Implementation of common technologies in Grid middlewares

or roadmap towards unified stack

Oxana Smirnova, for
B. Kónya L. Field, J. White, P. Fuhrmann, M. Cecchi,
J.K. Nilsen and the rest of EMI project

Grid 2012, 17 July 2012, Dubna
Context: EMI project

- European FP7 project
- 24 partners, coordinator: CERN
- 3 years, ends in April 2013
- 1115 person-months
- 12 MEUR funding from EU, the rest – from partners

EMI brings together the major EU middleware developers
EMI: 3 years of harmony

Before EMI

3 years

After EMI

Implement agreements

Common products

Standard repositories

Standards, common new technologies, users’ and Infrastructures’ requirements

17/7/2012 www.eu-emi.eu
EMI releases

Support period for a release:
12 months full support
+6 months updates
+6 months security updates only
EMI ecosystem
Unifying standards and agreements

APIs, libs, probes, sensors, utils
- Nagios probes
- APEL SSM
- CANL lib
- Compute lib
- EMI_datalib
- ...

Security
- VOMS
- gLEexec-WN
- Argus
- ...

Computing
- UNICORE/X
- ARC CE
- CREAM CE
- L&L
- WNoDeS
- WMS
- ...

Data
- dCache
- StoRM
- DPM
- LFC
- FTS3
- ...

Infrastructure
- ERIS
- EMIR
- ...

17/7/2012 www.eu-emi.eu
Overview of key common technologies

Major new features and services

- **EMI-ES**: common interface for job management
  - All EMI CEs support: Torque, SGE, LSF, SLURM
- **GLUE2**: information schema published by all the EMI services
- **EMIR**: a common registry for all EMI services
- **CANL**: a prototype of the common security library
- **ARGUS**: the EMI authorization solution
- **Nagios** probes exist for every EMI service
- **HTTPS** and **WebDAV** support in storage
- ...

Technical Agreements and Standardization

- **StAR**: Storage Accounting record
- **CAR**: Compute Accounting record
- **EMI Delegation Agreement**
- A common EMI **SAML profile**
- ...

17/7/2012
Industry standard protocols for accessing SEs and the catalog

- DPM and dCache ready for **NFS4.1**
- **HTTPS** offered by DPM, StoRM and dCache
- **WebDAV** support:
  - *in DPM and dCache*
  - *being developed in FTS3 and LFC*
- See more on dCache in Patrick Fuhrmann’s talk tomorrow
Show-case 2: EMI service Registry

EMIR: from implementation through performance testing to deployment planning

- EMIR is one central registry where all services can be discovered
  - No common registry before EMI
- Unified service discovery
- Quorum based, replicated Global Service Registry DB ensures high availability
- Service providers register to and push information to Domain Service Registries (DSR)
- Deployment hierarchy of DSRs is in investigation
- All services will publish information in EMIR
Show-case 3: EMI Execution Service

EMI-ES: from specification to implementations

- EMI-ES specification agreed as the common job management interface
- Web-service interface with:
  - Integrated support for data staging
  - Delegation capability
  - Re-engineered state model
  - Revised job description
  - GLUE2-based service and activity description
    - Clearly defined Port Types
- Implementation: at 80-90% completion
- Iterative schema definition
  - EMI-ES v1.2 revision in process
- Client side development ongoing
- Key EMI harmonisation effort
Show-case 4: Common Authentication Library

**CANL: from implementation to adoption**

- Defined and documented API for common security library
- Main features:
  - *Credentials handling*
  - *Trust store handling*
  - *Name constraints checking*
  - *CRL, OCSP support (on-line revocation)*
  - *Proxy: verification, generation, proxy CSRs, utilities*
  - *Partially unified error codes and messages*
  - *PKCS 11 support*
- Implemented in C, C++ and Java
- EMI products migrating to CANL:
  - VOMS, ARC HED, Trustmanager, L&B, UNICORE, dCache, CREAM, Argus, Pseudonymity, Hydra, STS
Show-case 5: CAR & StAR

- EMI agreement on accounting records
  - Compute accounting **CAR**
  - Storage accounting **StAR**
- To be used "everywhere", e.g.
  - accounting sensors
  - APEL
- Feeding all this back to OGF
  - UR 2.0 (?)
### Show-case 5: CAR & StAR

#### CAR:
- Definition of usage record for computing
- Mainly a profile of the OGF UR1.0
- Some differences:
  - *Added support for groups/VOs*
    - Using Group+GroupAttribute like in StAR
  - Definition encompasses both usage records and aggregated records
  - Aggregated schema follows APEL SSM guidelines

#### StAR:
- Definition of usage record for storage
- Inspired by OGF UR1.0
- Summarises used space
- XML-based schema
- Non-overlapping/non-contiguous records means no space used
  - *ValidDuration configured by sys-admin*
- Supports handling of groups (aka VOs)
- Scope is limited to consumption of storage space
The last peak (Year 3 development plans)

General strategy:

- Complete product developments:
  - FTS3, GFAL2
  - STS
  - EMI Datalib
- Product hardening, focus on usability
- Integration and adoption of common EMI solutions (EMIR, CANL)
- Migration plans, compatibility
In the pipeline: STS

- Getting Grid certificates is the largest hurdle
- Users often have other credentials
  - University ID (library, Eduroam etc)
  - Lab ID (like NICE ate CERN)
  - Bank ID
  - etc
- Security Token Service (STS) can help
  - For simplified credential management
  - STS transforms an existing security token into another security token
  - SOAP-based Web Service

forge.switch.ch/redmine/projects/sts/wiki
In the pipeline: emi_datalib

Before EMI: different approaches to transfer handling

- **gLite approach:**
  - posix-like data access
  - pluggable architecture
  - several external clients (e.g. experiments software) use GFAL and lcg_util libraries
  - written in C

- **ARC approach:**
  - file-based data-moving library – libarcdata2
  - pluggable architecture
  - libarcdata2 is used by the ARC command tools, the ARC CE and external clients to transfer files
  - written in C++
In the pipeline: emi_datalib

- **Common EMI Datalib**
  - posix-like interface from GFAL2
  - higher-level file-based interface from libarcdata2
  - adds transfer interface to GFAL2 for initiation and monitoring of 3rd-party transfers
  - libarcdata2 will use GFAL2 through plugin
  - external clients requiring posix-based byte-wise data access can use GFAL2 directly
  - file-based data moving clients (e.g. lcg_util CLI, ARC CLI and ARC CE) can use libarcdata2
  - FTS3 and parts of the lcg_utils and ARC CLIs will use the 3rd-party transfer interface of GFAL2
  - python library will be created to the needed functionality from the lcg_util python API
  - all plugins are moved under GFAL2 though some plugins are file-based and not posix – allows posix access without needing to know physical file locations
Summary

• Three-year-long EMI project has given an excellent one-time opportunity to work together
• Most development-intensive phase of EMI is just behind us
• With EMI-2 Matterhorn EMI delivered several long-sought common solutions:
  – Common libraries (CANL)
  – Common service (EMIR)
  – Common implemented interfaces (EMI-ES over CEs)
  – Agreements and adoption plans (CAR-STAR, ...)
• The last year will still bring some interesting development, but the focus is now on product hardening
  – Planned for EMI 3: STS, FTS3, GFAL, emi_datalib