

UNICORE

Introduction to the Intel Client
and a look behind the scenes...

Grid Summer School, July 28, 2004

Ralf Ratering

Intel

Parallel and Distributed Solutions Division (PDSD)



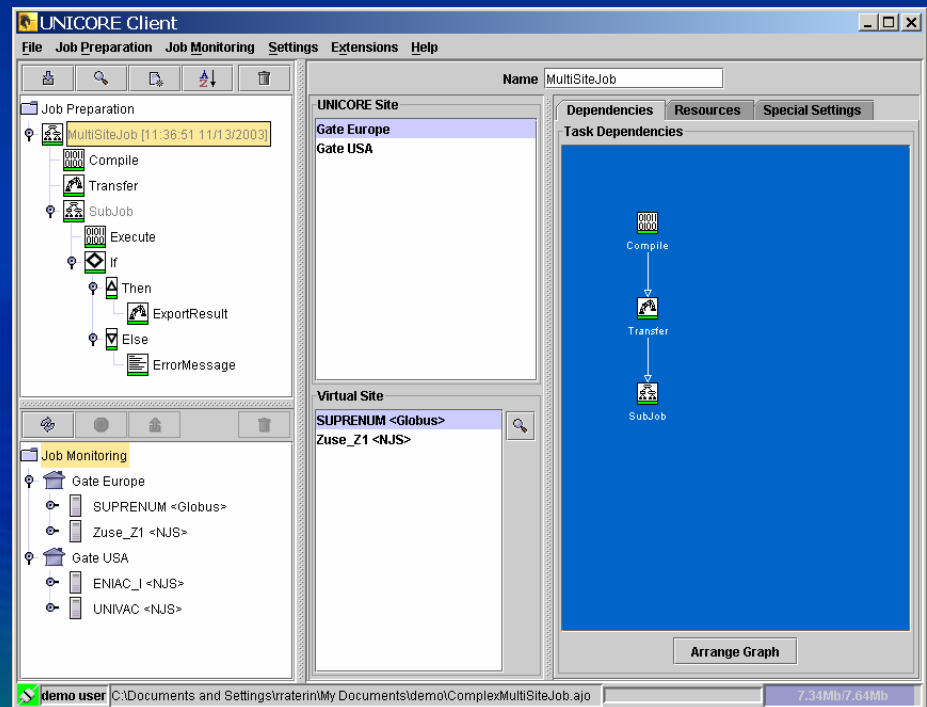
Outline

- Getting started with the UNICORE client
- Constructing jobs in the client
- Integrated application support
- A real-world application

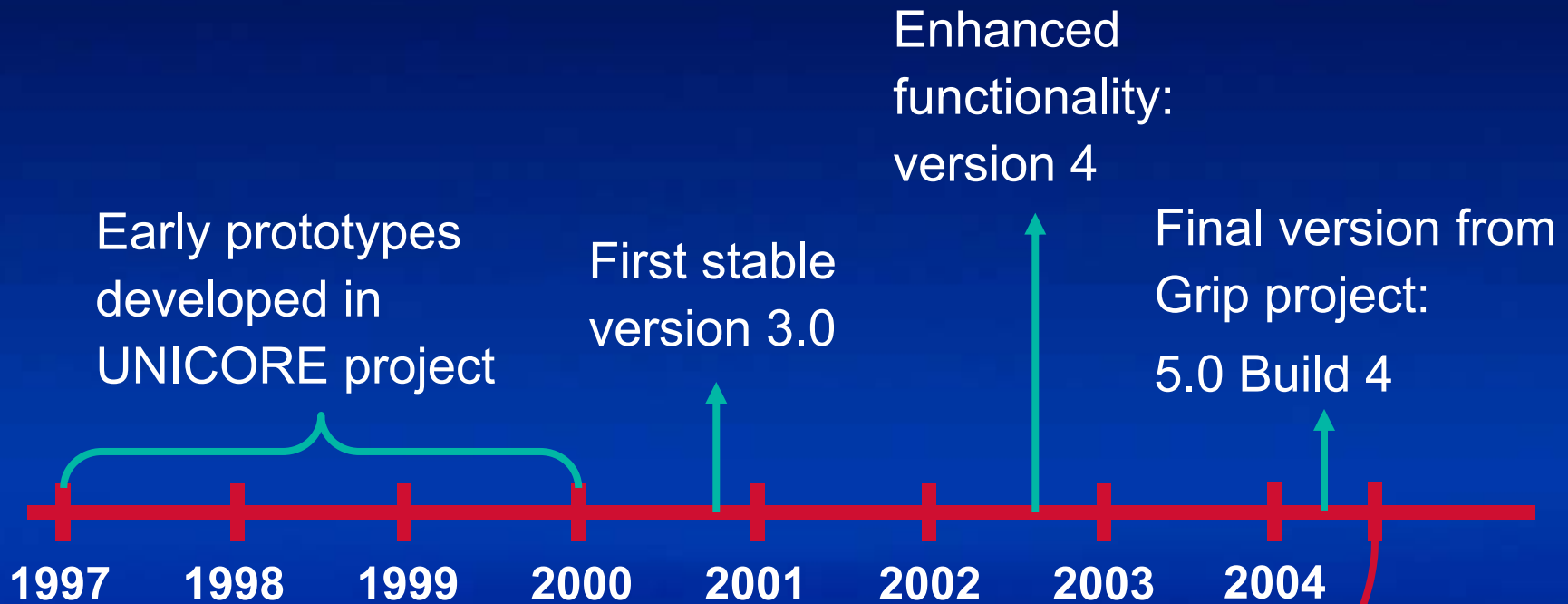
The Intel UNICORE Client

- Graphical interface to UNICORE Grids
- Platform-independent Java application
- Open Source available from UNICORE Forum

- Functionality:
 - Job preparation, monitoring and control
 - Complex workflows
 - File management
 - Certificate handling
 - Integrated application support



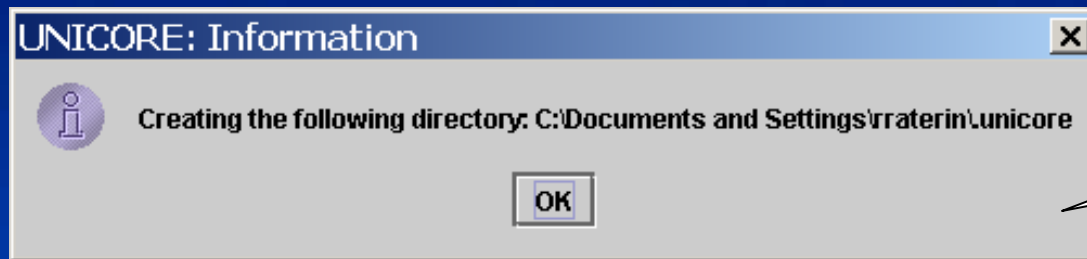
History of UNICORE Client Versions



Now: UNICORE 5.1
OpenSource project at
unicore.sourceforge.net

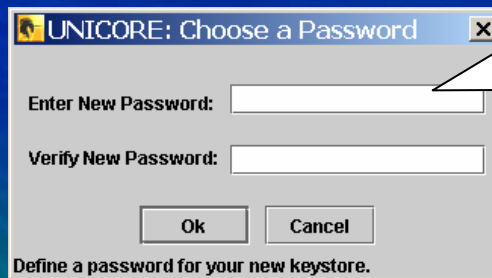
Starting the Client

- Prerequisites: Java \geq 1.4.2



UNICORE configuration directory <.unicore> in your HOME directory

- Automatically creates an empty keystore and imports trusted certificates from „cert“ directory

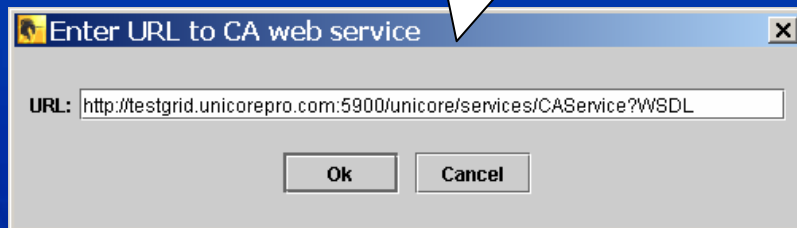


Define password for your unicare keystore file (.unicore/keystore)

