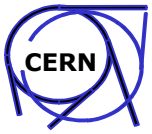
R-GMA for Distribution of Monitoring Data

Piotr Nyczyk, IT/GD CERN

3D Workshop

CERN, 14 December 2004

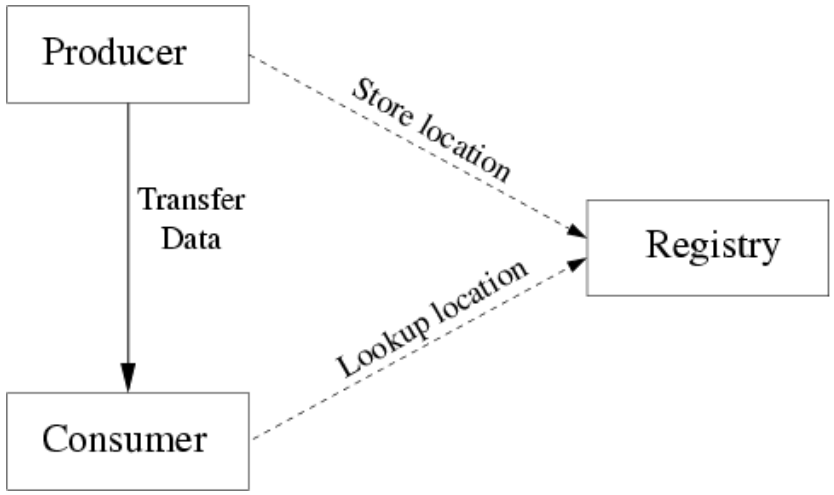


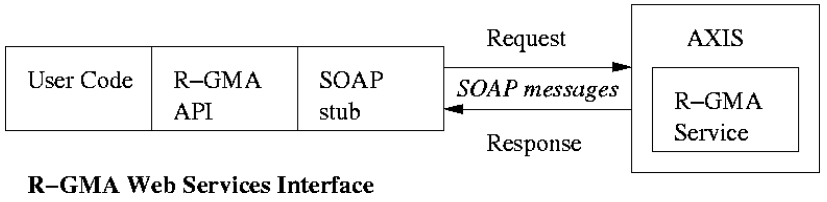
Outline

- R-GMA basics
  - Virtual database concept
  - Types of queries and producers
  - Archivers (Secondary producers)
- R-GMA in EGEE/LCG operations monitoring
  - "Central monitoring bus"
  - MySQL based R-GMA Archivers
  - Reporting tool prototype
- Advantages
- Known issues
- References

