GridICE

The eyes of the grid

PART I. Introduction to Grid Monitoring

Sergio Andreozzi

PART II. GridICE: architectural insight

Sergio Fantinel

PART III. GridICE: live demo

Gennaro Tortone

OUTLINE – PART I

- Grid Monitoring
  - Problem definition
  - Requirements for an ideal Grid Monitoring Service

- GridICE 1.0:
  - Architecture overview
  - Data flow: from resources to users

Defining Grid Monitoring Service

- **Grid Monitoring Service**:
  - the activity of **measuring** significant **grid resources related parameters**
  - in order to
    - **analyze usage, behavior and performance of the grid**
    - **detect and notify**
      - **fault situations**
      - **contract violations (SLA)**
      - **user-defined events**

Grid Monitoring: problem definition

- The Grid involves a huge number of worldwide distributed resources
- Monitoring of those distributed resources is a vital determinant for the whole system
- Different actors require different views of monitoring information:
  - **Virtual Organization managers** require the ability of observing and analyzing the performance of the "actual" system they are using (this can dynamically change over time)
  - Both **site administrators** and **grid operation center** managers require performance analysis and fault detection of the resources for which they are responsible
  - **Grid Service developers** require the ability of analyzing the behavior of their applications (e.g., how does a resource broker dispatch jobs over a set of available resources)

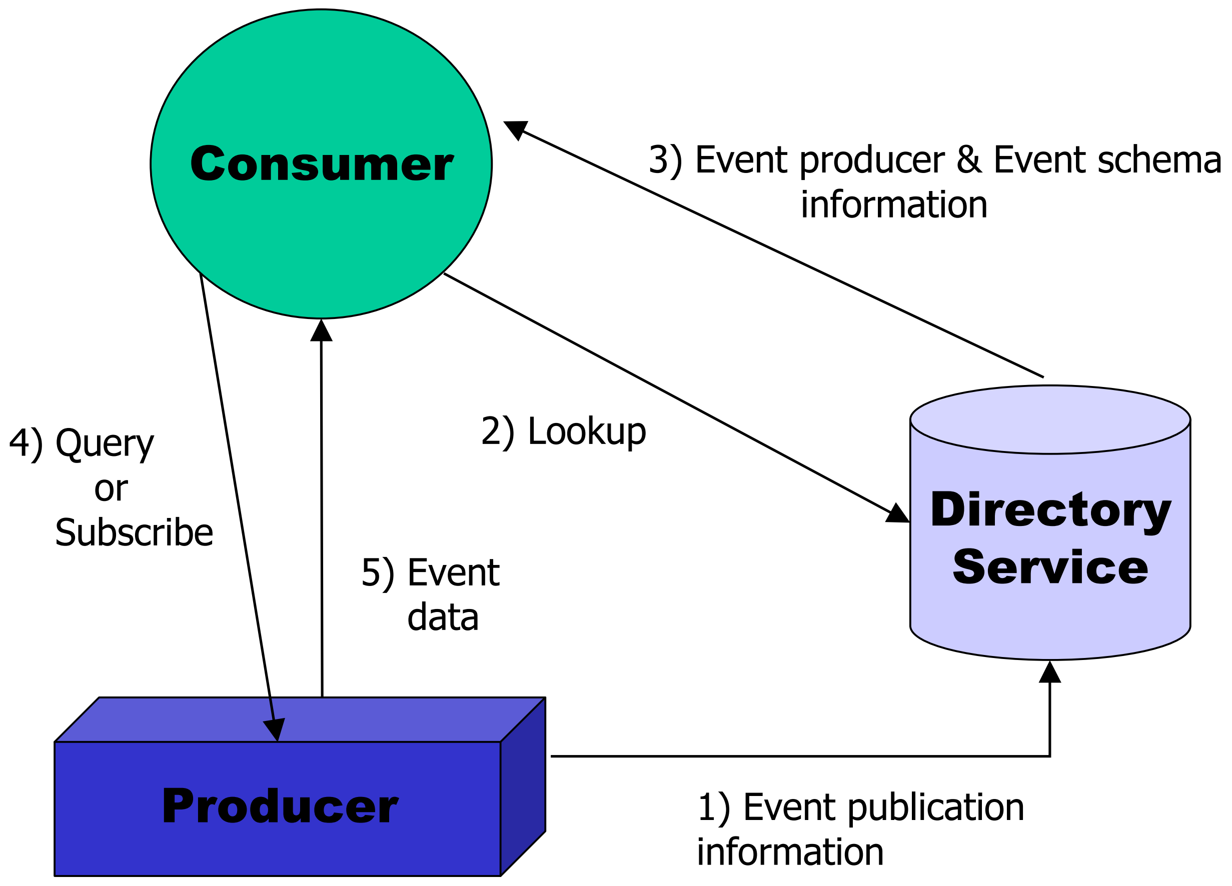
Requirements for an ideal Grid Monitoring Service

