executed
- Dynamic (in the WS scheme) means here we do not necessarily know the format of all the information which will be involved along the path done by our application while executing, but we will access this information anyhow. This is done through **a query to the UDDI directory**.
- *OGSA* is further concerned by the **creation of transient instances of web services**, by the management of service instances, to address the real issue of creating and destroying dynamically accessible interfaces to the states of distributed activities.

OGSA and DataGRID

- Globus has announced that the next major version of their toolkit (version 3) will be based on OGSA structure
  - Beta release foreseen for Spring 2003
- DataGRID members are participating to the OGSA specifications
- Mapping between existing DataGRID middleware components and OGSA and being defined and we are following closely the evolution of OGSA

Outlook

- The work is not finished!
  - second part of EDG project will be at least as stimulating and challanging as the first part was.
- Advances are planned for all aspects of the EDG middleware.
- The project is following the development of the OGSA paradigm for distributed computing.

Interaction with Sister Projects

- CrossGrid
  - Using the same security certs.
  - Testbed sites install EDG software
    - Extending it for needs of intensive interactive applications
  - Participating in the EDG testing activities
  - Representatives in each projects architecture & management groups
- DataTAG (EDT)
  - EDT is deploying EDG sw to investigate inter-operability with US projects (iVDGL, GriPhyN, PPDG)
  - Results feedback into EDG software releases
    - (e.g. GLUE compatible information providers/consumers)
- NorduGrid
  - Using the same security certs.
  - Involved in EDG architecture work
    - Good ideas for gatekeeper and MDS configuration
    - Helped develop GDMP and GSI extensions for Replica Catalog
    - Involved in Glue schema work
    - Security policy
  - Mware testing
  - Working in WP8 (HEP applications)
- iVDGL/GriPhyN/PPDG
  - US members in EDG architecture group
  - Looking for common packaging and toolkit usage solutions

No strict boundaries with a large cross-fertilization of ideas, software and people
DataGRID is learning from the experiences in these projects