R-GMA Basics and key concepts

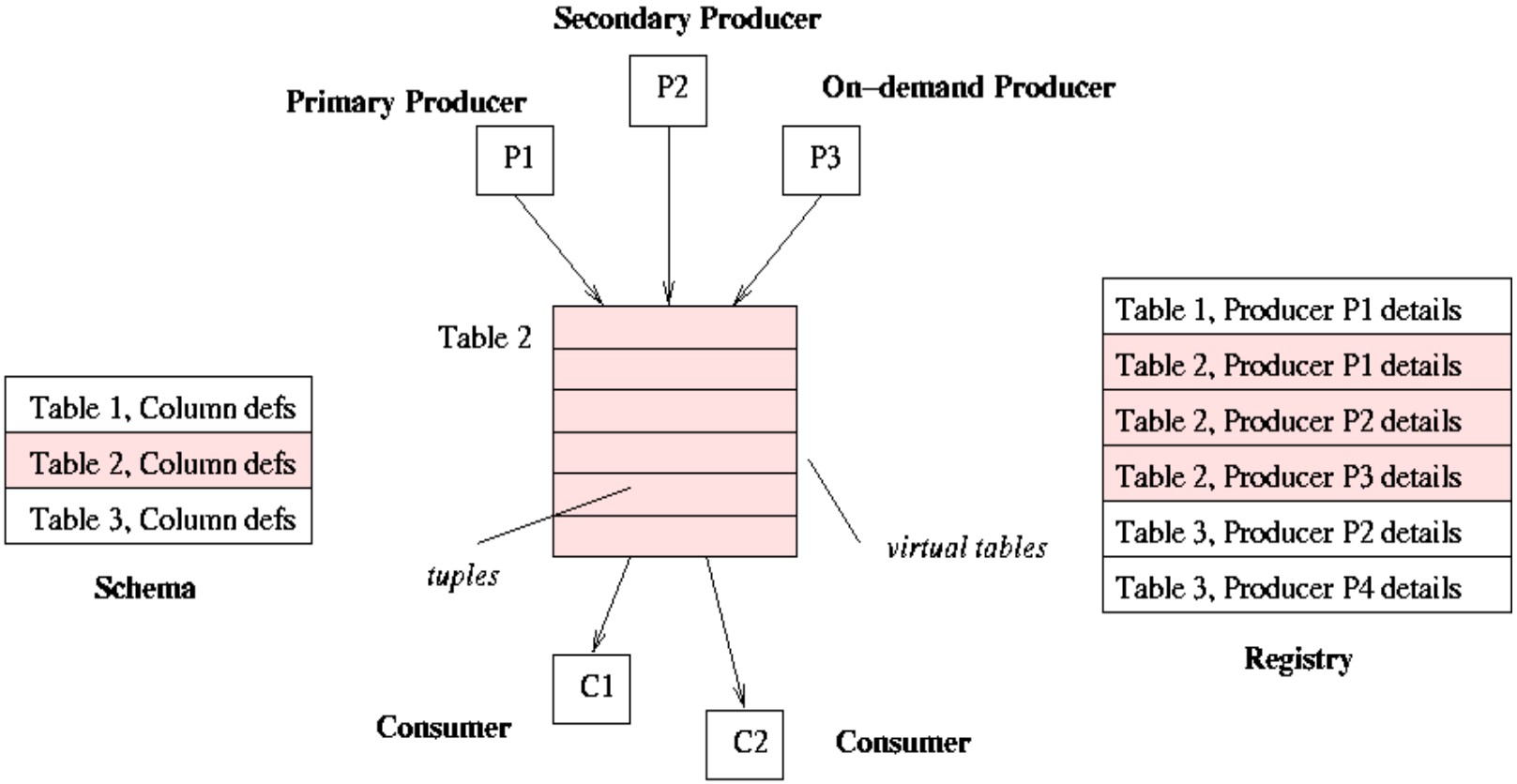
- Monitoring framework for computing Grids – developed by EGEE-JRA1-UK, currently used by EGEE/LCG2 and experiment apps
- Based on GGF GMA definition – Producer-Consumer architecture
- But! With relational data model – the whole system appears as one large relational database
- Data is propagated from Producers to Consumers on many different levels and locations: core centers, regional centers (ROC), resource centers (RC)
- Central Registry is used to locate the data (producers) automatically.
- Webservice interface (servlets)
- However, it is not a general distributed RDBMS!





R–GMA Web Services Interface

Virtual Database concept



Virtual Database concept (cont.)

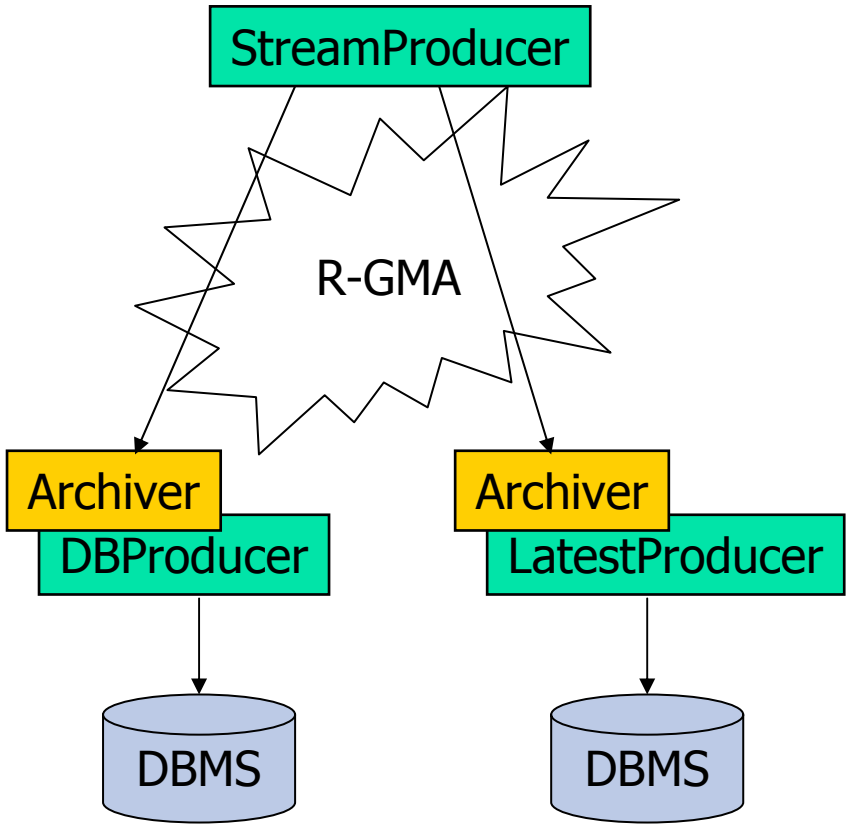
- Consumer Producer architecture
- Uses a subset of SQL for queries and inserts
- All data tuples with *timestamps*
- Data is published by INSERT operation on Producers
- Queries by SELECT operation on Consumers
- No UPDATE and DELETE operations! However by publishing a new tuple with the existing value of primary key can "overwrite" the data.
- WHERE predicate can be associated with each producer to optimize queries (Producer declares what scope of data it will publish)

