Middleware Service and Its Future

Ian Bird
IT Department, CERN
LCG Deployment Area Manager and EGEE Operations Manager

LCG-Asia Workshop, Taipei
26th July 2004
Overview

- Brief overview of what is in LCG-2
- What is the strategy to evolve this?
- Overview of gLite and what is expected
- The services that will be put in place
- Operations requirements on middleware services
Where are we with LCG-2?

Status of existing middleware
- Based on existing components

Application level services
- User interfaces
- Applications

“Higher level” services
- Information system
- Resource Broker
- Data management

“Basic” services
- User access
- Information system
- Information schema
- Data transfer
- Security

System software
- Operating system
- File system
- Local scheduler

Hardware
- Computing cluster
- Network resources
- Data storage

Closed system (?)
- RedHat Linux
- VDT (Globus, GLUE)
- PBS, Condor, LSF, ...

EDG
- LCG, experiments

EU DataGrid

VDT

LCG-Asia Workshop – 26 July 2004 - 3
Many problems seen with the existing middleware:

- Functional and performance issues
  - RLS, RB
- Underlying models and architecture:
  - Resource advertising and matching
  - Fixed information schema
- Lack of consistent interfaces
- System level issues not addressed
  - Exception handling, failover, single points of failure, management interfaces, etc.
- ...

Status of LCG-2
Background – LCG service evolution

- **LCG-2**
  - Current base for *production* services
  - Evolved with certified new or improved services from the preproduction

- **Pre-production Service**
  - Early application access for new developments
  - Certification of selected components from gLite
  - Starts with LCG-2

- **Migrate to new mware in 2005**
  - Organising smooth/gradual transition from LCG-2 to GLite for production operations
Parallel strategy

- **LCG-2**
  - Existing production infrastructure
  - Continue (realistic) enhancements and developments
  - Aim at creating baseline solution for:
    - Basic data handling, batch job production work

- **gLite (LCG-n)**
  - Re-engineered services – based on experience gained with LCG-2
  - Hardened – robust, manageable services
  - Functionality aimed at batch production and analysis work

- **Convergence?**
  - gLite has to demonstrate superiority over LCG-2
    - On the pre-production service, some service deployed in production
  - Supersede LCG-2
What’s next? → gLite from EGEE
Guiding Principles

➤ Lightweight (existing) services
  ▪ Easily and quickly deployable

➤ Interoperability
  ▪ Allow for multiple implementations

➤ Resilience and Fault Tolerance

➤ Co-existence with deployed infrastructure
  ▪ Run as an application (e.g. on LCG-2; Grid3)
  ▪ Reduce requirements on site components
    • Basically globus and SRM
  ▪ Co-existence (and convergence) with LCG-2 and Grid3 are essential for the EGEE Grid service

➤ Service oriented approach
  ▪ Follow WSRF standardization
  ▪ No mature WSRF implementations exist to date, hence: start with plain WS – WSRF compliance is not an immediate goal
Approach

- Exploit experience and components from existing projects
  - AliEn, VDT, EDG, LCG
- Design team works out architecture and design
  - Architecture: [https://edms.cern.ch/document/476451](https://edms.cern.ch/document/476451)
  - Feedback and guidance from EGEE PTF, LCG GAG, LCG Operations
- Components are initially deployed on a prototype infrastructure
  - Small scale (CERN & Univ. Wisconsin)
  - Get user feedback on service semantics and interfaces
- After internal integration and testing components are delivered to SA1 and deployed on the pre-production service
gLite Services

Access Services
- Grid Access Service
- API

Security Services
- Authorization
- Authentication
- Auditing

Information & Monitoring Services
- Information & Monitoring
- Job Monitoring

Data Services
- Metadata Catalog
- File & Replica Catalog
- Data Management
- Storage Element

Job Management Services
- Accounting
- Site Proxy
- Job Provenance
- Computing Element
- Package Manager
- Workload Management
- Works in push and pull mode
- Site policy enforcement
- Exploit Condor-G and globus gatekeeper
WMS

- Exploit pull model –
  - Better requirements matching
  - Reduced role of IS
- Also works in push mode
Computing Element

- Layered service interfacing
  - various batch systems (LSF, PBS, Condor)
  - Grid systems like GT2, GT3, and Unicore
- CondorG as queuing system on the CE
  - Allows CE to be used in push and pull mode
- Call-out module to change job ownership (security)
- Lightweight service
  - should be possible to dynamically install e.g. within an existing globus gatekeeper
Data Management

- Scheduled data transfers (like jobs)
- SRM based storage
- Reliable file transfer
Catalogs

