

UNICORE

 Distributed
European
Infrastructure for
Supercomputing
Applications

Accessing (Super-)Computer Resources via Grid

Michael Rambadt
m.rambadt@fz-juelich.de



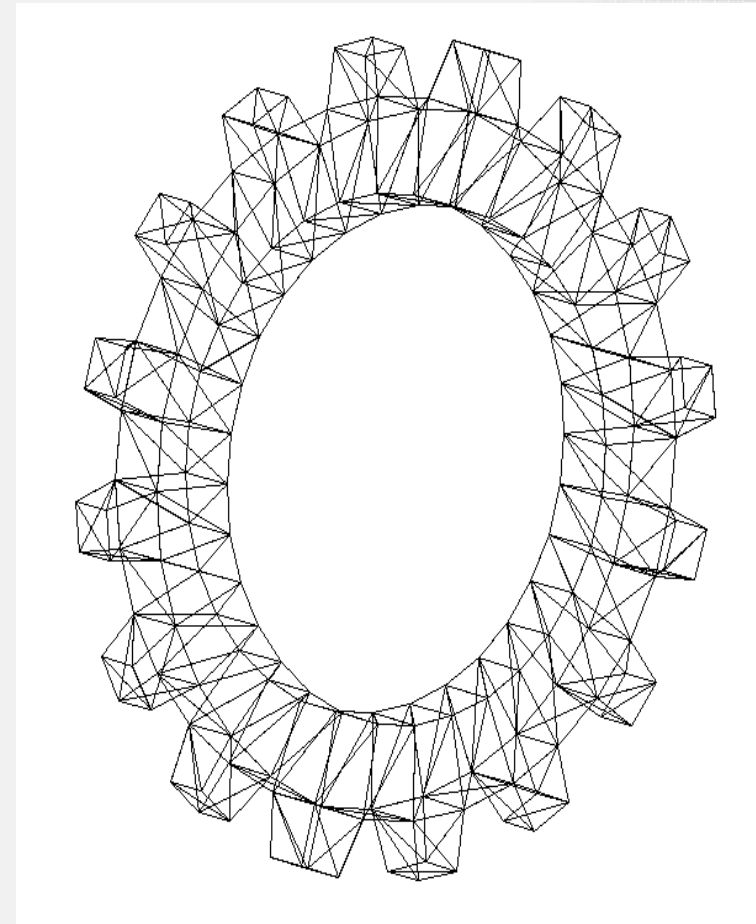
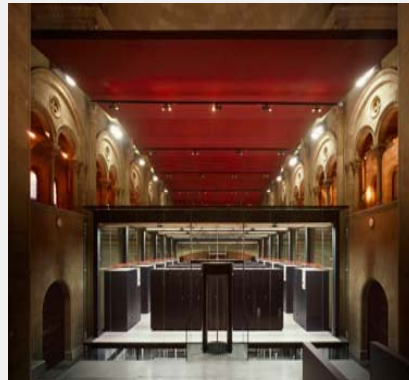
Table of content



- Motivation
- UNICORE Features
- UNICORE Architecture
- UNICORE Abstract Job Object (AJO)
- UNICORE Client
- UNICORE Security
- UNICORE deployment and further development
- UNICORE dissemination

Motivation: Why UNICORE ?

- Scientists need computational and storage related resources



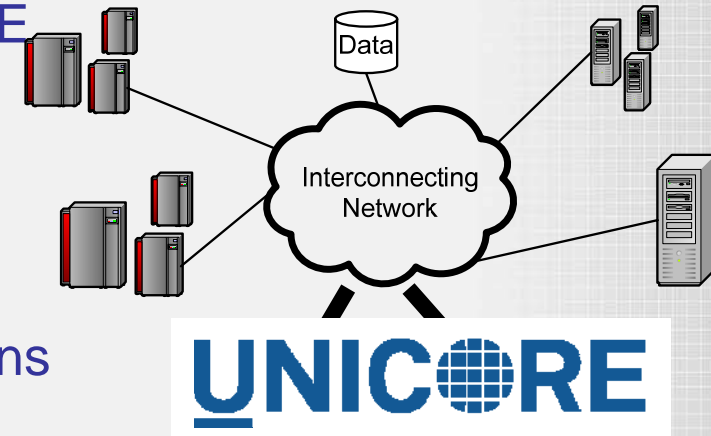
Motivation: Why UNICORE ?

- Supercomputers are managed by Resource Management Systems (RMSs) that handle the scheduling
- But: There are many RMSs available
- A proprietary way of job submitting
 - IBM Loadleveler → `llsubmit...`
 - Torque Resource Manager → `qsub...`
- Different job description languages (# of nodes, memory requirements...)



Motivation: Why **UNICORE** ?

- Solution: Grid Middleware **UNICORE**
- Define job workflows in abstract manner
- Immediate portability of job definitions for other systems with other architectures
- No 'learn overhead' if a new RMS is used
- Applications across multiple supercomputers/clusters → 'going meta'



What is UNICORE ?

