Unicore: A Workflow Portal

Dr. David Snelling
Fujitsu European Centre for Information Technology
Computer Science Seminar, University of Birmingham
7th February 2002
Acknowledgements

- **Unicore Projects: BMBF**
  - [www.Unicore.de](http://www.Unicore.de)
  - [www.Unicore.org](http://www.Unicore.org)

- **EuroGrid Project: EU**
  - [www.EuroGrid.org](http://www.EuroGrid.org)

- **Fujitsu**
  - Matching support and then some
  - ASP development

- **Pallas**
  - Many of the slides are from their archive.
Outline

- Unicore Overview
- Grid Application Example
  - Stochastic Option Pricing (American)
- Future Directions
  - The GRIP Project
  - Comparison to Globus
  - Interoperability with Globus
- Demonstration
Motivation

- **HPC computer platforms are widely available**
- **Network capacity is catching up**
- **(Many) users are still in the stone–age**
  - access via telnet, ssh
  - cumbersome authentication mechanisms (SecurID, skey, …)
  - use primitive batch systems (NQS, LL, …)
  - need to know the systems they’re working on
- **As a result**
  - users stick to “their” systems
  - new platforms spell trouble
  - loss of flexibility at the centres
Motivation (ctd.)

**Seamless access systems try to help**
- use the same access mechanism (browser, client, …)
- use the same authentication mechanisms (certificates, …)
- hide system details (compile/link, run jobs, …)
- provide user–level job model (task graph, …)
- allow job monitoring and steering

**Lots of projects worldwide**
- SDSC Hotpage
- AHPCRC TeraWeb
- Mississippi State DMEFS
- UNICORE
- Globus (although it’s a toolkit not an environment …)
**Issues in Access Systems**

- **Intended users**
  - end–users in a specific domain
  - general HPC end–users
  - application developers

- **Supported use modes**
  - just batch execution
  - add interactive steering
  - general purpose interactive

- **How transparent do you want to be?**
  - just do the access and monitoring part
  - add a uniform job model
  - add a uniform resource model
  - add a uniform data model
  - add automatic scheduling amongst resources
Issues in Access Systems (ctd.)

- **System architecture**
  - one entry point to a collection of resources
  - multiple entry points, one per resource

- **Security mechanisms**
  - authentication
  - authorization
  - data integrity and confidentiality

- **Implementation technique**
  - Buzzword Prone (IP–based, at the end of the day)

- **System deployment**
  - integrate into existing system administration procedures
  - provide accounting and billing (industrial environment)
  - get users to employ it (they are a conservative lot, you know)
**The UNICORE Approach**

- **Focus on Access to HPC Systems in Batch Mode**
- **Support Nomadic Users**
- **Integrate into Existing Computing Center Environments**
  - Total site autonomy
  - Security architecture and policy
  - Administrative procedures and policy
- **Support Data Transfer and Migration**
  - User WS ↔ HPC platform
  - User WS ↔ Archive systems
  - HPC platform ↔ HPC platform
  - HPC platform ↔ Archive systems
- **Emphasize Security Issues**
  - User authentication and authorization
  - Data integrity (and confidentiality)
The UNICORE Approach (ctd.)

- **Support Distributed Computing**
  - But not classical meta–computing

- **Abstract Job Modeling**
  - Jobs modeled as a hierarchical task graph
  - Abstractions of domain concepts, not system/site specific
  - Server *incarnates* into platform–specific commands and flags

- **Rely on Existing Technology**
  - Java for software components
  - SSL, https for communication

- **Publish Interfaces and Protocol**
  - UNICORE Forums acts as keeper

- **Make Reference Software Available**
  - License modeled on Sun’s Community Source License
  - “Open Source, but not Free Software”
UNICORE Usage Scenarios

- **UNICORE is useful for**
  - nomadic users
  - distributed and diverse computing resources
  - distributed jobs
  - standard applications (with special frontends)

- **UNICORE was not designed for**
  - application developers
  - interactive jobs (yet)
  - It depends on your definition of interactive
Nomadic Users

- **Use the same access and authentication from anywhere**
  - submit computational jobs
  - monitor and control submitted jobs

- **Transfer data from/to the local workstation/laptop**
  - upload input data
  - download result data

- **Access to permanent data storage**
  - Unix file systems on target sites
  - archive systems (soon)

- **Helpful features**
  - checking of resource requests
  - file selectors for remote files
Distributed & Diverse Resources

- **Use the same methods to access any platform**
  - submit computational jobs
  - monitor and control submitted jobs

- **Transfer data from/to any platform**
  - Unix file systems on target sites
  - archive systems (soon)

- **Easy migration between platforms**
  - just change destination system (and maybe resource requests)

- **Helpful features**
  - checking of resource requests
  - no need to remember different accounts …
Workflow Support

- If Then Else Constructs
  - Based of task status or return code

- Repeat Until and Do For Constructs
  - Tail recursion really

- Job Hold
  - Probing results for application steering

- File Status Checks
  - Existence, attributes, ownership, modification date

- Planned
  - Event model and API.
Unicore Architecture

- Agents
- Interfaces
- Key Technologies
- Candidate Missing Technologies
**Interfaces**

- **Client Plugins**
  - Application specific support (CPMD, Gaussian, ...)