- File Catalog
  - Filesystem-like view on logical file names

- Replica Catalog
  - Keep track of replicas of the same file

- (Meta Data Catalog)
  - Attributes of files on the logical level
  - Boundary between generic middleware and application layer
Storage Element

‘Strategic’ SE
- High QoS: reliable, safe..
- Usually has an MSS back-end
- Place to keep important data
- Needs people to keep running
- Heavyweight

‘Tactical’ SE
- Volatile, ‘lightweight’ space
- Enables sites to participate in an opportunistic manner
- Lower QoS
Storage Element Interfaces

- **SRM interface**
  - Management and control
  - SRM (with possible evolution)

- **Posix-like File I/O**
  - File Access
  - Open, read, write
  - Not real posix (like rfio)

**Management**

**User**

- POSIX API File I/O
- SRM interface
- rfio
- dcap
- chirp
- aio
- Castor
- dCache
- NeST
- Disk
PEP and PDP will in most cases be part of the secured service, rather than separated services.
How will we proceed?

- **Service challenges on LCG-2**
  - Set baselines and measurements

- **Production (batch focused) service**
  - Will continue to be based on LCG-2 – currently 64 sites
  - Continue to make (reasonable) enhancements and fixes

- **Pre-production service**
  - Now being built
    - within framework of EGEE, but will welcome other sites
  - Starts with LCG-2
  - Expect to deploy gLite components/services as they become available
    - Via our full certification process!!
  - Once validated on pre-production (applications, deployers, operators) migrate into production service, either
    - To replace existing components, or
    - In parallel with existing services
  - Replacement services must bring migration strategies
  - Management functionality as important as application functionality

- **Convergence as gLite (or external) components can replace existing components**
  - There will of course still be major changes that cannot be simple replacements
Certification, Testing and Release Cycle

DEVELOPMENT & INTEGRATION
UNIT & FUNCTIONAL TESTING

CERTIFICATION TESTING
Integrate
Basic Functionality Tests
Run Certification Matrix
Run tests C&T suites Site suites

APP INTEGR
HEP EXPTS
BIO-MED
OTHER TBD

DEPLOY
DEPLOYMENT PREPARATION
DEPLOYMENT RELEASE
Certified release tag
Deployment release tag

PRE-PRODUCTION
Production tag

PRODUCTION

LCG-Asia Workshop – 26 July 2004 - 20
Operational issues

- Portability of code
  - Must support wide variety of platforms – avoid existing limitations

- Operational aspects need more attention
  - Phasing in new services and phasing out old in a transparent way
    - clear state model for services has to be defined that allows this
  - Moving state:
    - state should be transportable (services have to be reborn)
      - A service stopped on node A should be restart able on node B by moving the state between the nodes
  - ONE set of APIs for administrative and monitoring interfaces
    - with authorization to allow remote operations GOCs
  - Unique logging format for all services
    - needed for debugging and audit trails
  - Accounting interface
    - currently software filters multiple log files and joins them (not ideal)
  - Software has to be able to reflect resource usage policies as complex as the sites have implemented them locally
    - Time dependent policies (use only at night…)
Operational requirements – 2

More things to consider

- **Scalability**
  - analysis and testing of scalability of services needed

- **Single point of failures have to be avoided**
  - for each service a short risk analysis is needed:
    - the effect on the overall system when:
      - the service is not accessible for different duration
      - the underlying hardware dies
      - the service is overloaded
  - for services that are essential it has to be clear how the service can be deployed in a redundant way
    - effect on scalability, performance,

- **Exception handling**
  - services have to be prepared to handle non standard situations
    - confronted with old protocols/ corrupted data
    - resource exhaustion (memory, CPU)
  - currently services sometimes just crash or get blocked:

Security – must not be an afterthought

- **Authentication, authorization, auditing, etc must be part of the design and must not degrade functionality or usability**
Adding and removing VOs has to be a light weight operation for a site

- Currently many resources/services are done per VO (paths, services, etc.)
  - hard to manage
  - error prone to add new VO (needs modifications in many places)
- VO based policies needed for resource utilization
- Adding a VO and defining the policy should be done at one place for a site and not for each service/resource independently
Summary

- Existing LCG-2-based production service will remain
- Parallel pre-production service will
  - Deploy LCG-2 and components from gLite and other sources
  - Be used to validate new components, which can then be move into production
- Validation will be done by:
  - Applications (ARDA, biomed, etc), System managers and operators, security groups
  - Will not be based only on functionality
  - Data challenges, service challenges, continuous usage
- As far as possible aim for a smooth evolution from LCG-2 to a better service