Getting a Test Certificate

- „Import test certificates“ from „Settings->Keystore Editor“

CA web service endpoint

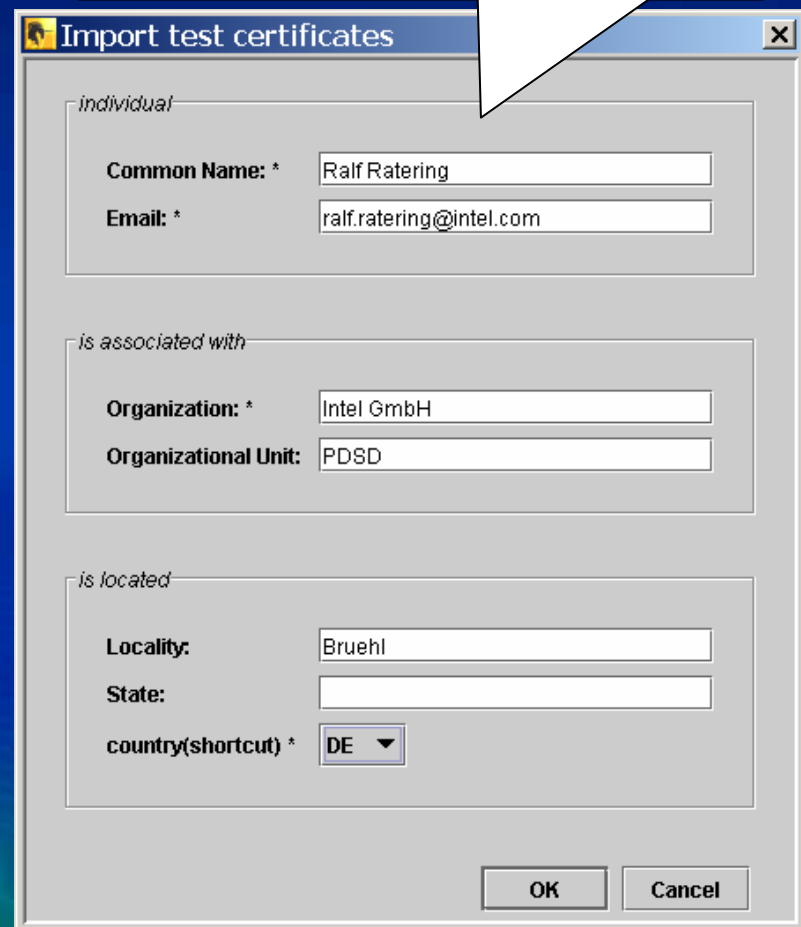


Enter URL to CA web service

URL:

Ok Cancel

Certificate signing request (CSR) Information will be used to generate a test certificate for you.



Import test certificates

individual

Common Name: *

Email: *

is associated with

Organization: *

Organizational Unit:

is located

Locality:

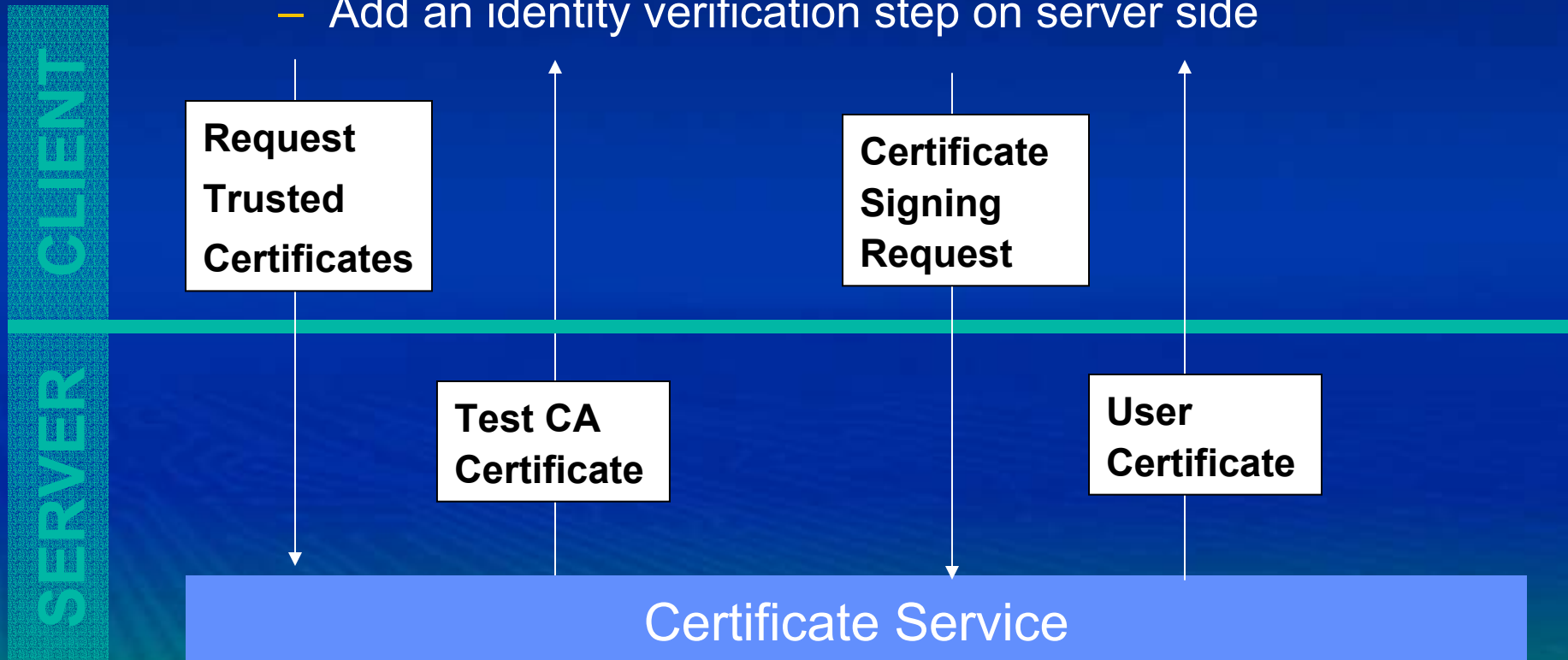
State:

country(shortcut) *

OK Cancel

Certificate Web Services

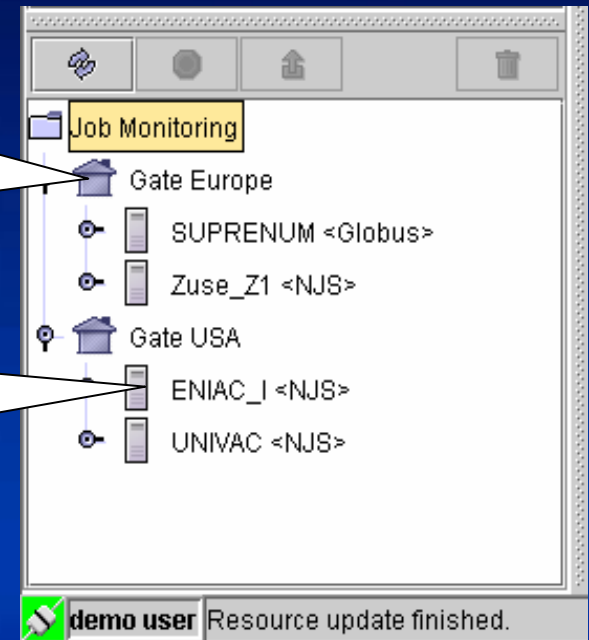
- Low Security Model for Test Grid Access
- Certificates are imported automatically into Client
- Currently implemented at Research Center Jülich:
 - Add an identity verification step on server side