Types of queries and producers

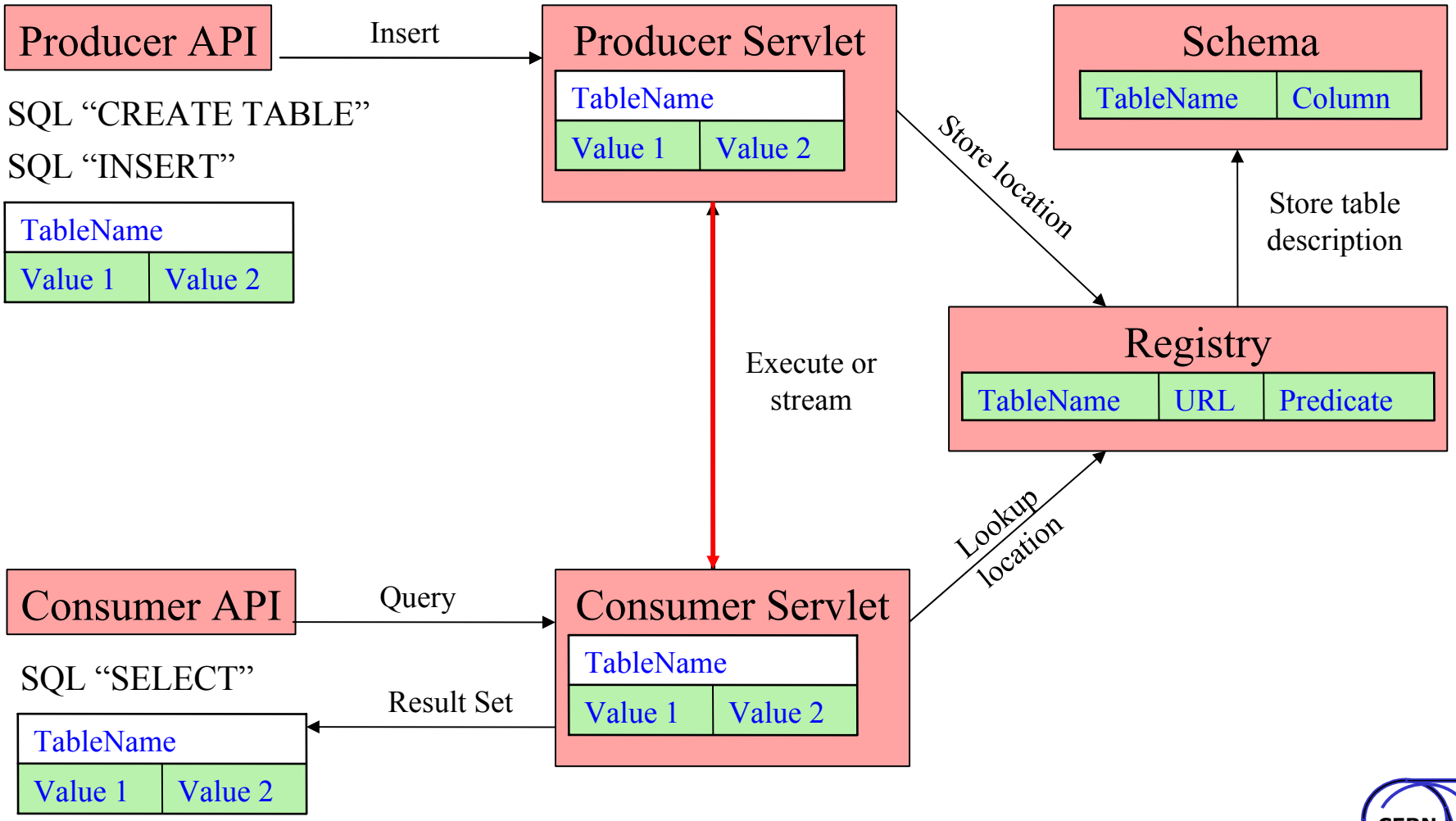
- R-GMA supports three type of queries (extension to SQL syntax):
  - CONTINUOUS SELECT – all new data as it is published
  - LATEST SELECT – last value for given key(s) according to *timestamps*
  - HISTORY SELECT – all historical data
- Corresponding Producers types:
  - *StreamProducer* – published data is "broadcasted" and lives for certain short period, answers CONTINUOUS queries.
  - *LatestProducer* – stores only last value for each key
  - *DBProducer* – stores all published tuples
- DBProducers and LatestProducers are using physical DBMS to store the data (MySQL, others?)