- **UN**iform **I**nterface to **CO**mputing **RE**sources
- **Seamless**, **secure**, and **intuitive** access to distributed resources and data
- A **vertically** integrated Grid system used in production and projects worldwide
- is interoperable with other Grid software (e.g. GT2.4, CONDOR)
- is developed towards OGSA-based UNICORE/GS with WS-RF interoperability

UNICORE Features



- Intuitive GUI with single sign-on
- X.509 certificates for AA and job/data signing
- only one opened port in firewall required
- workflow engine for
 - complex multi-site multi-step workflows
 - job monitoring
- extensible application support
- secure data transfer integrated
- resource management
- easy installation and configuration of client and server components
- full control of resources remains at each site
- production quality, ...

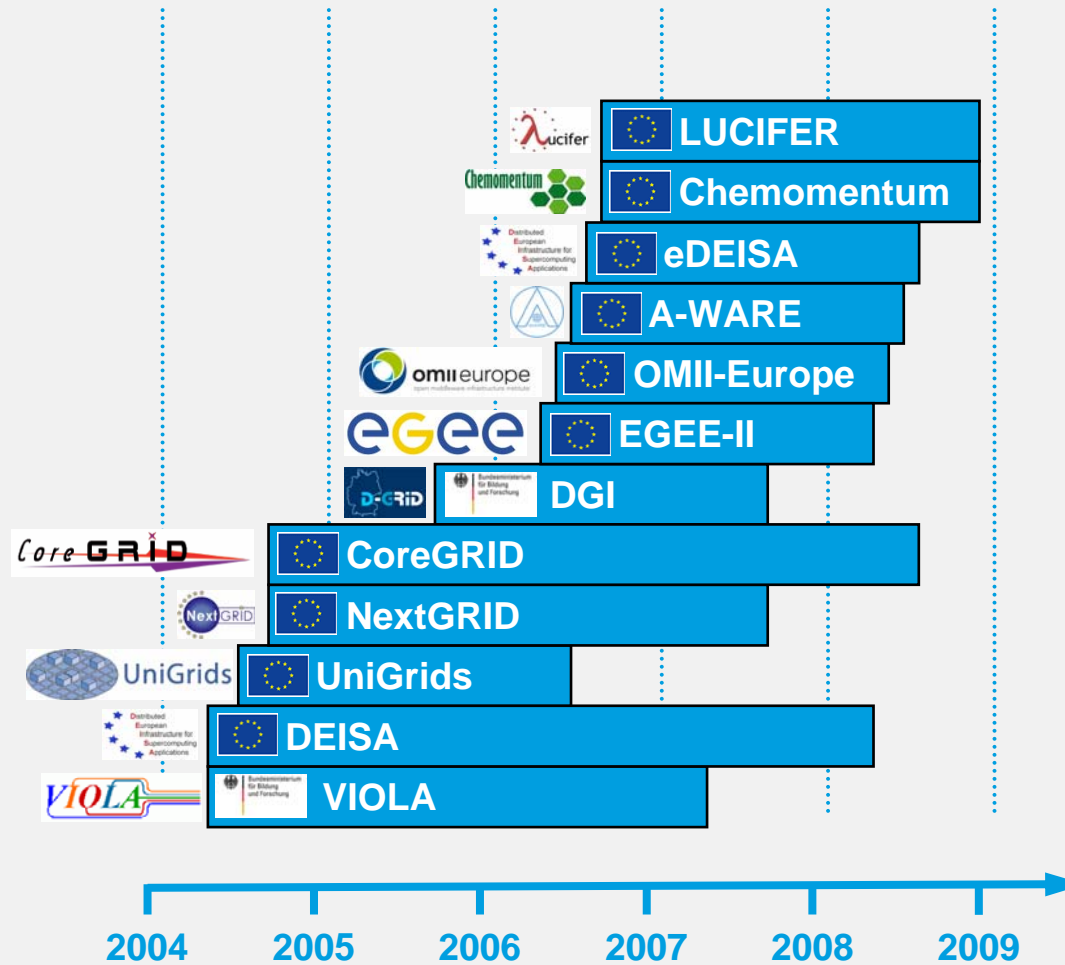
UNICORE development



- UNICORE 08/1997-12/1999
- UNICORE Plus 01/2000-12/2002
- EUROGRID 11/2000-01/2004
- GRIP 01/2002-02/2004
- OpenMolGRID 09/2002-02/2005