- An ideal Monitoring Service should:
  - flexibly scale with the number of available resources
  - be low intrusive for both resources and network usage
  - deal with time sensitive data
  - provide for efficient delivery of monitoring data
  - describe monitoring data in a standard format
  - allow topic-based monitoring data subscription
  - ensure data integrity
  - preserve the access control policies imposed by the ultimate owners of the data

Event-driven architecture

- Event-driven architecture (EDA) is an approach for designing and building applications in which events trigger messages to be sent between independent software modules that are completely unaware of each other
- An event source typically sends messages to middleware, and the middleware distributes the messages to the consumers that want to be notified of the events. Messages are typically sent only to those consumers that have subscribed to receive events
- The event itself is a complete description of an activity, such as the opening of a customer account or the clicking of a button
- **It is a suitable architecture for a grid monitoring service**

GGF Grid Monitoring Architecture proposal


Consumer
3) Event producer & Event schema information
2) Lookup
4) Query or Subscribe
5) Event data
Directory Service
Producer
1) Event publication information

GridICE 1.0 architectural overview

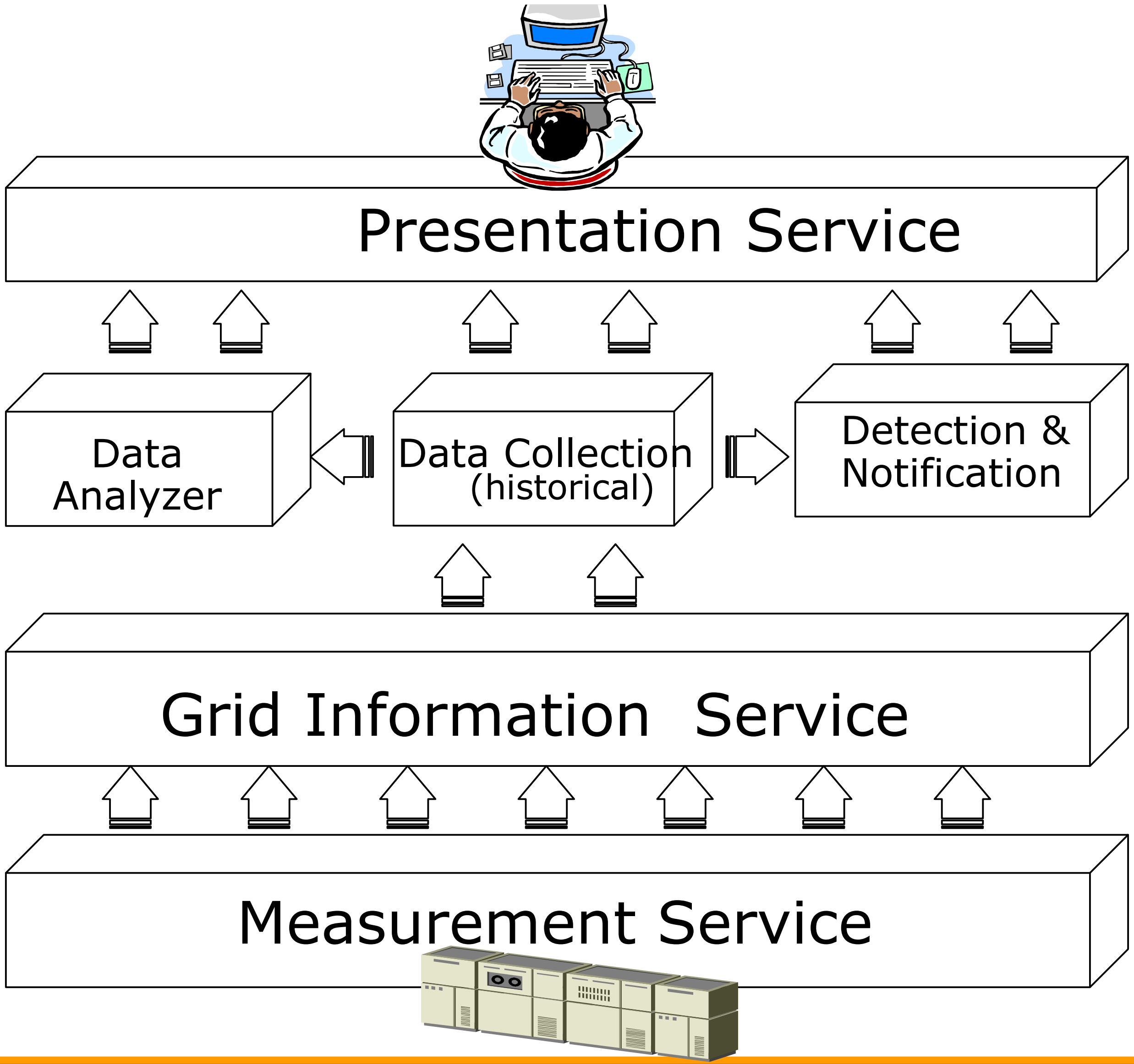
GridICE

- Grid Monitoring Service developed by INFN as part of the EU DataTAG project (WP4)
- Started on January 2003
- First release (short term) to be deployed asap in the current HEP Grid middlewares (EDG 1.4/2.0, LCG 0/LCG 1)
- Second release (medium term) to rely on a distributed event-based architecture paradigm

Components for a Grid Monitoring Service

- **Measurement Service**:
  - ➢ service able to probe the resources for certain parameters (especially QoS related)
- **Discovery Service**:
  - ➢ service able to find out which resources are currently available