Archivers (Secondary Producers)

- Archiver is a Consumer-Producer pair
- Consumer part is responsible for "catching" new data that is published by StreamProducers
- Producer part stores the data into DBMS (MySQL) and later can answer queries (Latest, History)
- Multiple archivers for fault tolerance (each has his own copy of data)



R-GMA

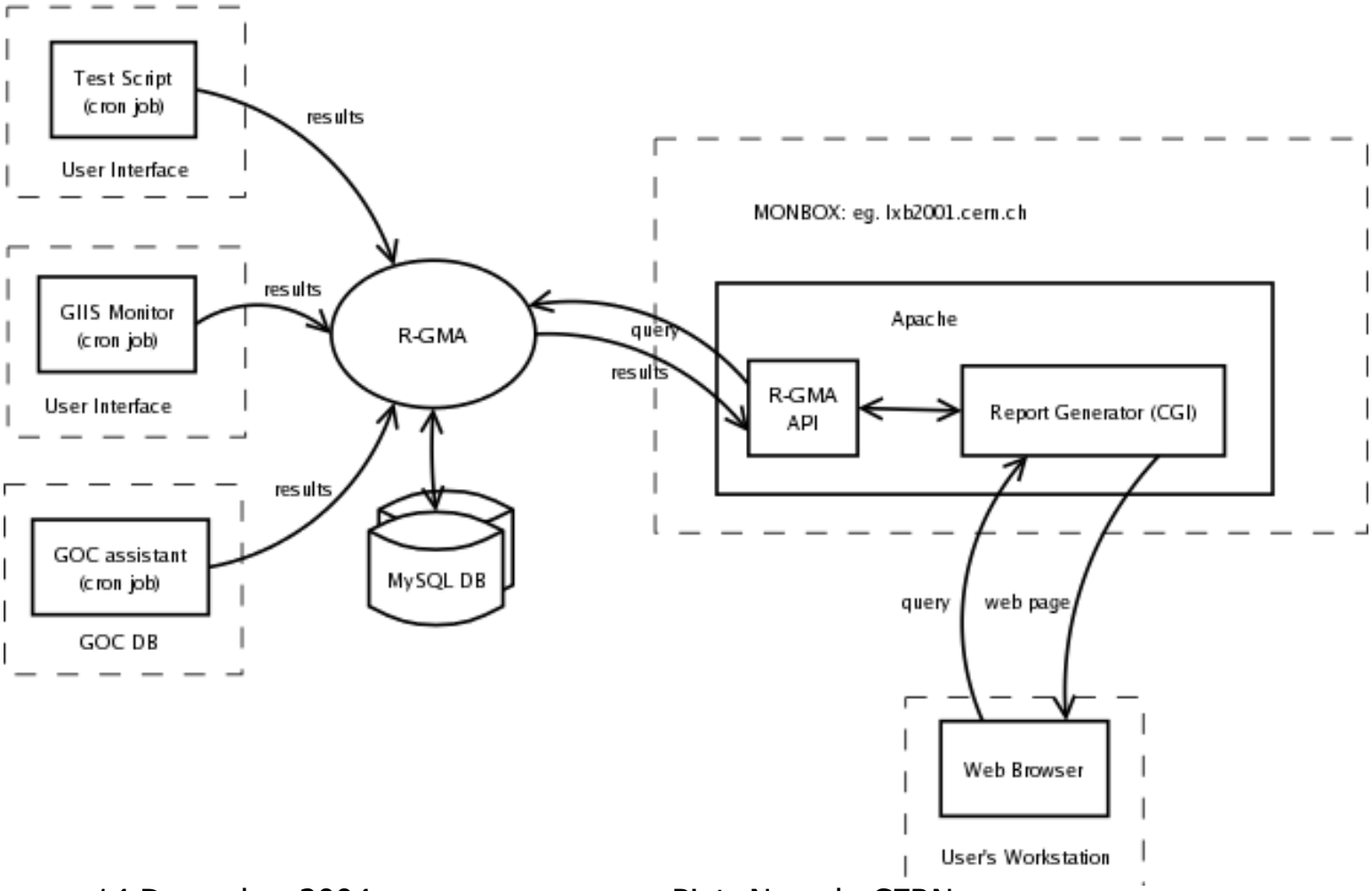

Producer API
Insert
Producer Servlet
TableName
Value 1
Value 2
Schema
TableName
Column
SQL "CREATE TABLE"
SQL "INSERT"
TableName
Value 1
Value 2
Store location
Store table description
Execute or stream
Registry
TableName
URL
Predicate
Lookup location
Consumer API
Query
Consumer Servlet
TableName
Value 1
Value 2
SQL "SELECT"