Ready to go? „Hello Grid World!“

UNICORE Site == Gateway
Typically represents a computing center

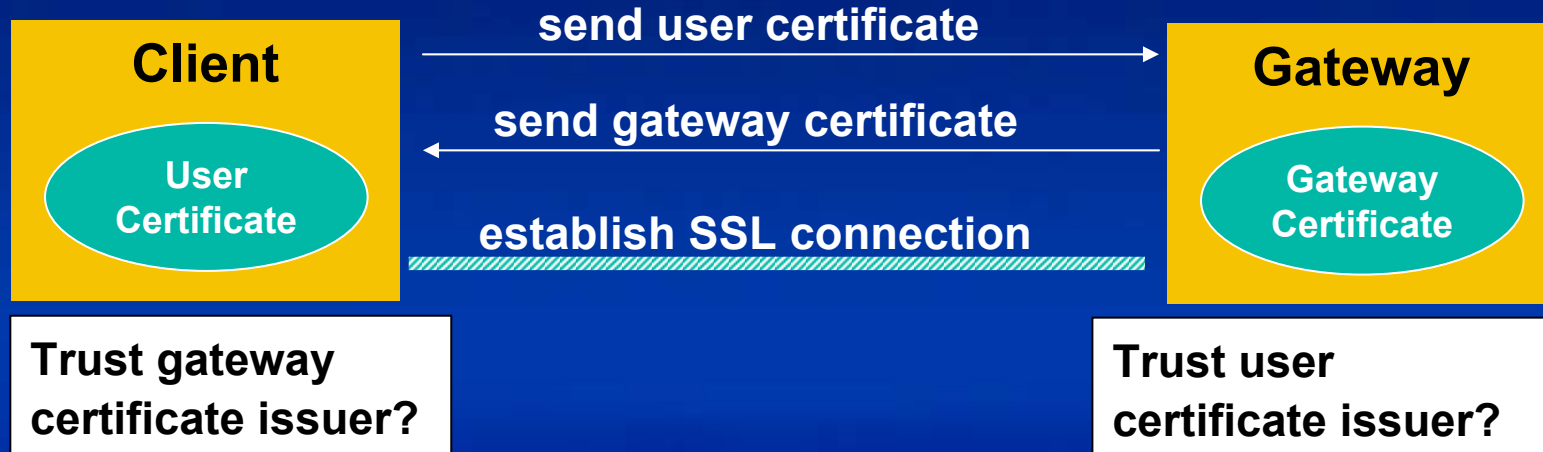
Virtual Site == Network Job Supervisor
Typically represents target system



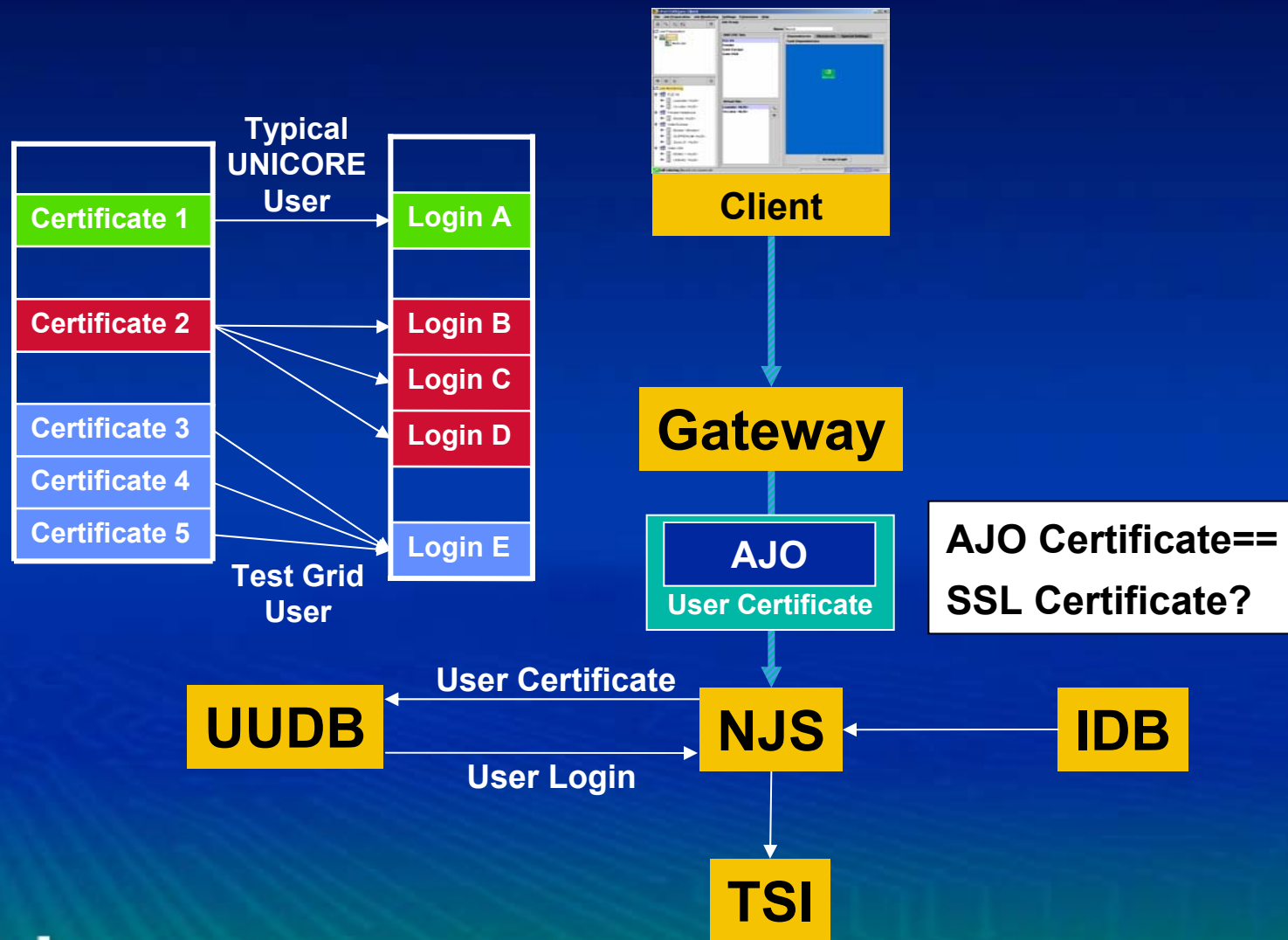
1. Execute a simple script on the UNICORE Test Grid
2. Get back standard output and standard error



Behind the Scenes: Authentication



Behind the Scenes: Authorization

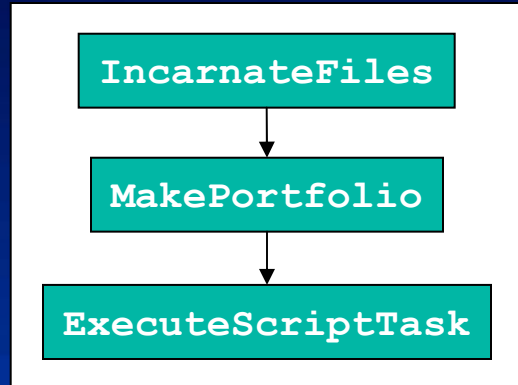


Behind the Scenes: Creation & Submission

CLIENT
SERVER

```
Script Editor Options  
File Edit  
echo "HelloWorld"  
date  
hostname  
ls -ltr
```

Script Container → Abstract Job Object













1. Create file with script contents
2. Wrap file in portfolio
3. Execute portfolio as script

Script_HelloWorld1234...