UNICORE development

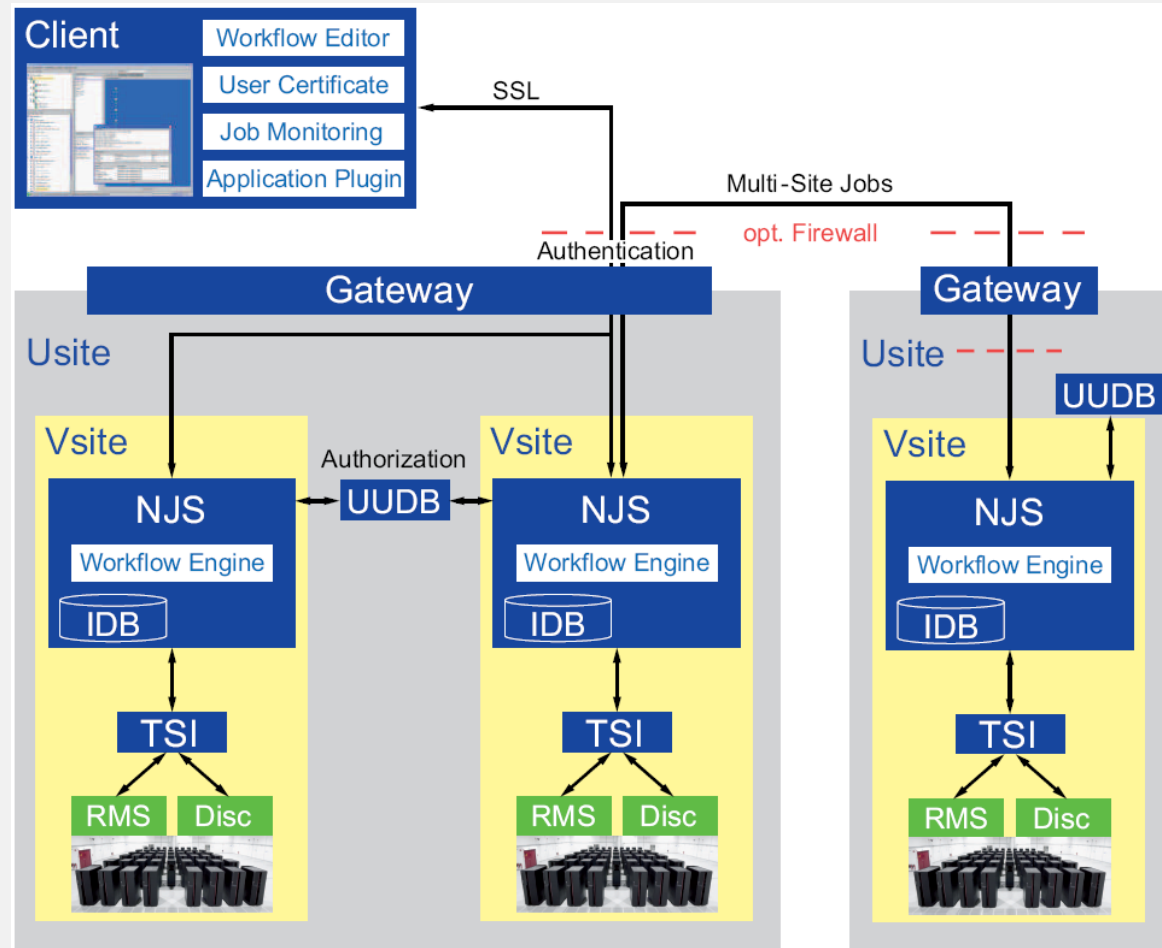


UNICORE Software Status

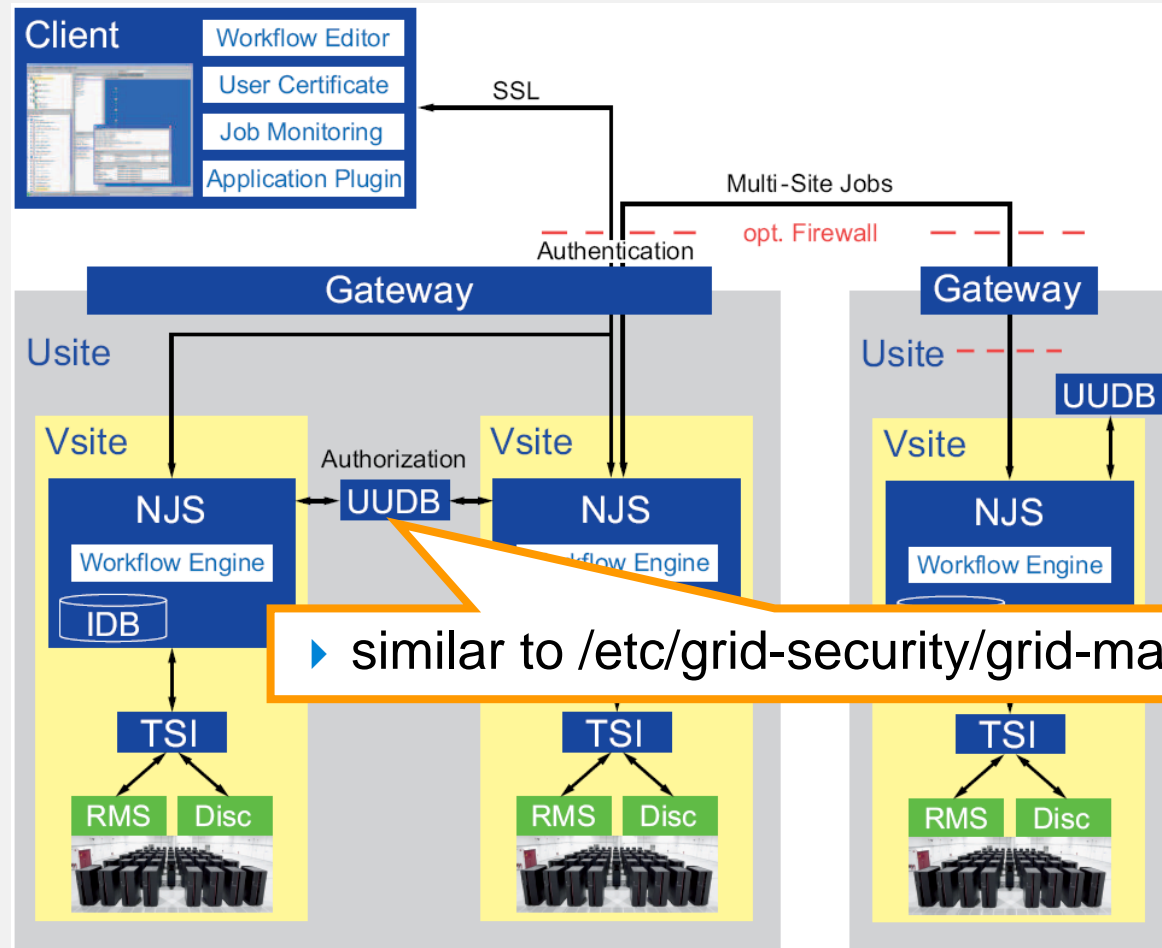


- Current version 5.6 (Client) / 4.6 (Server)
- User Client is platform independent (Java)
- Servers (Unix)
- Target systems (Unix)
 - “no batch“
 - IBM P690, Cray T3E, SP3, VPP, hpcLine, SR 8000, SX-5, PC-Clusters, ... , Globus 2.x as targets
 - NQS, LL, LSF, PBS, CCS, SGE, ...

