Practicals on R gLite

Lightweight Middleware for Grid Computing

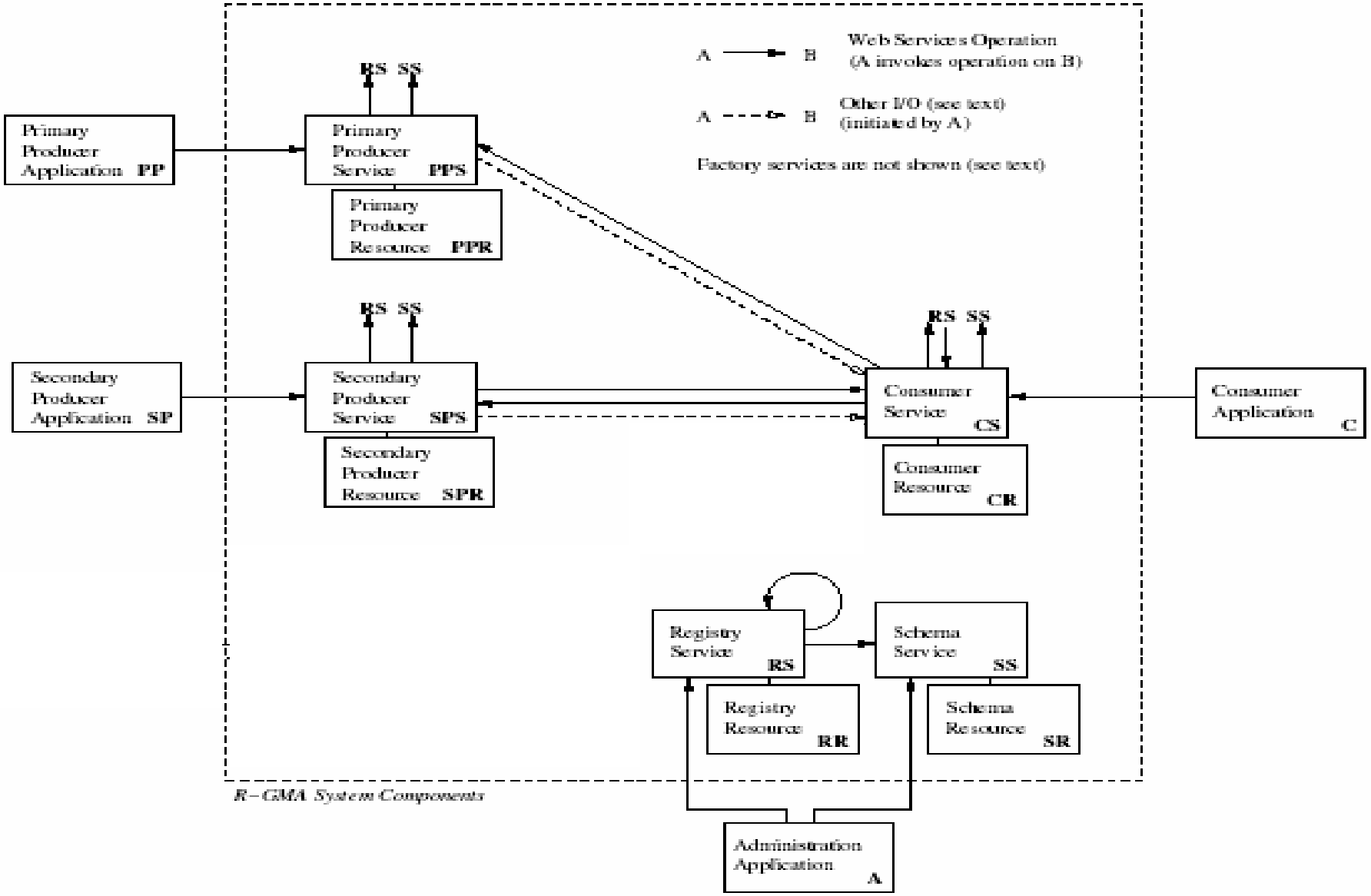
Valeria Ardizzone
INFN

First gLite tutorial on GILDA, Catania, 13-15.06.2005

www.eu-egee.org

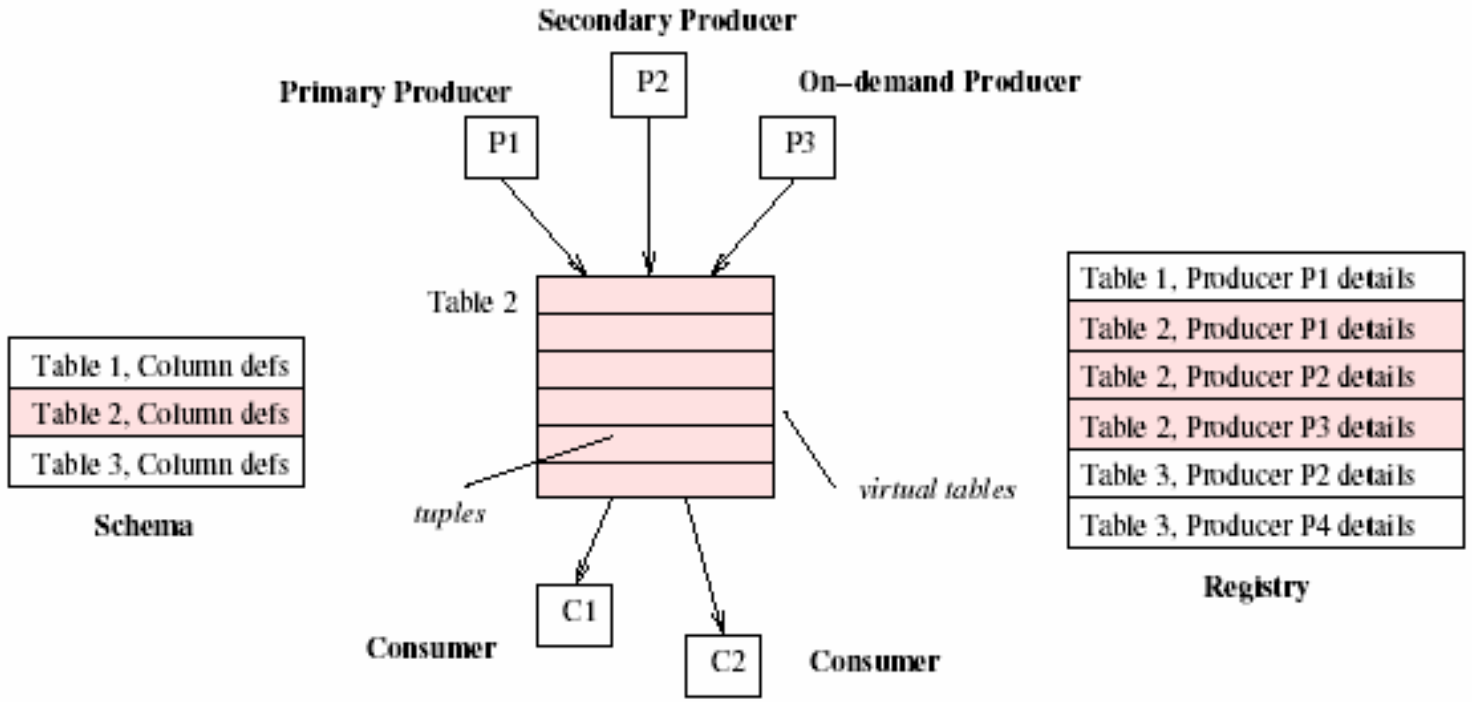
Information Society

Overview


A → B
Web Services Operation
(A invokes operation on B)
A ⇢ B
Other I/O (see text)
(initiated by A)
Factory services are not shown (see text)
Primary Producer Application PP
RS SS
Primary Producer Service PPS
Primary Producer Resource PPR
Secondary Producer Application SP
Secondary Producer Service SPS
Secondary Producer Resource SPR
Consumer Service CS
Consumer Resource CR
Consumer Application C
Registry Service RS
Registry Resource RR
Schema Service SS
Schema Resource SR
Administration Application A
R-GMA System Components

R-GMA Virtual Database

- R-GMA models the information infrastructure of a Grid as a set of
- **Consumers,** who request information
- **Producers,** who provide information
- and a single **Registry,** which mediates the communication between producers and consumers.



FUNCTIONALITY NOT YET IMPLEMENTED

- The API's in the current release of R-GMA are wrappers on top of the old EDG API's and servlets, so not all new R-GMA functionality is yet available. Specifically, the following functions are not available and will raise an appropriate exception if you try to use them:
- FILE storage for producers
- LATEST/HISTORY producers with non-DATABASE storage
- Combined LATEST and HISTORY producers
- HISTORY and LATEST retention periods (functionality is approximated)
- Multiple VOs (just pass NULL for the VO lists)
- TableAuthorization (just pass NULL)
- Schema replication is also not yet implemented.