Job Directory (USpace)
A temporary directory at the target system where the job will be executed



Monitoring the Job Status

-  **Successful:** job has finished successfully
-  **Not successful:** job has finished, but a task failed
-  **Executing:** Parts of a job are running or queued
-  **Running:** Task is running
-  **Queued:** Task is queued at a batch sub system
-  **Pending:** Task is waiting for a predecessor to finish
-  **Killed:** Task has been killed manually
-  **Held:** Task has been held manually
-  **Ready:** Task is ready to be processed by NJS
-  **Never run:** Task was never executed

The Primes Example

```
public void breakKey() {
    try {
        BufferedReader br = new BufferedReader(new FileReader("primes.txt"));
        while (true) {
            inputLine = br.readLine();
            st = new StringTokenizer(inputLine, " ");
            val = new BigInteger(st.nextToken());
            if ( (N.mod(val).compareTo(BigInteger.ZERO)) == 0) {
                p = val;
                q = N.divide(val);
                return;
            }
        }
    } catch (NullPointerException e) {
        System.out.println("Done!");
    } catch (IOException e) {
        System.err.println("IO Error:" + e);
    }
    p = BigInteger.ZERO;
    q = BigInteger.ZERO;
}
```

ArrBreakKey.java

```
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
...
```

Primes.txt

Demo 1: "Gridify" the Primes Example

CLIENT

ArrBreakKey.java

1. Import java file

SERVER

ArrBreakKey.java

2. Compile java file

ArrBreakKey.class

3. Execute class file

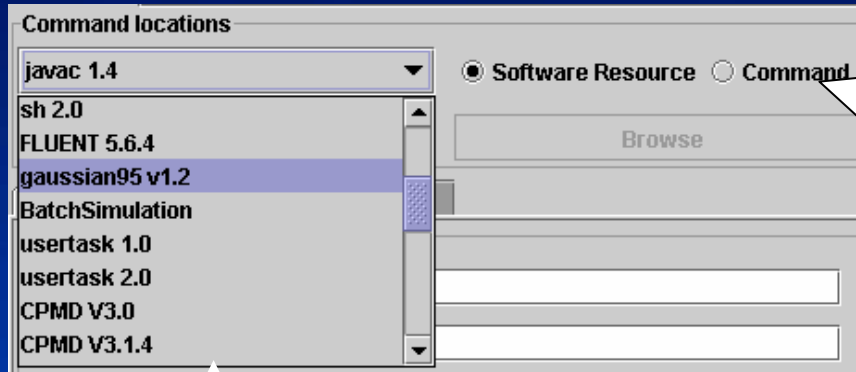
Job Directory
(USpace)

4. Get result in stdout/stderr



Behind the Scenes: Software Resources

CLIENT



Command Task
Executes a software resource, or command (a binary that will be imported into the job directory)

SERVER

```
APPLICATION javac 1.4
Description „Java Compiler“
INVOCATION [
  /usr/local/java/bin/javac
]
END
```

Incarnation Database (IDB)
Application Resources contain system specific information, absolute paths, libraries, environment variables, etc.

Behind the Scenes: Fetching Outcome

CLIENT
SERVER

Session Directory
Configurable in User Defaults:
Paths->Scratch Directory

stdout, stderr
stdout, stderr

Fetch Outcome

ArrBreakKey.java
↓
ArrBreakKey.class

2. Compile java file

stdout, stderr
stdout, stderr

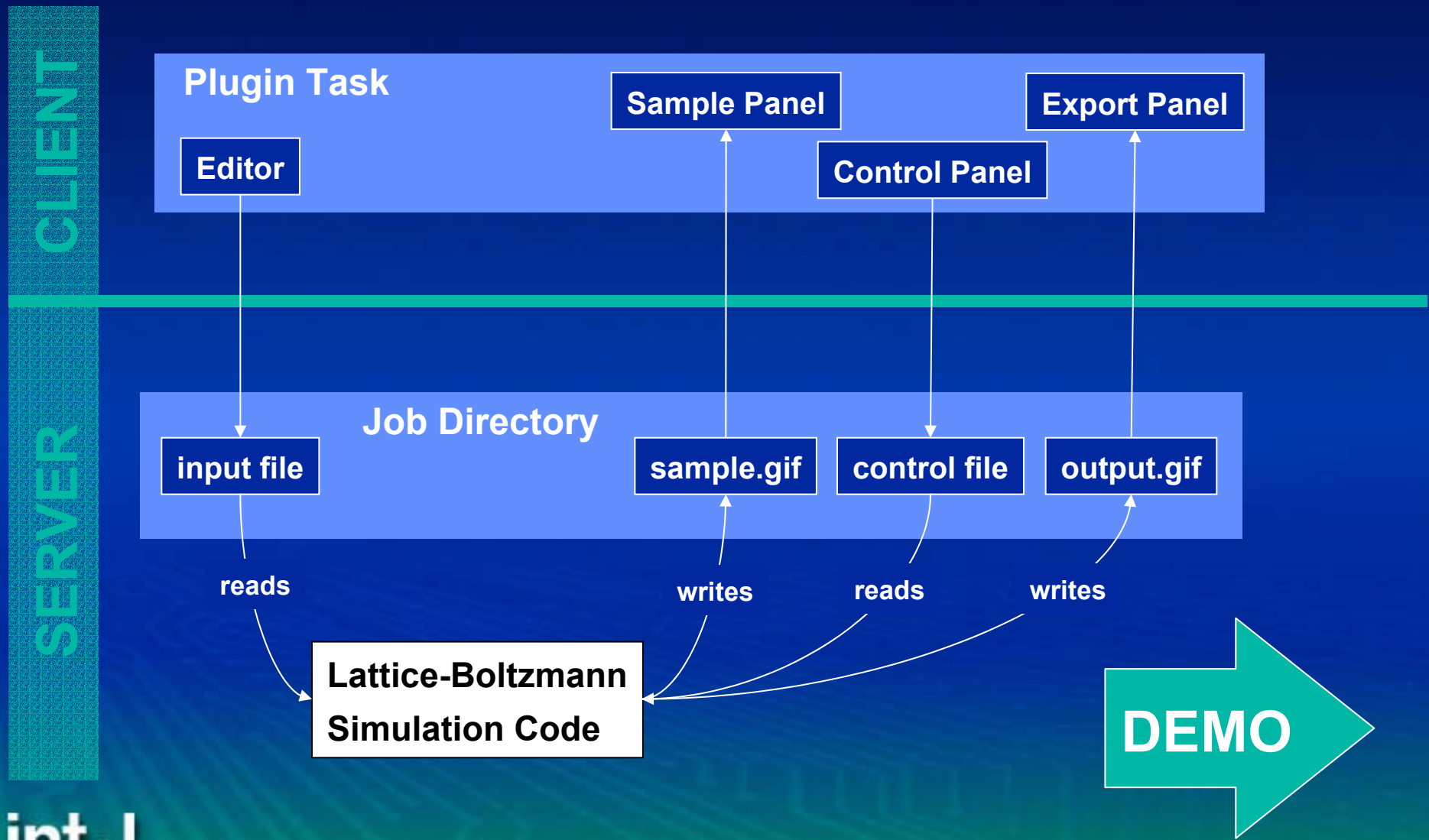
3. Execute class file

Job Directory (USpace)