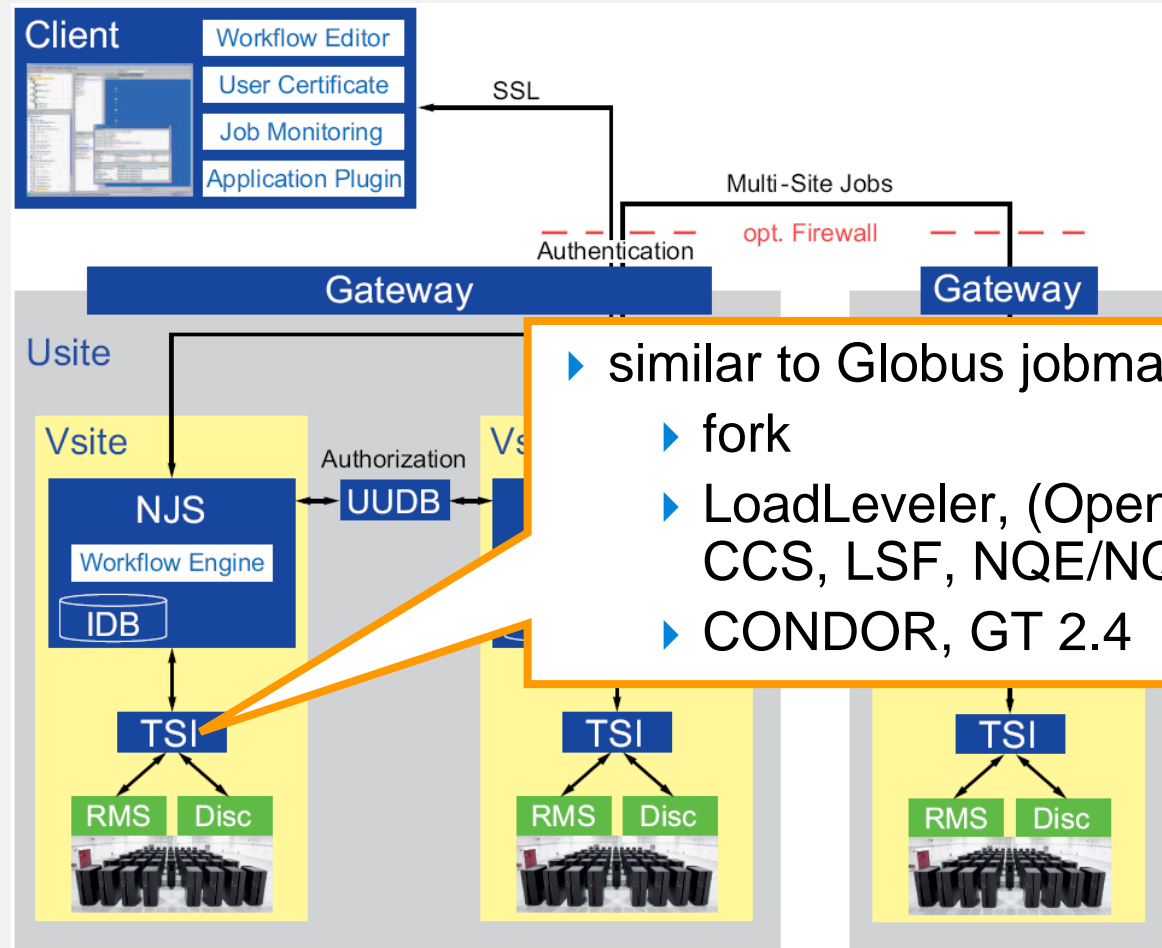
UNICORE Architecture



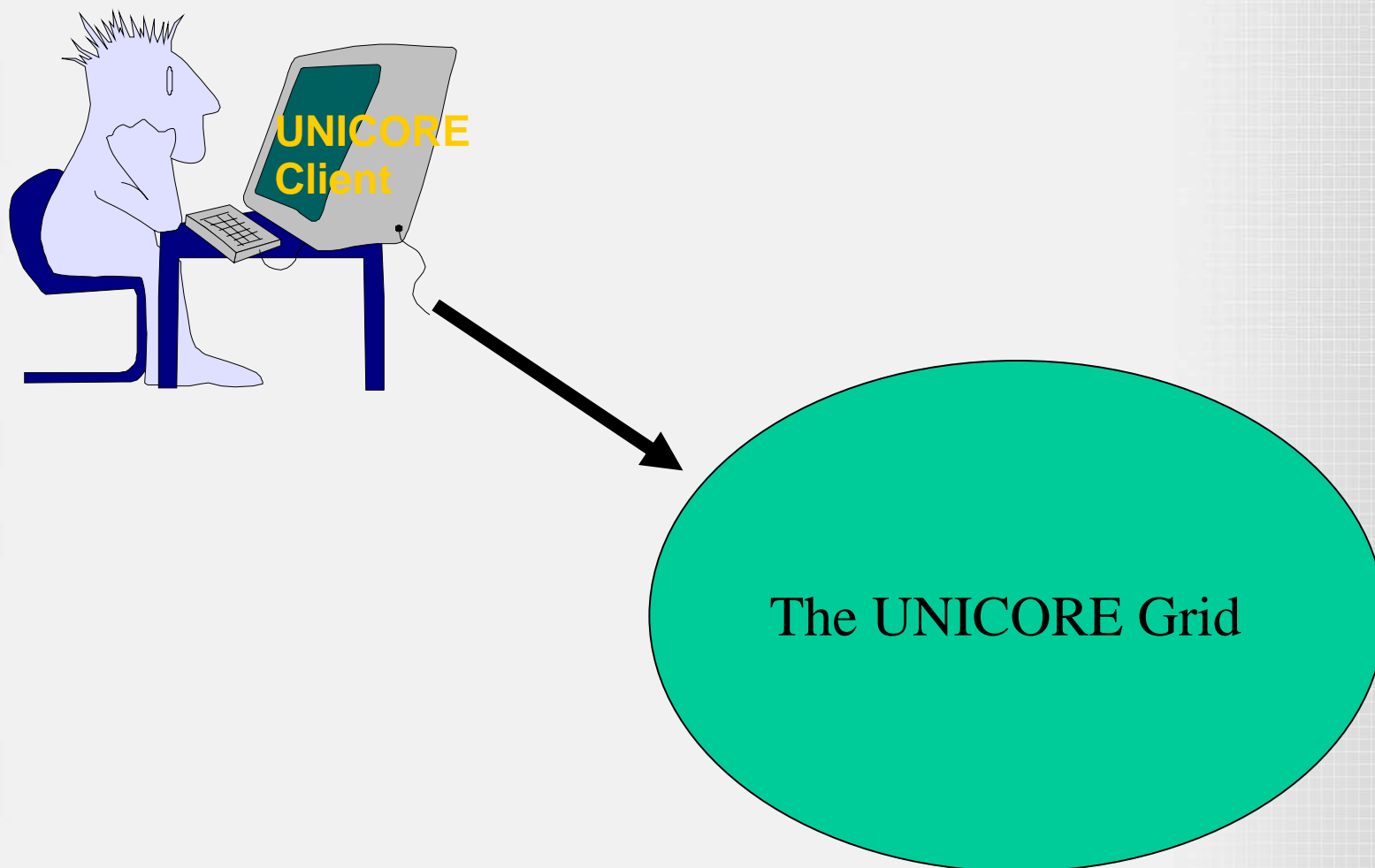
UNICORE Architecture



UNICORE Architecture



UNICORE View

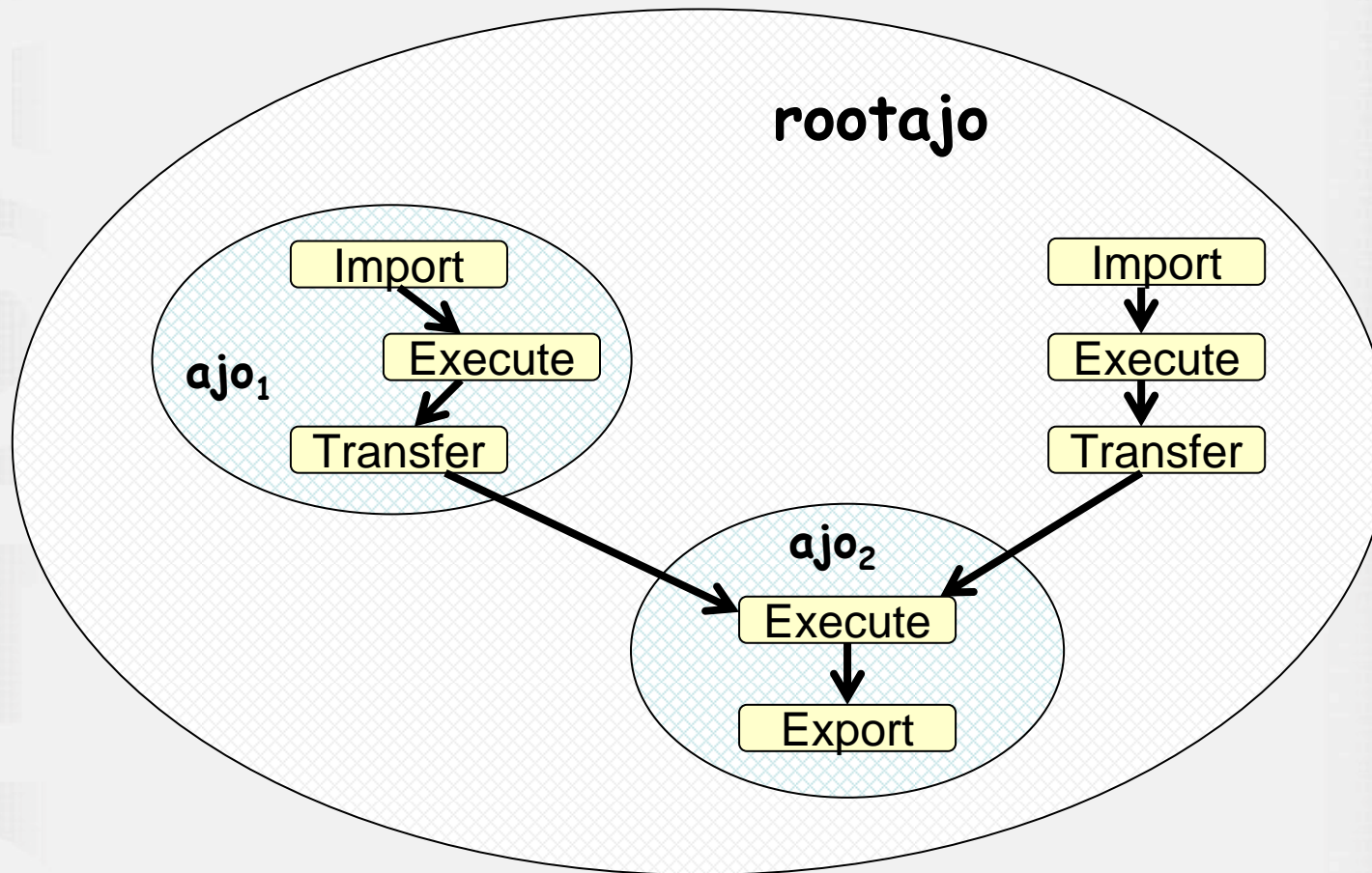


- Job contains
 - Sub-jobs and tasks
 - Dependency information
 - Without dependencies all tasks of a job are executed in “parallel”