Result Set
TableName
Value 1
Value 2

R-GMA in EGEE/LCG Grid monitoring

- Motivation:
  - We have a lot of monitoring tools (sensors) at different levels: tests running centrally, agents running on different sites, accounting information coming from Resource Brokers and sites …
  - Data must be easy accessible from a single point (reports)
  - Data must be archived but not in one place!
  - There are lots of monitoring frameworks, but we already have R-GMA infrastructure in EGEE/LCG2
- Decision: „we will use R-GMA as a central bus to distribute data between sensors and reporting tools”

R-GMA as "Central monitoring bus"

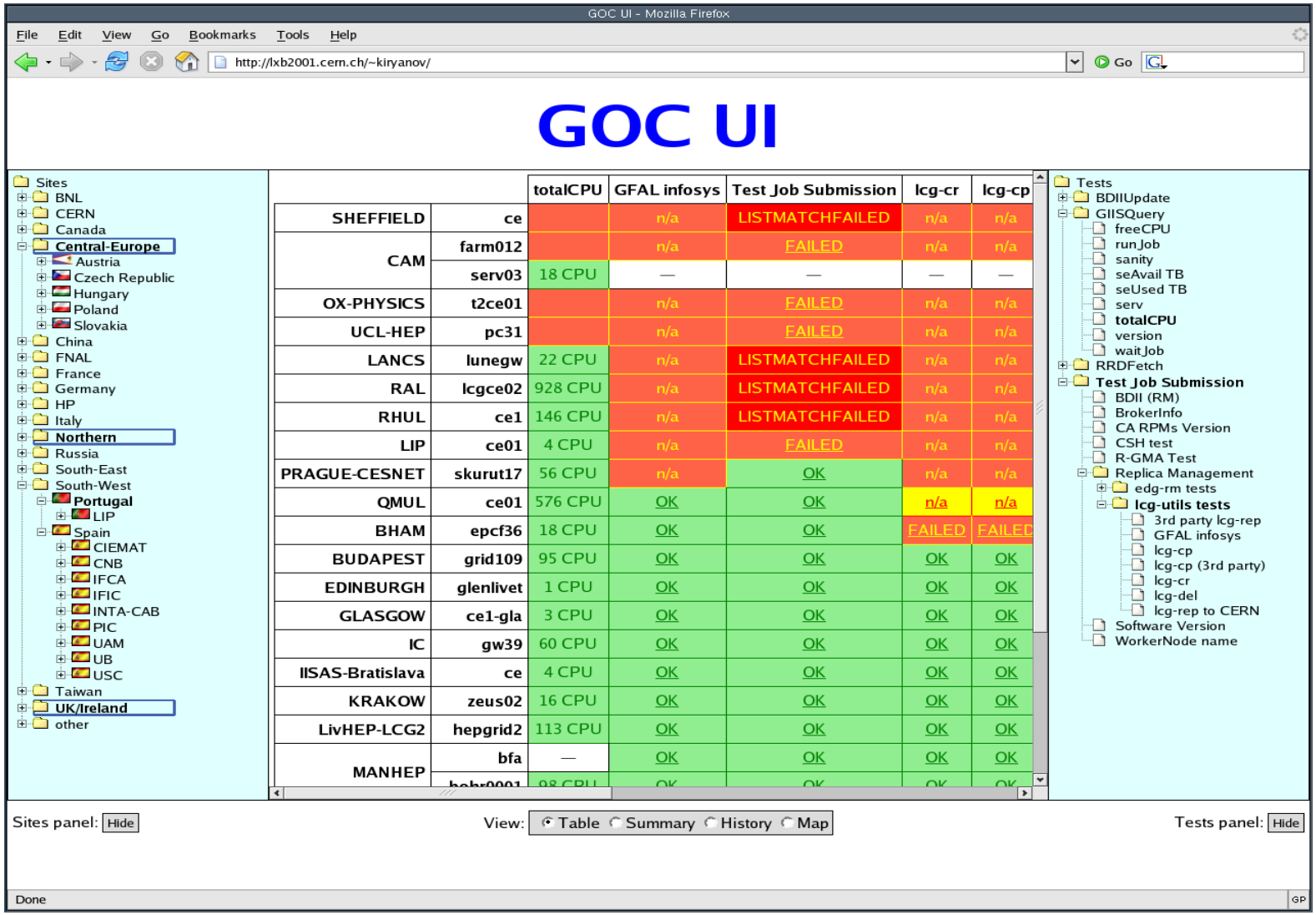
Diagram of R-GMA based monitoring system