Files Directory



Demo 2: Steer the Lattice Boltzmann Simulation

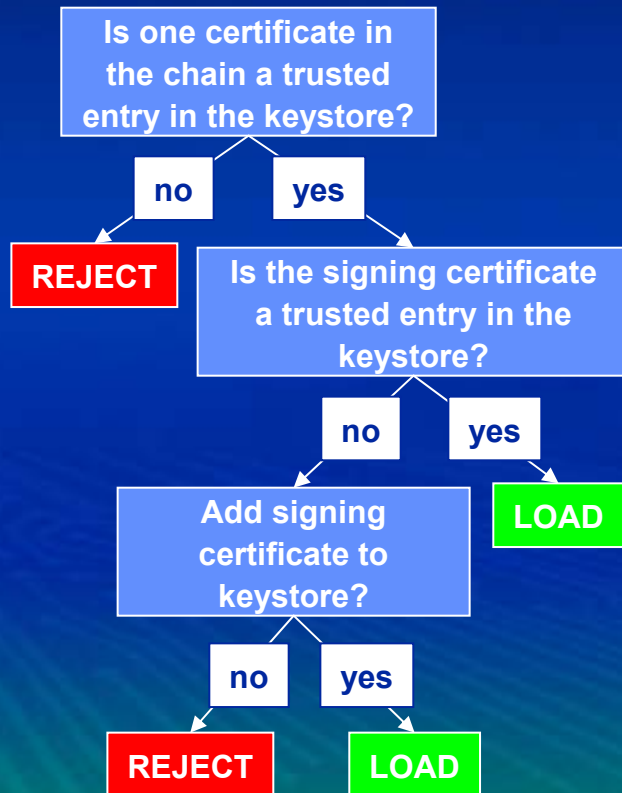


Behind the Scenes: Plug-in Concept

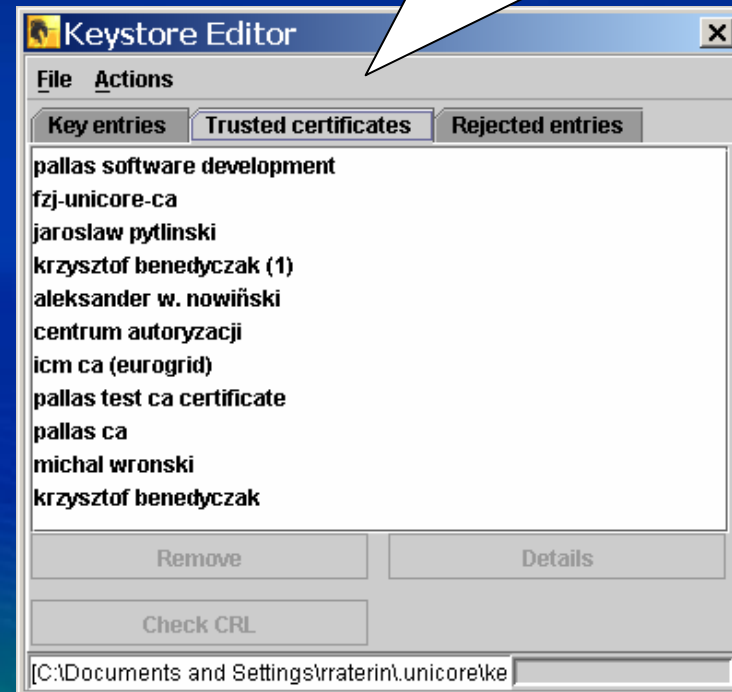
- Add your own functionality to the client!
 - Heavily used in research projects all over the world
 - More than 20 plug-ins already exist
- No changes to basic client software needed
- Plug-ins are written in Java
- Distribution as signed jar archives

Using 3rd Party Plug-ins

- Get plug-in jar file from web-site, email, CD-ROM, etc.
- Store it in client's plug-in directory
- Client will check plug-in signature



Import plug-in certificates from the actions menu in the keystore editor



Task Plug-ins

- Add a new type of task to the client GUI
- New task can be integrated into complex jobs
- Application support: CPMD, Fluent, Gaussian, etc.

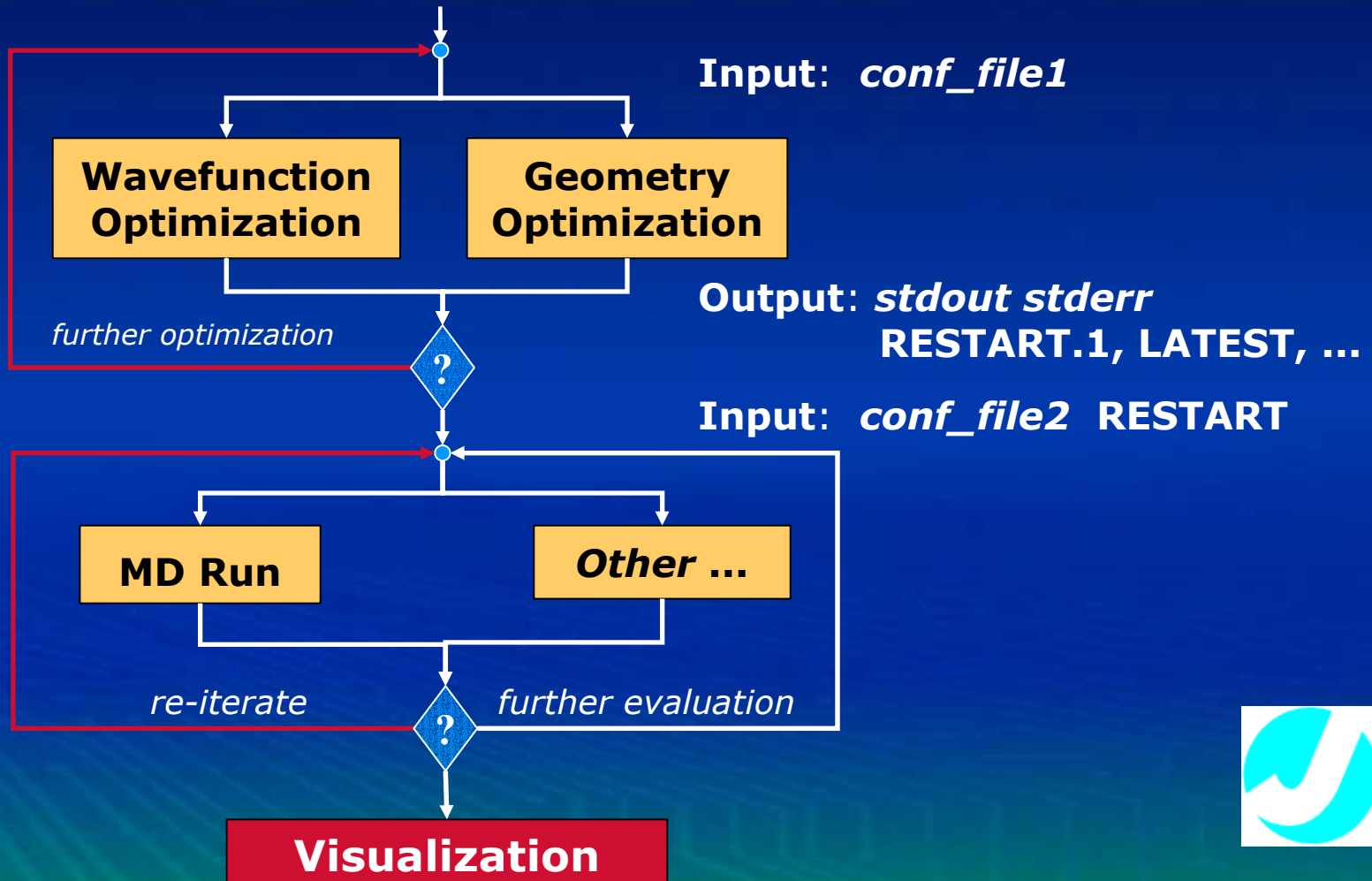
The screenshot displays the UNICOREpro Client interface. On the left, a list of tasks is shown, with 'Add POV-Ray' highlighted. A callout box labeled 'Add task item' points to this menu item. The 'Settings' menu is open, showing 'POV-Ray Defaults' selected, with a callout box labeled 'Settings item' pointing to it. The 'Task Dependencies' panel shows a 'POV-Ray' icon, with a callout box labeled 'Icon' pointing to it. A 'Plugin Info' dialog box is open, showing 'POV-Ray Plugin 1.0' by Ralf Ratering, with a callout box labeled 'Plugin info' pointing to it. The Intel logo is visible in the bottom left corner.

Callout boxes on the left side of the image:

- Add task item
- Settings item
- Icon
- Plugin info

A Task Plug-in: CPMD

- Workflow for Car–Parrinello molecular dynamics code



A Task Plug-in: CPMD

CPMD Plug-In Task used in UNICORE workflows

The screenshot shows the UNICORE 4.0 interface. The 'Execute CPMD' window is open, displaying the following configuration:

- Name: run_wf_optim
- Pseudopotential Library: /usr/local/lib/PP_LIBRARY
- Library Source: Root
- CPMD Version: CPMD V3.4.1
- Visualization: ENERGIES MOVIE

The 'CPMD Editor' window shows the input file content:

```
&CPMD
OPTIMIZE WAVEFUNCTION
&END

&SYSTEM
SYMMETRY
1
CELL
20.52 1.0 1.0 0.0 0.0 0.0
CUTOFF
12.
&END

&ATOMS
SILICON
*SI_SGS KLEINMAN-BYLANDER RAGGIO=1.7
LMAX=P
64
.0000 .0000 .0000 1 1
.0000 5.1300 5.1300 1 1
5.1300 .0000 5.1300 1 1
5.1300 5.1300 .0000 1 1
```

CPMD wizard assists in setting up the input parameters

The screenshot shows the CPMD Wizard 2.5.0 interface. The 'Atom type' dialog box is open, showing the configuration for the 'Si_SGS' atom type:

- Pseudopotential: SI_SGS
- Atomic coordinates table:

Atom	Number of atoms	X-coord	Y-coord	Z-coord
1	1	0	0	0
2	1	5.13	5.13	5.13
3	1	5.13	0	5.13
4	1	5.13	5.13	0
5	1	2.565	2.565	2.565
6	1	2.565	2.565	7.695
7	1	7.695	2.565	2.565
8	1	7.695	7.695	2.565
9	1	0	0	15.21

The dialog also includes options for 'Nonlocal parts of pseudopotential', 'Pseudopotential nonlocally', and 'Atomic basis'.