 - Workflow: doN, loops, if-then-else
 - Target system location
- Tasks are translated into batch jobs for the destination system by the servers (NJSs)

Abstract Job Object (AJO)

- Abstract, target system independent representation of a job
- Specifies actions to be performed by UNICORE
 - Execute task
 - File transfer task
 - Control task
- Contains dependency graph
- Contains resource requests (nodes, memory, time, ...)
- Contains data set descriptions for data to be streamed
- Realised as Java classes

AJO - Example

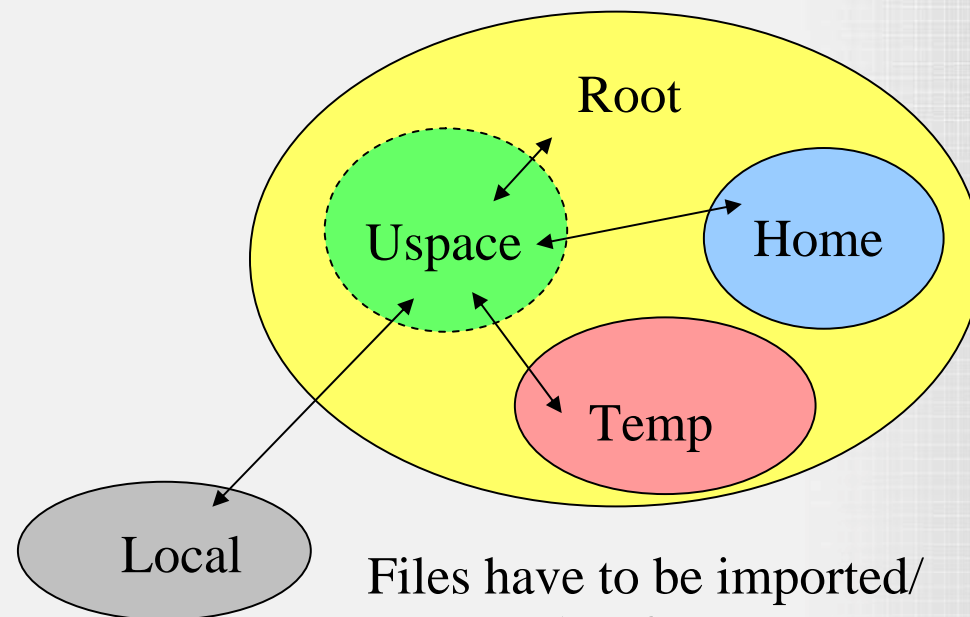


Data Model

- UNICORE file space per UNICORE job
- Non-permanent
- User has to specify remote data location explicitly
- Data import / export / transfer