R-GMA Command Line Tool

- The R-GMA command line tool provides simple shell-like access to the R-GMA distribuited information and monitoring system.
- To Start the R-GMA command line tool, run the simple command “rgma” on your $home:

```
[glite-tutor] /home/vardizzo > rgma

Welcome to the R-GMA virtual database for Virtual Organisations.
You are connected to the following R-GMA registry services:

https://rgmasrv.ct.infn.it:8443/R-GMA/RegistryServlet

Type "help" for a list of commands.

rgma>
```

- **The command line tool can be used in batch mode in two ways:**

rgma -c <command>

Executes <command> and exits. The -c option may be specified more than once in which case the commands are executed in the order they appear on the command line.

rgma -f <file>

Executes commands in <file> sequentially then exits. Each line should contain one command.

INFORMATION COMMANDS

- Displays help for a specific command

rgma>help <command>

- Exit the R-GMA command line

rgma>exit or quit

- To show a list of all table names:

rgma> SHOW TABLES

- To show information about a table MyTable:

rgma> DESCRIBE MyTable

- To show a table of properties for the current session:

rgma> SHOW PROPERTIES

- To show a list of all R-GMA producers that produce the table MyTable:

rgma> SHOW PRODUCERS OF MyTable

Query Consumer

- The subset of SQL understood by R-GMA is that defined by the SQL92 Entry Level standard.
- The algorithm used by the mediator makes a distinction between simple and complex queries, as defined below. Continuous queries must always be simple. Latest and history queries may be complex, and any SQL command processor used by an R-GMA producer that supports latest or history queries, must support complex queries.

Simple Query

A simple query is defined SQL SELECT statement without the following:

- Joins: only one table is allowed
- DISTINCT, HAVING, GROUP BY, ORDER BY, UNION, INTERSECT, MINUS
- Aggregation operators such as AVG, MIN, MAX, COUNT, SUM
- Calculations, other than numerical ones involving only the operators: +, -, *, / and **
- WHERE clauses involving operators other than: =, >, <, >=, <=, and AND

Complex Query

- A complex query is anything permitted by the SQL92 Entry Level standard.

In the short term, R-GMA will not support this type of features.

QUERYING DATA

- Querying data uses the standard SQL SELECT statement, e.g.

rgma> SELECT * FROM ServiceStatus

- The behaviour of SELECT varies according to the type of query being executed. In R-GMA there are three basic types of query:

LATEST Queries only the most recent tuple for each primary key

HISTORY Queries all historical tuples for each primary key

CONTINUOUS Returns tuples continuously as they are inserted

- The type of query can be changed using the SET QUERY command:

rgma> SET QUERY LATEST

rgma> SET QUERY CONTINUOUS

- The current query type can be displayed using

rgma> SHOW QUERY

Maximum AGE of tuples

- The maximum age of tuples to return can also be controlled. By default there is no limit. To limit the age of latest or historical tuples use the SET MAXAGE command which takes a value and a time unit (seconds, minutes, hours, days - default is seconds). The following are equivalent:

rgma> SET MAXAGE 2 minutes

rgma> SET MAXAGE 120

- The current maximum tuple age can be displayed using

rgma> SHOW MAXAGE

- To disable the maximum age, set it to none:

rgma> SET MAXAGE none

Query Timeout

- The final property affecting queries is the timeout. This controls how long the query will execute for before exiting automatically. For a latest or history query the timeout exists to prevent a problem (e.g. network failure) from stopping the query from completing. For a continuous query, the timeout indicates how long the query will continue to return new tuples. The default timeout is 1 minute and it can be changed using

rgma> SET TIMEOUT 3 minutes

rgma> SET TIMEOUT 180

- The current timeout can be displayed using

rgma>SHOW TIMEOUT

INSERTING DATA

- The SQL INSERT statement may be used to add data to the system:

rgma> INSERT INTO Table VALUES ('a', 'b', 'c', 'd')

- In R-GMA, data is inserted into the system using a Producer component which handles the INSERT statement. Using the command line tool you may work with one producer at a time. If you change the properties of the producer, a new one is created and all INSERT statements will go to the new producer.
- Producers may be configured to answer the three different types of query. All producers can answer continuous queries, but ability to answer latest and history queries is optional and is changed using :

rgma> SET PRODUCER latest
rgma> SET PRODUCER latest history
rgma> SET PRODUCER continuous

Producer

- The current producer type can be displayed using:

rgma>show producer

- For a producer that can answer latest and/or history queries, tuples must be deleted after some time interval to prevent the system becoming full of old data. This is controlled by the latest and history retention periods for a producer. The producer will return tuples to Latest queries provided they are younger than the latest retention period (LRP), and similarly for history queries and the history retention period (HRP).

rgma> SET PRODUCER latestretentionperiod 30 minutes

rgma> SET PRODUCER historyretentionperiod 2 hours

To show the current producer HRP/LRP use:

rgma>SHOW PRODUCER HRP

rgma> SHOW PRODUCER LRP.