A Task Plug-in: CPMD

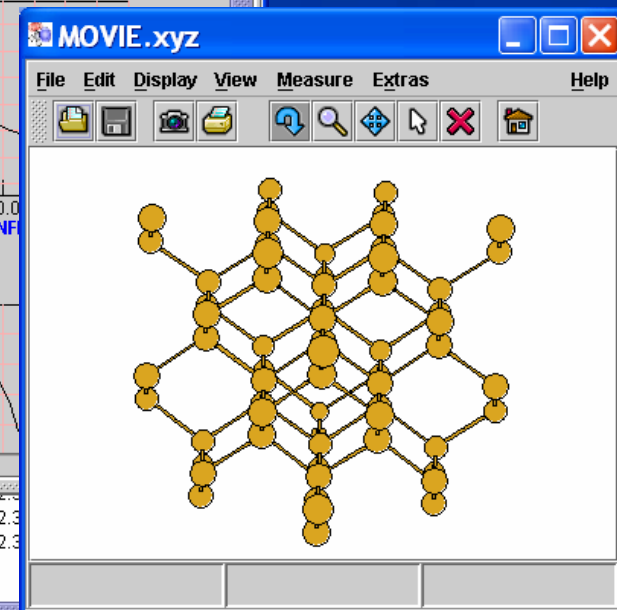
- Visualize results



The screenshot displays the UNICORE 4.0 build 8 Job Monitoring interface. The main window shows a job tree for 'cpmd_job1' with steps: 'run_wf_optim', 'if_okay', 'Then', 'run_mdrun', 'Else', and 'save_files'. The 'run_mdrun' step is currently active. The 'ENERGIES' tab is selected, showing the 'ENERGIES file' path and 'Time step' set to 15. Below this, four graphs are displayed: 'EKINC' (Kinetic Energy), 'TEMP' (Temperature), 'ECLASSIC' (Classical Energy), and 'EHAM' (Hamiltonian Energy). The 'EKINC' graph shows a peak around 10.0 NFI. The 'TEMP' graph shows a decrease from 298.0 to 294.0. The 'ECLASSIC' and 'EHAM' graphs show values around -252.3485 and -252.34836 respectively. A table of data points is visible at the bottom of the graph area.

Time step	Energy	Temperature	Kinetic Energy		
18	0.00062668	293.478	-252.4361343633	-252.348336318	-252.3
19	0.00062622	293.018	-252.4366744521	-252.3489903004	-252.3
20	0.00063139	292.520	-252.4365299368	-252.3489948630	-252.3

Wed Nov 13 13:47:46 CET 2002: [INF] Reading data finished.



Supporting an application at a site

- Install the application itself
- Add entry to the Incarnation Database (IDB)

```
APPLICATION CPMD 3.4.1
Description „Car Parrinello Molecular Dynamics Code“
INVOCATION [
    export JOBTYPPE=8E8;
    /usr/mpi/bin/mpiexec -p IAPAR -n $UC_PROCESSORS
    /usr/local/bin/cpmd.x      $CPMD_FILE  $PP_LIBRARY
]
```

Extension Plug-ins

- Add any other functionality
- Resource Broker, Interactive Access, etc.

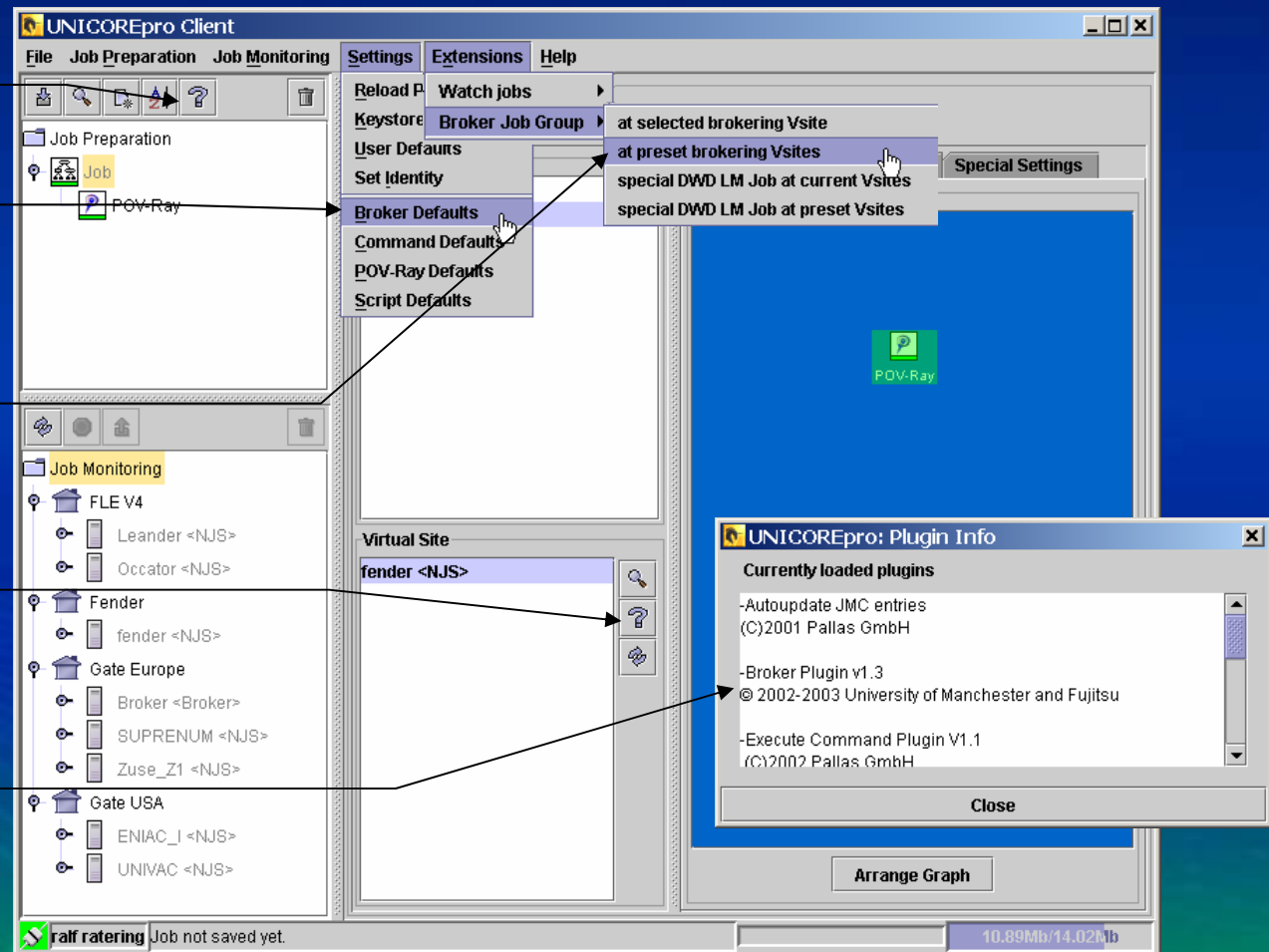
JPA toolbar

Settings
item

Extensions
menu

Virtual site
toolbar

Plugin info



An Extension Plug-in: Resource Broker

- Specify resource requests in your job
- Submit it to a broker site
- Get back offers from broker

The screenshot displays the UNICORE 4.1 software interface. The main window is titled 'UNICORE 4.1' and has a menu bar with 'File', 'Job Preparation', 'Job Monitoring', 'Settings', 'Extensions', and 'Help'. The 'Job Preparation' tab is active, showing a 'Job Group' with the name 'SubJob_Job'. Below this, there are tabs for 'Unicore Site', 'Resources', and 'Special Settings'. A dialog box titled 'Offers from Broker' is open, displaying a table of offers. The table has columns for 'Virtual Site', 'Unicore Site', 'Start Time', 'End Time', and 'Cost'. The offers are as follows:

