General Introduction to UNICORE 6

Bernd Schuller and the UNICORE team
Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH
July 16, 2012
XSEDE12 Chicago
Outline

- UNICORE overview
- Data management and job model
- Workflows
- Applications in UNICORE
- Clients
Integrated, complete Grid middleware stack

- Easy to install, configure, administrate and monitor
- Excellent application and workflow support
- Suited for both high performance and high-throughput usage
- Wide variety of clients: GUI, commandline, APIs
- Java/Perl based, supports common operating and resource management systems
  - OS: UNIXes, MacOS X, Windows. RMS: no-batch, Torque, LoadLeveller, LSF, SGE, ...
- Active developers, responsive to user wishes ;-) , quick and efficient support
- Open source, BSD licensed, visit http://www.unicore.eu
Basic Architecture

Client Layer
- Graphical Client
- Command Line Client

Web Service Layer
- UNICORE Atomic Services
- OGSA-*

Execution Management Layer
- XNJS - Backend

Target System Interface
- Local RMS (e.g. Torque, LL, LSF, etc.)

Resource Layer
- Gateway
- AuthZ Attributes
- AuthZ Policies
- IDB (Resource description)
Clients
Building, submitting and monitoring jobs and workflows

Integrated data and storage management

Pluggable credentials

“Simple view” for novice users

Based on the Eclipse framework

Extensibility through plug-ins

Examples: Shibboleth, CIS view, Admin Interfaces (UVOS, XUUDB, dynamically deploy/undeploy services), etc.

Installation/update mechanism for plug-ins and GridBeans
UNICORE 6 commandline client (UCC)

Provides access to all the UNICORE 6 functionality from the commandline, plus

- Batch mode for high-throughput job processing
- Scriptable, easily extensible
- Shell autocomplete, Emacs integration
Services
Basic services

- Job submission and management
  - Single jobs (stage-in, execution, stage-out)
- Storage system access
  - File/directory management (mkdir, rm, ls etc)
  - Access to job working directories
  - Access to admin-defined storages (HOME, TMP, ...)
- File transfer
  - Import/Export to local machine
  - Server-to-server (scheduled, managed)
- Metadata management (optional)
- Service registry (per container or shared)
Storage support

- UNICORE storage management service ("SMS") provides an abstract filesystem-like view on a storage resource

- Common operations
  - mkdir, delete, listDirectory, etc

- Initiate filetransfers
  - Import/export files from/to the client
  - Send/receive files from/to other servers

- Various storage types (backends) available
  - Filesystem, Apache Hadoop, iRods (prototype!)

- Designed for extensibility (e.g. Amazon S3, …)
UNICORE 6

Storages and data movement

Storage instances
(HOME, TMP, ...)

Job directories

UNICORE Server

Server-to-server data movement

Local Filespace

Client

Client Server Import & Export

Storage instances (HOME, ...) Data staging

UNICORE Server

Non-UNICORE

UNICORE Server
File transfer

- Both client-to-server and server-to-server FT available
- Builtin: BFT transfer (based on HTTTPs)
  - Single open port needed, (almost) full UNICORE security
  - Simple interface (bulk write, read supports byte ranges), fast (several MB/sec.)
- Builtin: OGSA ByteIO (uses SOAP messages)
  - Single port, full UNICORE security
  - Rich interface (POSIX-like, block read/write, etc), slow (~400kB/sec)
- High-performance UFTP solution is available as a separate download
Additional options for data staging

- GridFTP
  - Uses existing globus-url-copy
  - Proxy generated on the client and sent with the job
- Plain HTTP and HTTPS
- FTP and SCP (including client credentials)
- „mailto“ for stage-out :-(
Integrated storage management in the UNICORE Rich client Grid browser

- Create files
- Drag and drop from/to desktop environment
- Copy and paste
- Remote file editing
Metadata management

- Decentralized approach: „metadata management service“ (MMS) associated with each storage service („SMS“)
- Schema-free: metadata is key-value pairs
- User can create, edit, delete metadata
- Metadata indexing
Metadata management: implementation