Secondary Producer

- Using the R-GMA command line you can work with one secondary producer at a time, which can consume from multiple tables. If the properties of the secondary producer are changed a new one is created, and is automatically set up to consume from the same tables as the previous one.
- To instruct the secondary producer to consume from the table MyTable:

rgma> SECONDARYPRODUCER MyTable

- Like the producer, the secondary producer may be configured to answer latest and/or history queries:

rgma> SET SECONDARYPRODUCER latest

(By default the secondary producer can answer both latest and history queries.)

- The current secondary producer type can be displayed using:

rgma> SHOW SECONDARYPRODUCER

Some Esercises

- Show all tables that are present in the Schema: show tables
- Choose one table where you want insert your information and see the its attribute: describe <NameTable>
- Show which producer is enable at this moment for your type query and set it to latest query:

  show producer

- Some examples of query:

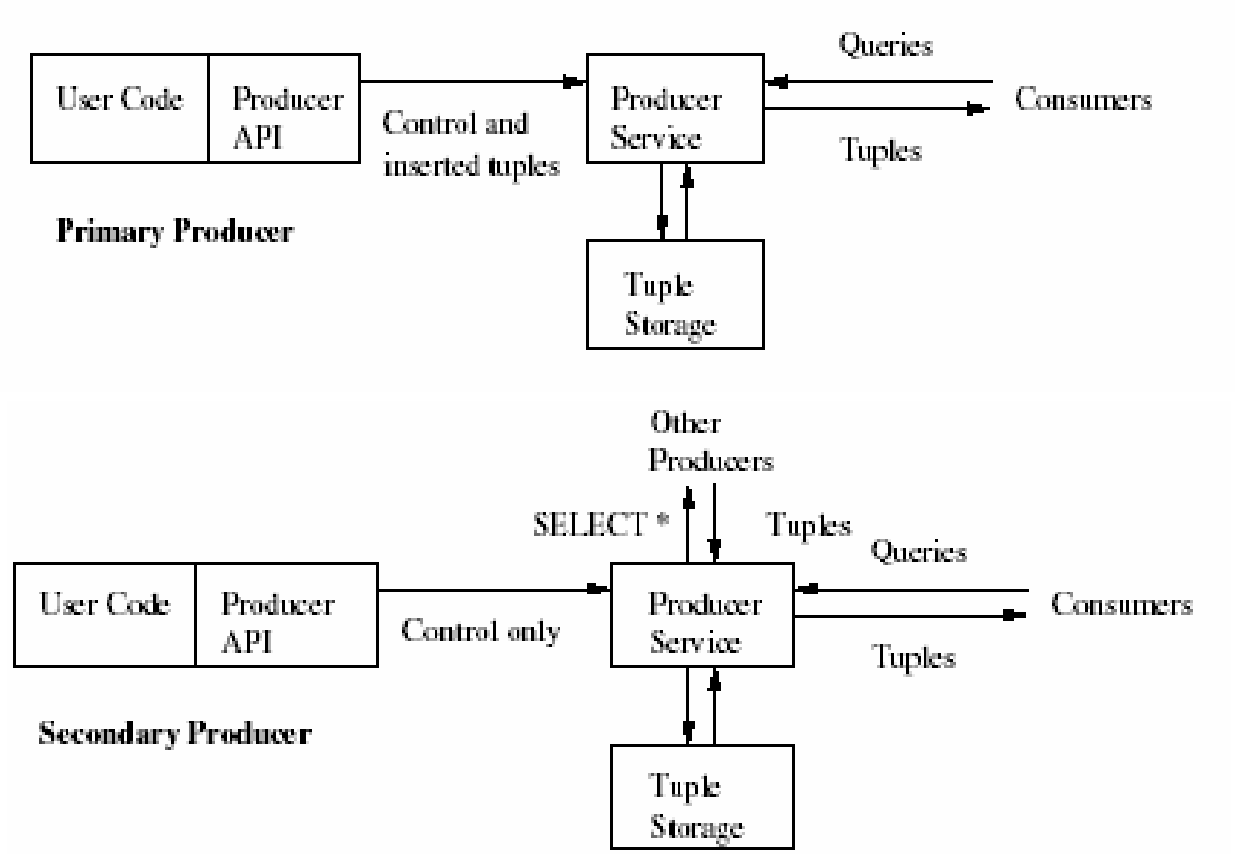
  select * from Site

  select SysAdminContact,UserSupportContact from Site

  select Name from Site where Latitude > 0

  select Endpoint, Type from Site,Service where Site.Name = Service.Site_Name and Latitude > 0