MySQL based Archivers

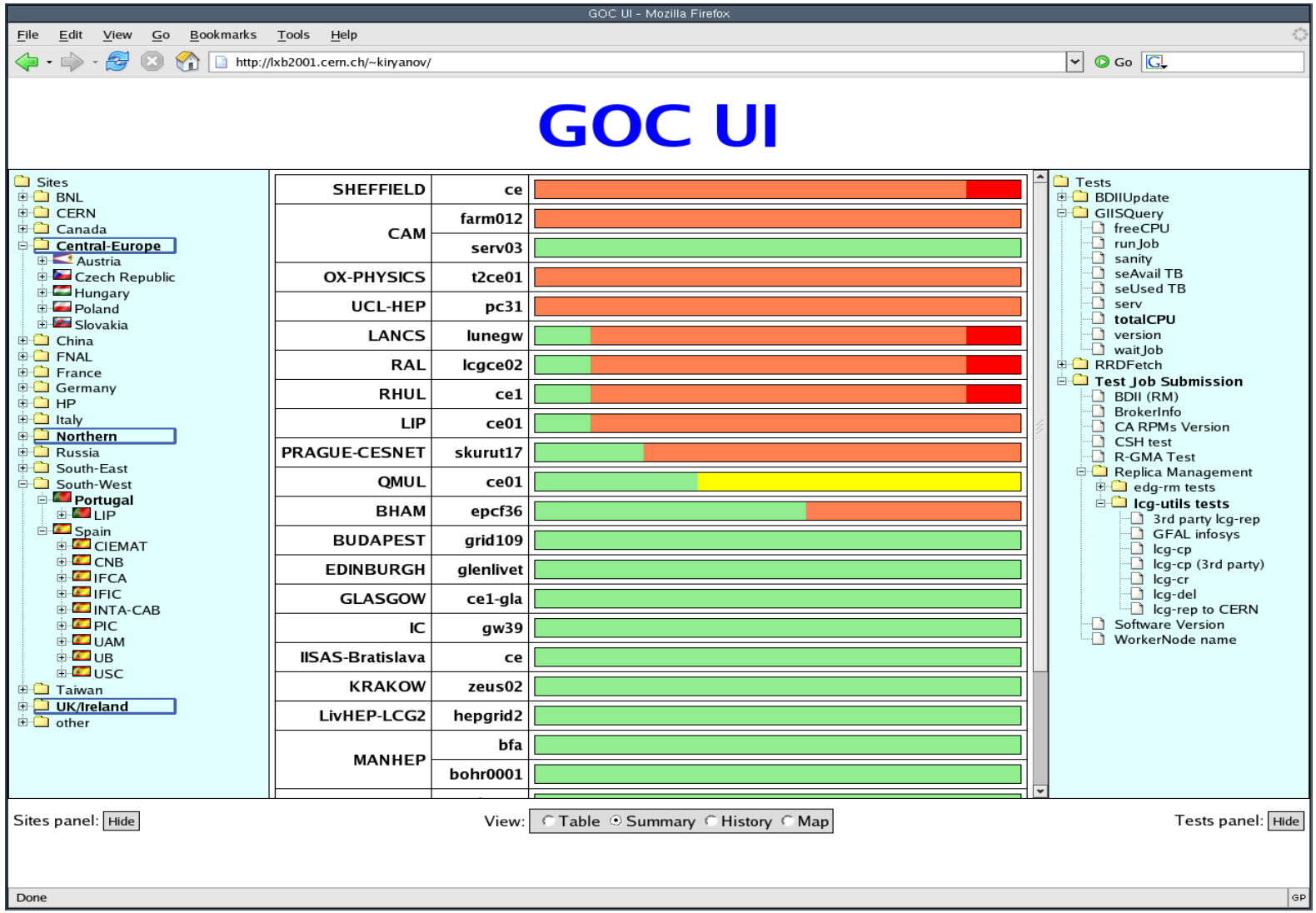
- Currently we use MySQL as a backend for Archivers
- A single set of archivers contains two archivers: Archiver+LatestProducer, Archiver+DBProducer
- Two sets of Archivers for fault tolerance: CERN, Taipei
- Data survives archiver failure/restart as it is stored physically in external MySQL DBMS
- Insert rate: ~20M tuples/month, measurements each 5 seconds