- **Unicore Protocol Layer**
  - Resource discovery, job submission, job control

- **Incarnation Data Base and TSI Interface**
  - Incarnation support

- **Unicore User Data Base**
  - User mapping and authorization

- **Broker Interface**
  - Interface defined and implemented in NJS.

- **Independent File Transfer Interface**
  - Interface defined
Protocol Layering

- Abstract Job Object
- Java Zip Stream (File Transfer Only)
- Unicore Protocol Layer
- SSL
- IP

<table>
<thead>
<tr>
<th>AJO</th>
<th>Java Zip Stream</th>
<th>Structure, Integrity, &amp; Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unicore Protocol Layer</td>
<td>Authorization, Packetizing, &amp; Control</td>
</tr>
<tr>
<td></td>
<td>SSL</td>
<td>Authentication, Integrity, &amp; Encryption</td>
</tr>
<tr>
<td></td>
<td>IP</td>
<td>Underlined functions can be turned off.</td>
</tr>
</tbody>
</table>
Key Technologies

- Seamless Computing Model
  - Job abstraction
  - Incarnation
  - File staging and transfer support

- Security Model
  - X509 certificate based security model
  - Consigner plus endorser model
  - Several CA policies
  - Fully flexible firewall support

- Generic Client
  - No changes to application needed

- Portable Server
  - Java + Perl
Missing Technologies

**Missing Models**
- Distributed directory services
- Multi-language protocol (C++ version of UPL)
- Application level message passing
- Event management
- Synchronized job start

**Missing Implementations (Expected Soon)**
- Resource reservation
- Interactive processing
- File streaming
- Meta-data support
- Dynamic support for new sites on the fly.
Option Pricing Example

- American Options
  - Multiple exercise dates

- Value Based on Statistical Model
  - Correlation between assets in the option
  - Monte Carlo simulation of a branching tree

- Quality of Random Numbers Critical
  - Low discrepancy sequences

- Computational Factors
  - Computationally intensive
  - Fast response important
An Option Pricing ASP

Multiple Locations
- Sales point
- Compute server
- Overflow compute server
- Confidential database

Demonstration
- At the end of the talk
Future Work: GRI P

Grid Interoperability Project
- The usual suspects from Europe
  - FZJ, Pallas, DWD, *fecit*
  - Universities: Manchester, Southampton, Warsaw
- Argonne National Labs

Goals
- Standards input from Europe
- Inter-grid applications
- Interoperability interfaces
Parallels and Contrasts

Resource Description
- Unicore: Same model for discovery and request
- Globus: Different models for discovery and request

Jobs vs Applications
- Unicore: Workflow environment
- Globus: Application APIs and toolkit

Security
- Unicore: End-to-End security model
- Globus: Requires transitive trust

Incarnation and Grounding
- Unicore: Incarnation of abstractions at server
- Globus: Client side grounding (substitution to ‘normal form’)?

Protocol Complexities
- Unicore: Polling
- Globus: Call Back

Dynamic versus Static?
What to Do?

- Run a Globus Job on Unicore Resources?
- Run a Unicore Job on Globus Resources?
- Political
  - More Globus Sites
  - Globus at some Unicore sites
- Security
  - Unicore is more strict
- Workflow
  - Globus has a simple “job chain” model

Unicore@Globus
Unicore on Globus Resources

- **Workflow Portal for Globus**
- **Grid_Proxy_Init on the Client**
  - Put the Globus Proxy Certificate in the SSO
- **Globus Resources**
  - Mapping LDAP → Unicore resources
  - Mapping Unicore resources → RSL
- **Mapping Unicore Tasks → RSL**
  - Execute application task only (not user or script tasks)
  - USpace in the user's home directory on the Globus system
  - Transfer between USpaces is via GridFTP
  - Import/Export for Home, Absolute-Home, Spool@Home
Run Unicore Tasks in Globus

- **Use IDB and TSI Existing Mechanisms**
  - Java implementation of TSI to ease development (not Perl)

- **Globus TSI is Polled by the NJS**
  - Mimic NQS (globus_qsub, globus_qstat, globus_kill...)
  - Globus then manages outstanding jobs submitted, like NQS

- **Logging**
  - Limited to what Globus sends back to TSI

- **Support Globus (permanent) Certificates**
  - Note that Globus temporary certificate cannot be used as Consigner or Endorser
A Prototype Globus USite

Globus USite

Client

Resource Mapper

FZJ
G_Vsite

UoM
G_Vsite

Checker
Brokers

IDB

G_TSI

GRIS

Transfer via GridFTP

Resource Mapper

GRIS

UoM

GRIS
Demonstration