UNICORE File Spaces

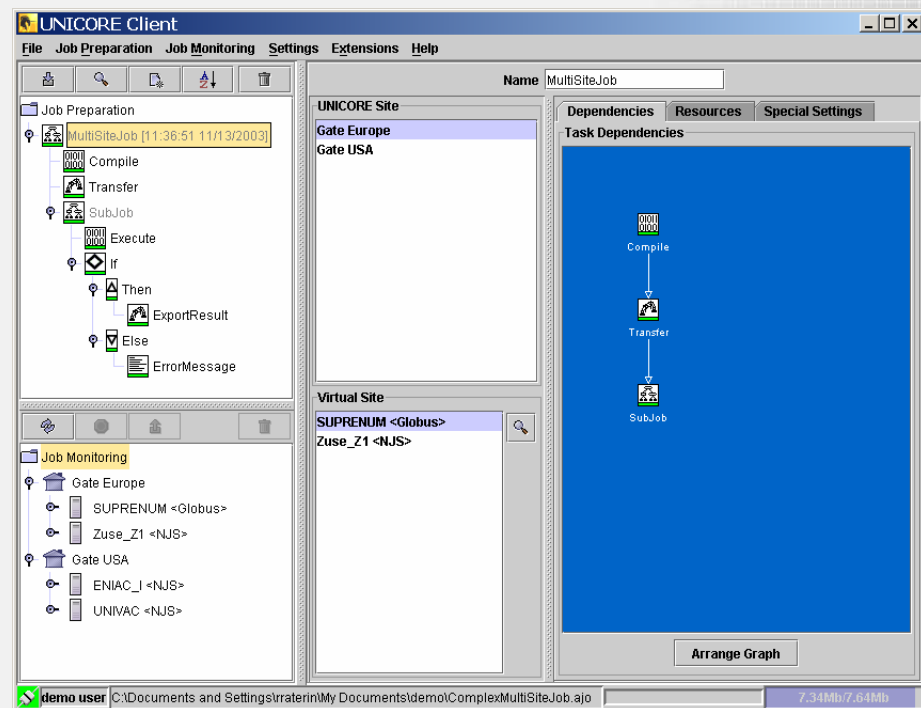
- Temporary file space at the Vsite
 - Job Directory (Uspace) for execution of a (sub-)job
- Permanent file spaces at the Vsite
 - DEISA_HOME
 - DEISA_DATA
 - DEISA_SCRATCH
 - Home: user home
 - Root
 - Temp
- At the Client system
 - Local



Files have to be imported/
exported to/from Uspace

UNICORE Client

- Graphical interface to UNICORE Grids
- Platform-independent Java application
- Functionality:
 - Job preparation, monitoring and control
 - Complex workflows
 - File management
 - Certificate handling
 - Integrated application support



The screenshot shows the UNICORE Client application window. The interface is divided into several panes:

- Job Preparation:** A tree view on the left showing a job named 'New_DEISAJob' with sub-items 'New_DoN1', 'Loop1', and 'Pow_Ray'. A callout bubble labeled 'Job Preparation' points to this pane.
- Job Monitoring:** A list of virtual sites on the left, including DEISA_BSC, DEISA_CSC, DEISA_Cineca, DEISA_FZJ, DEISA_Idris, DEISA_LRZ, DEISA_RZG, and DEISA_Sara. A callout bubble labeled 'Job Monitoring' points to this list.
- Usites:** A list of physical sites in the center, including DEISA_BSC, DEISA_CSC, DEISA_Cineca, DEISA_FZJ, DEISA_Idris, DEISA_LRZ, DEISA_RZG, DEISA_Sara, FZ-Juelich, and VIOLA CAESAR. A callout bubble labeled 'Usites' points to this list.
- Vsites:** A list of virtual sites in the center, including BSC Marenostrum <NJS>, CINECA SP5 <NJS>, CSC <NJS>, FZJ JUMP <NJS>, IDRIS ZAHIR <NJS>, LRZ Altix <NJS>, RZG SP4 <Globus>, and SARA ASTER <NJS>. A callout bubble labeled 'Vsites' points to this list.
- Workflow Management:** A task dependency graph on the right showing a flow from 'New_DoN1' to 'Pow_Ray'. A callout bubble labeled 'Workflow Management' points to this graph.

The status bar at the bottom shows the user 'michael rambadt (2)' and the message 'New_DEISAJob not saved yet.' The system tray shows '15.43Mb/18.99Mb'.

- Security model based on X509 public key infrastructure
- Credential consists of a public and a private key
- No userid and password authentication
- Password protected keystore
- Single sign on
- UNICORE accepts following private key formats:
 - RSA (pkcs12)
 - E.g. Openssl 0.9.7x
 - Java keystore (jks)
 - SUN Java
- Certificates provided for DEISA by all EUGridPMA compliant CAs
- Two server site security entities:
 - Gateway – Authentication
 - NJS – Authorisation

UNICORE Security - Client



- Access to password protected keystore
- Encrypted Keystore contains all imported certificate(s) and the user's private key(s)
- UNICORE Keystore editor allows to
 - Generate a X509 certificate request
 - Import/export .p12 or .jks keystores
 - Import public keys
- The User has to import (at least) three certificates into the Client
 - Pluginsigner's certificate (public key)
 - Gateway signer's certificate (public key)
 - User's signed public key

UNICORE Security - Gateway



- Gateway authenticates the user
- Following checks are performed on certificates presented by a client
 - Certificate is issued by one of the trusted CA (e.g. DFN-CA)
 - Certificate is within its validity period
 - Certificate has not been revoked (if check for Certification Revocation Lists (CRL) is activated)
- Gateway accepts only SSL connections from Clients and other NJSs
 - SSL-Handshake
- Optional SSL connection between Gateway and NJS

UNICORE Security - NJS



- NJS authorizes the user
- Access the UNICORE user Database (UUDB)
 - Maps the user's certificate to his xlogin on the target system
- Development of a „DEISA own“ – UUDB
 - Stores just the Distinguished Name (DN) of the User's certificate
- Only users presenting the DN stored in the UUDB can connect to the target system
- NJS authorises other NJSs
 - Explicit UUDB entry

UNICORE Deployment



- At all project partner sites
- DEISA sites (IDRIS, CINECA, RZ Garching, ...)
- Naregi project (Japan)
- UNICORE 6
- ...