Reporting Tool Prototype



		totalCPU	GFAL infosys	Test Job Submission	lcg-cr	lcg-cp
SHEFFIELD	ce		n/a	LISTMATCHFAILED	n/a	n/a
CAM	farm012		n/a	FAILED	n/a	n/a
	serv03	18 CPU	—	—	—	—
OX-PHYSICS	t2ce01		n/a	FAILED	n/a	n/a
UCL-HEP	pc31		n/a	FAILED	n/a	n/a
LANCS	lunegw	22 CPU	n/a	LISTMATCHFAILED	n/a	n/a
RAL	lcgce02	928 CPU	n/a	LISTMATCHFAILED	n/a	n/a
RHUL	ce1	146 CPU	n/a	LISTMATCHFAILED	n/a	n/a
LIP	ce01	4 CPU	n/a	FAILED	n/a	n/a
PRAGUE-CESNET	skurut17	56 CPU	n/a	OK	n/a	n/a
QMUL	ce01	576 CPU	OK	OK	n/a	n/a
BHAM	epcf36	18 CPU	OK	OK	FAILED	FAILED
BUDAPEST	grid109	95 CPU	OK	OK	OK	OK
EDINBURGH	glenlivet	1 CPU	OK	OK	OK	OK
GLASGOW	ce1-gla	3 CPU	OK	OK	OK	OK
IC	gw39	60 CPU	OK	OK	OK	OK
IISAS-Bratislava	ce	4 CPU	OK	OK	OK	OK
KRAKOW	zeus02	16 CPU	OK	OK	OK	OK
LivHEP-LCG2	hepgrid2	113 CPU	OK	OK	OK	OK
MANHEP	bfa	—	OK	OK	OK	OK

Reporting Tool Prototype



Advantages

- No need for configuration of access points both for sensors and for reporting tools – data is located automatically by the registry
- No risk of data loss in case of failure of monitoring software – R-GMA is just for data transport, not for storage
- Flexibility – by using predicates, one can setup a number of archivers with different policies in different physical places, fault tolerance
- Usage of SQL and relational database model makes it all elegant

Known issues

- R-GMA Registry (and Schema) is a single point of failure – will be fixed in next release
- Supported subset of SQL sometimes not sufficient (aggregate functions, GROUP BY, etc.)
- Lack of schema modifications – once table is defined it can't be changed or removed
- Performance issues: eg. lack of DB indices – however this can be done manually using direct access to DBMS
- Support for other DBMS: Oracle?

References

- R-GMA Home Page: http://www.r-gma.org/
- EGEE-JRA1-UK Home Page: http://hepunx.rl.ac.uk/egee/jra1-uk/
- R-GMA based monitoring system for EGEE/LCG2 operations: http://goc.grid.sinica.edu.tw/gocwiki/RgmaUnifiedMonitoringSystem