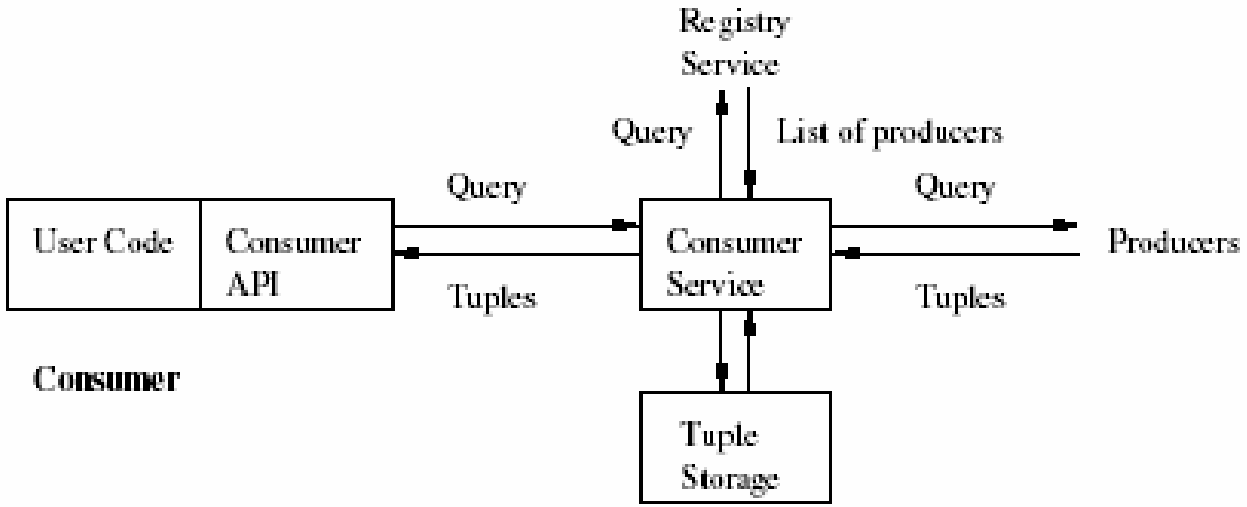
Producer



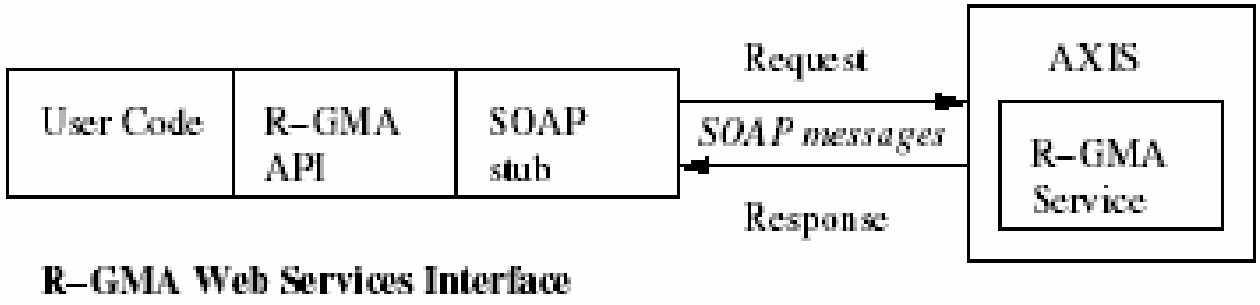
- The producer service is a process running on a server on behalf of the user code;
- Each tuple published by a primary producer carries a time-stamp, added by the producer, which, together with the key columns, is similar to a primary key for the table.

Consumer

- In R-GMA, each consumer represents a single SQL SELECT query on the virtual database.



Web Services Architecture



R–GMA Web Services Interface

- R-GMA provides APIs for Java, C++, C and Python languages, to make it easier for user applications to interact with the R-GMA services.
- Each operation of each service is represented by a method (or function) in the API, and that method simply packages up its parameters into a SOAP message and sends it to the Web Service for execution. Any return values or errors (exceptions) are passed back to the caller. The API transparently manages any authentication required by the server, and looks after the resource identifier.
- Direct interaction between a user application and the R-GMA Web Service
  (by-passing the API) is also supported.