- Metadata storage directly as files on the storage
- Uses well-known open source libraries
- Indexer and search engine: Apache Lucene
- Metadata extraction framework: Apache Tika
- Client tools currently only available in UCC
  - Example: list file properties including metadata

```bash
schuller@zam994-t400:/$ ucc-vsgc ls -l u6://VSGC-2/Home/Documents/refcard-hadoop.pdf -m
{
  "Content-Type": "application/pdf",
  "Creation-Date": "2010-09-23T16:25:05Z",
  "Last-Modified": "2010-09-23T16:25:11Z",
  "created": "Thu Sep 23 18:25:05 CEST 2010",
  "creator": "Adobe InDesign CS5 (7.0)",
  "producer": "Adobe PDF Library 9.9",
  "resourceName": "/Documents/refcard-hadoop.pdf",
  "trapped": "False",
  "xmpTPg:NPages": "6"
}
```
Job model
Job description

JSDL 1.0 (OGF standard)

- What to execute
  - Application name / version (mapped to executable by UNICORE)
  - or: Executable path
  - Arguments
  - Environment variables
  - Optional stdin/stdout redirect

- Data staging specification
  - Into job directory from URL
  - From job directory to URL

- Resources requested (number of CPUs, etc)
**Job execution**

1. **submit**
   - 1.1 create
   - 1.2 return job address

3. **start**

4. **wait until done**
   - 2. import data
   - 5a. export data
   - 2. stage-in data

5b. **stage-out data**
   - 5b. stage-out data

**Client**
- Local Filespace

**Job**

**Target System Service**

**USpace**

**Remote Storage Spaces**
Client Plugins / GridBeans

- Application specific graphical interface
  - Generates job description from user input
  - Provides graphical user interface for input data
  - Provides graphical user interface for output data
- Consist of
  - Job description generation code
  - One or more user interface modules

Developer's Guide:
Workflow system
null
Workflow features

- Simple graphs (DAGs)
- Workflow variables
- Loops and control constructs
  - while, for-each, if-else
- Conditions
  - Exit code, file existence, file size, workflow variables
- Clients
  - UNICORE Rich client
  - Commandline client
Workflow: For-each loop

Iterate over files or variables

Data files can be local or remote
Workflow System summary

- Workflow engine
  - High-level process enactment
  - Pluggable, domain-specific workflow languages
- Service orchestrator
  - Resource brokering based on pluggable strategies
  - Low-level Grid job execution and monitoring
  - Multiple instances can be deployed for scalability
- Tracing service
  - Logs workflow events with timestamps
  - Useful for performance evaluations and reporting
Information systems
Aggregates service information from multiple UNICORE 6 sites

An Entry contains at minimum:

- Service address
- Service type
- Server DN

Content is held up-to-date

- Services register with local registry
- Local registries push info to global registry
- Entries in global registries expire
- ...if not re-published in time
Application Integration
Application integration

- Application software installed on target system
  - Add entry in UNICORE server configuration (IDB file)
  - Graphical Client
    - *Generic plugin*
    - *Application specific plugin*
- User's own executable
  - Upload (import) it to the target system
  - Use it in a shell script
Example: MOPAC

Molecular Orbital PACkage, a semi-empirical quantum chemistry program

Configuration file entry on the server side (IDB file)

```xml
<idb:IDBApplication>
  <idb:ApplicationName>CMOPAC</idb:ApplicationName>
  <idb:ApplicationVersion>7.0</idb:ApplicationVersion>
  <jsdl:POSIXApplication
    <jsdl:Executable>/usr/local/c9m-qsar/bin/mopac</jsdl:Executable>
    <jsdl:Argument>$INPUT</jsdl:Argument>
    <jsdl:Argument>$OUTPUT</jsdl:Argument>
    <jsdl:Argument>$ERRLOG</jsdl:Argument>
  </jsdl:POSIXApplication>
</idb:IDBApplication>
```
Example: MOPAC - 2

Use it through Client‘s generic plugin:
Example: MOPAC - 3

Specify input and output in the Generic plugin:
Example: MOPAC - 4

application specific plugin
Example: MOPAC - 5

Display results (application specific plugin)
Example: MOPAC - 6

Display results (application specific plugin)
Thank you!
Common Information Service

- Aggregates information from multiple UNICORE 6 sites
- Store data as GLUE 2
- Present site information to the user
- View information through
  - Web browser via Google Maps
  - UCC
  - URC view
- Publishes records as ATOM feeds