UNICORE Deployment - UNICORE 6



- Next generation of UNICORE
- Based on OGSA (Open Grid Services Architecture)
- Compliant with WS-RF (Web Services Resource Framework)
- Thus interoperability with other Grid middlewares (e.g. Globus 4)
- Available as alpha release on Sourceforge
- Beta release in July 2007, final release in end 2007

OGSA and WS-RF

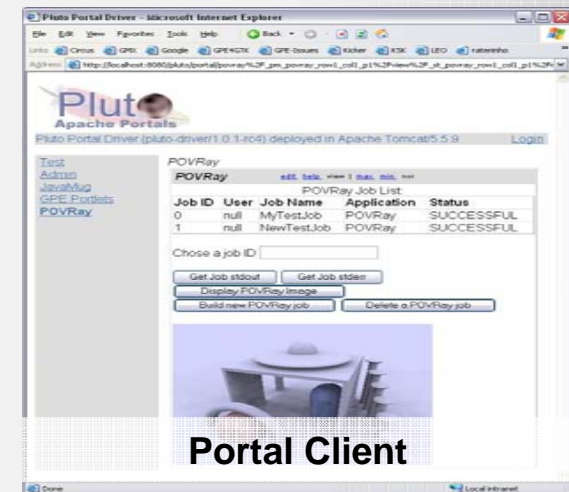
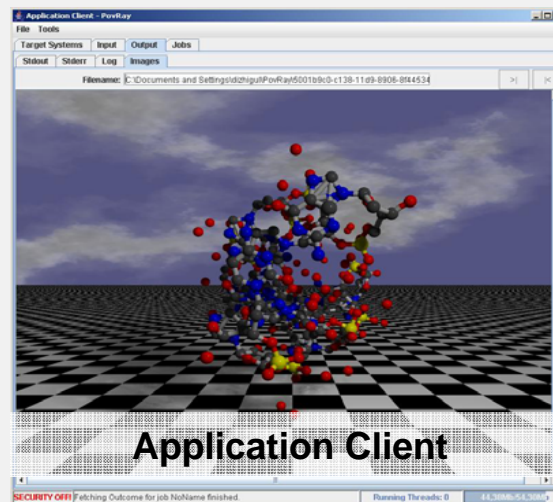
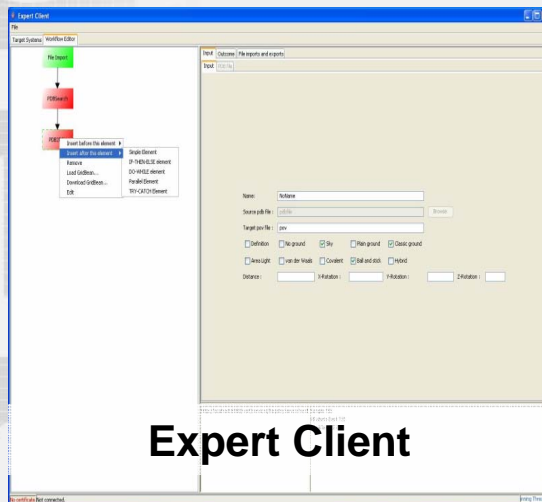
- OGSA (Open Grid Services Architecture): All components of a grid (storage, compute nodes,...) are represented by Grid Services
- One implementation of OGSA: WS-RF (Web Services Resource Framework)
- Services are standardized, they know how they can interact
- Interoperability over a heterogeneous network

WS-RF (Web Services Resource Framework)

- Web Services themselves are stateless: they retain no data between invocations
- But: Grid Services need to keep track of available storage and CPUs, submitted and running jobs,...
- WS-RF defines a standard for stateful Web Services
 - Web Services communicate with Resource Services which store data
 - Clients talking to Web Services have to specify the Resource Services to be used

Grid Programming Environment (GPE) Clients

- Interoperable client framework
 - Expert Client with full access to the Grid
 - Lightweight client with application specific interface
 - Portal Client which can be integrated in UPortal, GridSphere, Jetspeed, ...



UNICORE Dissemination



UNICORE OPEN SOURCE

- <http://www.unicore.eu>

UNICORE FORUM

- <http://www.unicore.org>

UNICORE SUMMIT

- <http://summit.unicore.org>

UNICORE Dissemination - Live CD



- Complete “out-of-the-box” usage of UNICORE 5
- Bootable Linux OS with UNICORE 5 pre-installed
- Does not harm your system → sandbox scenario
- For testing, evaluating, ...

**AVAILABLE HERE
AND AS ISO-IMAGE
ON SOURCEFORGE**