  - ➢ rely on Grid Information Service
- **Detection and Notification Service**:
  - ➢ Fault situations, SLA violations, user-defined events
- **Data Analyzer**:
  - ➢ Performance, Usage, general reports and statistics
- **Presentation Service**:
  - ➢ web-based graphic user interface
  - ➢ role-based view

Data flow from resources to user



Measurement Service

service able to probe the resources for certain parameters

- **Parameters**:
  - Based on Glue Schema (vers. 1.1)
  - Richer host related parameter set
  - soon:
    - Glue Network Service, Job Details Monitoring, Collective Services
- **Collecting observations:**
  - Worker node related:
    - Rely on EDG WP4 fmon (customized) in order to collect worker nodes params at cluster head node
  - Injecting params in the GIS:
    - Standard EDG4Glue + extensions for worker nodes info

Discovery Service

service able to find out which resources are currently available:

- rely on available Grid Information Service in order to be able to exploit resource automatic discovery
- at the moment, MDS 2.x is supported
- Mixed set of GIS can fit into the architecture

Detection and Notification Service

Rely on the following Nagios functionalities:

- Activity Scheduler
- Event Notification

- At the moment a pre-defined set of events is checked for notification
- Dynamic event configuration is foreseen as a low priority development task

Presentation Service

- Web-based graphic user interface

- Role-based view:
  - ➢ VO-manager
    - Resources available to the VO
    - Total running jobs owned by users part of the VO
  - ➢ Site manager and Grid Operation Center Manager
    - Status of local resources
  - ➢ User (work in progress)
    - Total Accessible Free Processor
    - Requires interaction with organization-based authorization services (e.g. VOMS)

Next presentations

- Sergio Fantinel
  - GridICE 1.0: Architectural insight
- Gennaro Tortone
  - GridICE 1.0: Live demo