Virtual Site	Unicore Site	Start Time	End Time	Cost
T3E_turing	Manchester Univer...	1:49 PM to 2:19 PM	1:49 PM to 2:19 PM	1.00741E-4 CSAR ...
T3E_turing	Manchester Univer...	1:49 PM to 2:19 PM	1:50 PM to 2:20 PM	1.00741E-4 CSAR ...
O2000_fermat	Manchester Univer...	1:49 PM to 2:49 PM	1:49 PM to 2:49 PM	1.61875E-4 CSAR ...
O2000_fermat	Manchester Univer...	1:49 PM to 2:49 PM	1:49 PM to 2:49 PM	1.61875E-4 CSAR ...
O300_wren	Manchester Univer...	1:49 PM to 1:49 PM	1:49 PM to 1:49 PM	2.06475E-4 CSAR ...
O300_wren	Manchester Univer...	1:49 PM to 1:49 PM	1:49 PM to 1:50 PM	2.06475E-4 CSAR ...

Below the table, there is a 'Tickets in detail' dialog box showing the following information:

The tickets contain the following:

Ticket for Task Script1
Ticket ID: org.unicore.AAIdentifier@2bbe9ee4
Ticket valid until: Tue Jul 08 14:49:23 BST 2003

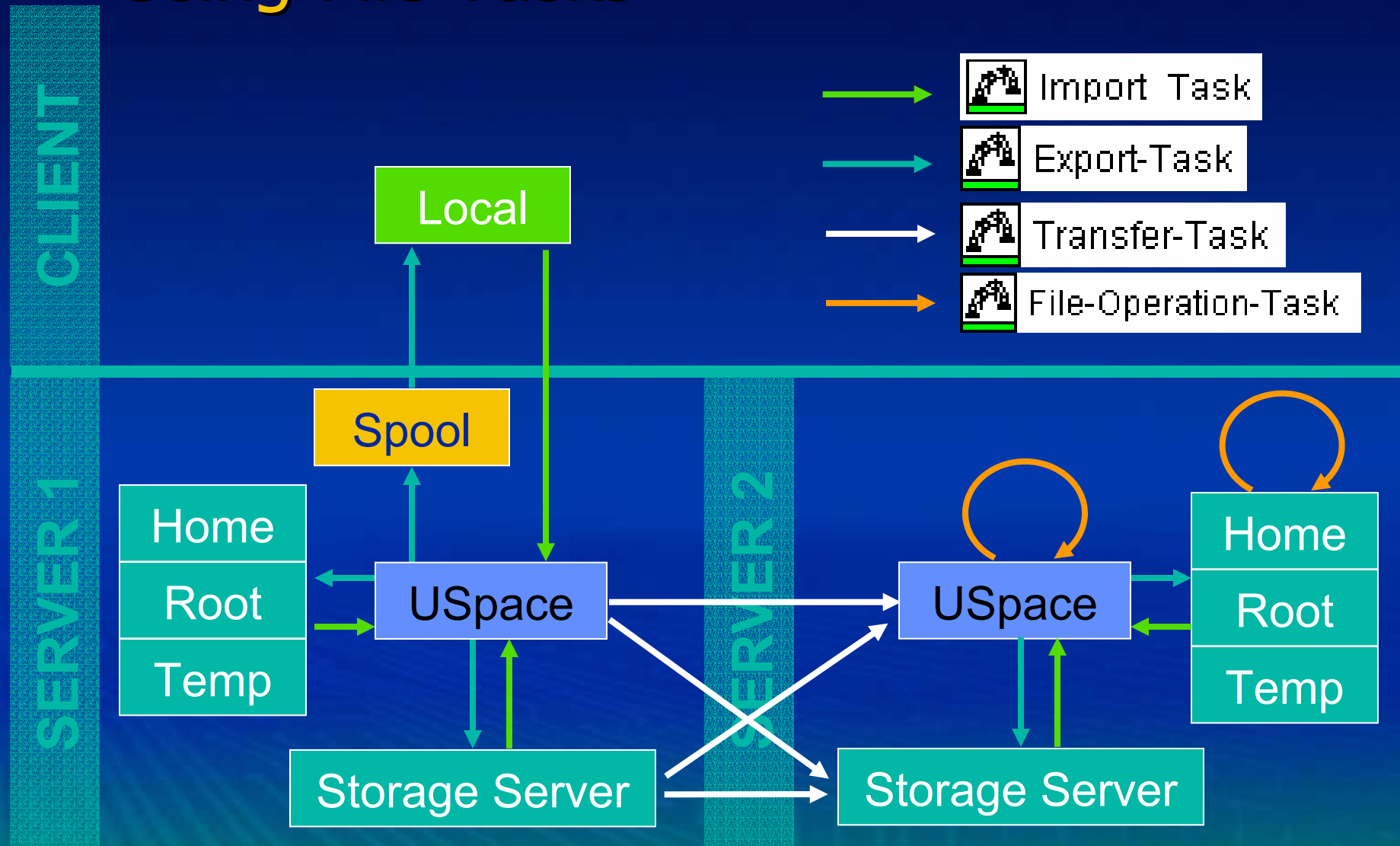
ResourceSet:
Node (Number of Nodes) = 1.0 Nodes
Processor (Number of PEs per Node) = 1.0 Processors per node
Memory (Total Amount of Memory) = 20.0 Megabytes per node
RunTime (Time per Job) = 15.0 Seconds
org.unicore.resources.QoSCheck: QoSCheck (QoSCheck)



Existing Plug-Ins (incomplete)

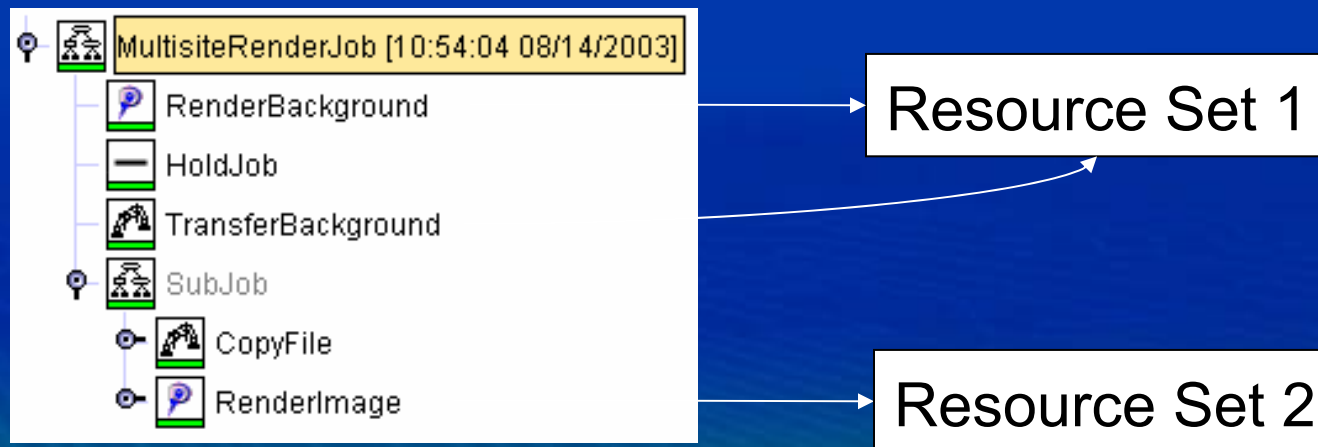
- CPMD (FZ Jülich)
- Gaussian (ICM Warsaw)
- Amber (ICM Warsaw)
- Visualizer (ICM Warsaw)
- SQL Database Access (ICM Warsaw)
- PDB Search (ICM Warsaw)
- Nastran (University of Karlsruhe)
- Fluent (University of Karlsruhe)
- Star-CD (University of Karlsruhe)
- Dyna 3D (T-Systems Germany)
- Local Weather Model (DWD)
- POV-Ray (Pallas GmbH)
- ...
- Resource Broker (University of Manchester)
- Interactive Access (Parallab Norway)
- Billing (T-Systems Germany)
- Application Coupling (IDRIS France)
- Plugin Installer (ICM Warsaw)
- Auto Update (Pallas GmbH)
- ...

Using File Tasks



How to specify resource requests?

- Tasks can have resource sets containing requests
- If not resource set is attached, default resources are used
- Resource sets can be edited, loaded and saved
- If a resource request does not match resources available at a site, the client displays an error

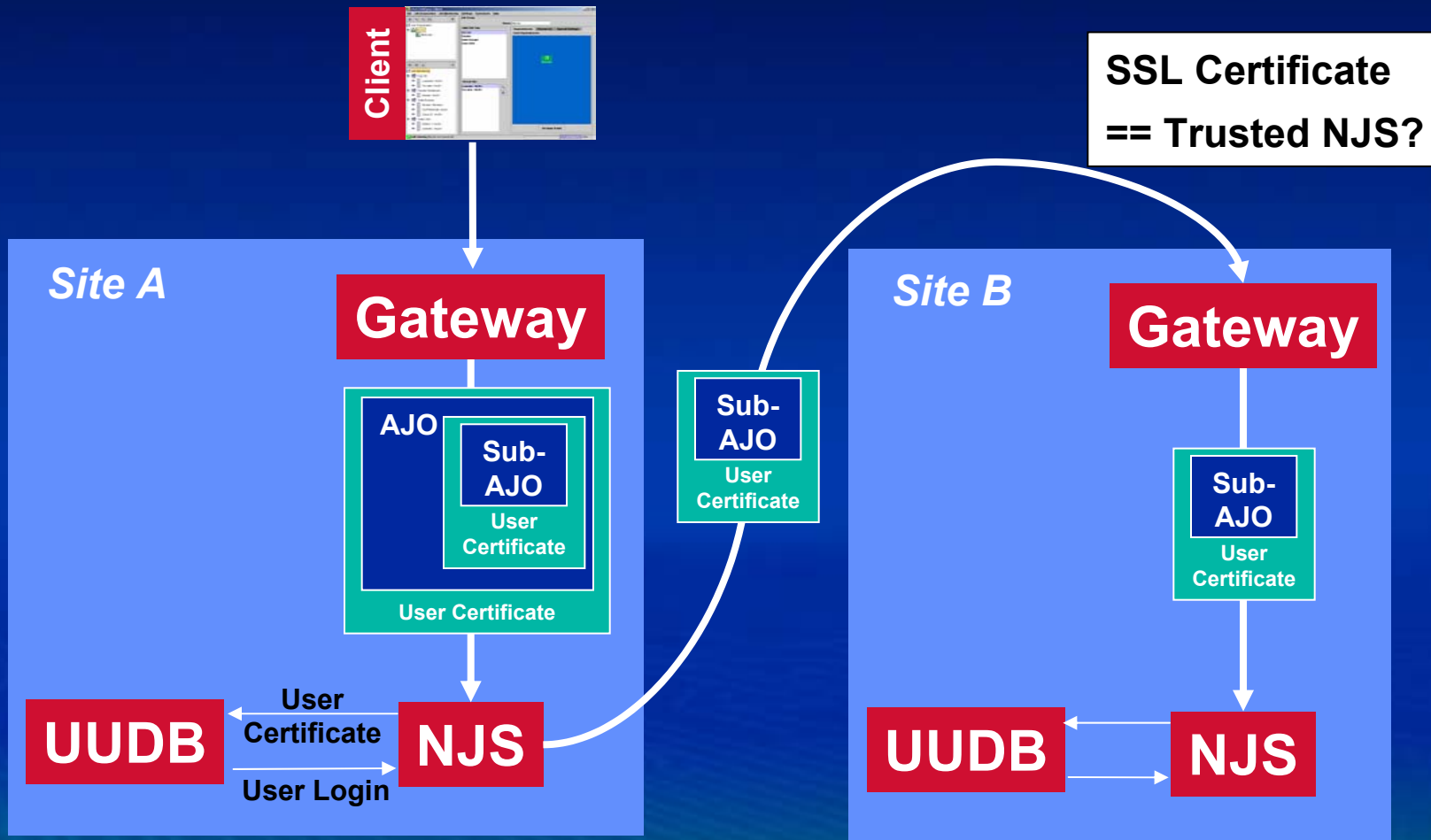


Demo 3: Run a multi site job

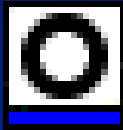
1. Use the primes example
2. Compile the source file on one virtual site
3. Transfer the resulting class file to a sub job running at a different virtual site
4. Execute the class file in the sub job



Behind the Scenes: Authorization



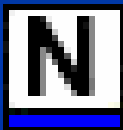
Complex Workflow: Control Tasks



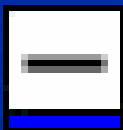
Do Repeat Loop



If Then Else



Do N Loop



Hold Task

Return Code Test

Repeat Until:

Value: disabled
return code equals
return code not equal
successful
not successful

Task:

File Test

Repeat until:

Location: disabled
file exists
file not exists

Filename: file is readable
file is writable
file is executable

Time Test

Repeat Until:

Time (hh:mm:ss): disabled
execution time after

Date (mm/dd/yyyy): Today

UTC Time: 15:43:03 12/03/2003

Demo 4: Test the return code in a loop

```
import java.util.Random;

public class Application {

    public static void main(String[] args) {

        Random rnd = new Random(System.currentTimeMillis());

        double random = rnd.nextDouble();

        System.out.println("RANDOM: " + random);

        int exitCode = (int) (5*random);

        System.out.println("EXIT CODE: " + exitCode);

        System.exit(exitCode);

    }

}
```

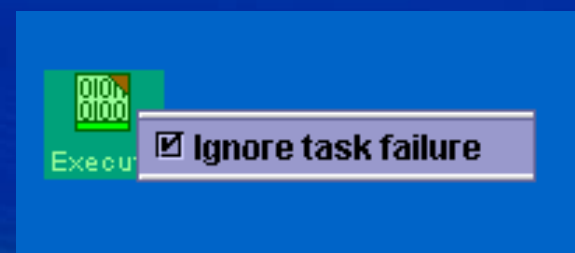
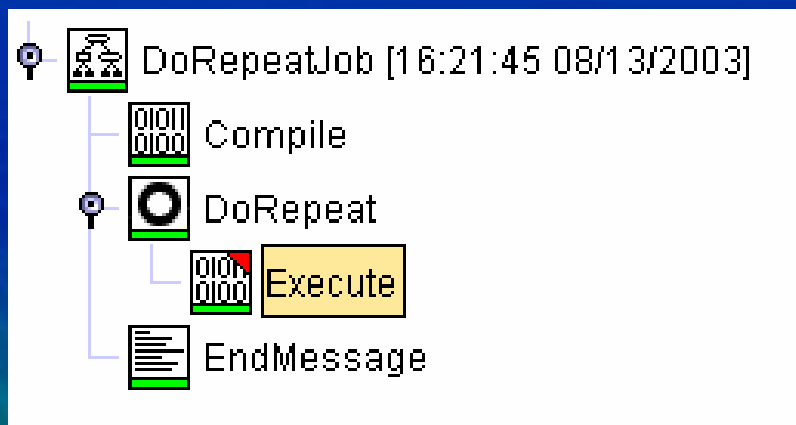
Repeat execution until it fails with a exit code 2!



DEMO

Behind the Scenes: Ignore Failure

- UNICORE jobs stop execution when a task fails
- Sometimes Task failure is acceptable
 - If and DoRepeat conditions
 - Tasks that try to use restart files
 - Whenever you do not care about task success
- Set „Ignore Failure“ flag on Task



Right Mouse Click in
Dependency Editor

Loops: Accessing the iteration counter

- Iteration variable: `$UC_ITERATION_COUNTS`
- Lives on server side
- Supported in
 - Script Tasks
 - File Tasks
 - Re-direction of stdout/stderr
- Nested loops: iteration numbers are separated by „_“, e.g. „2_3“
- Caution: counter will not be propagated to sub jobs

Integrated Application Example: POV-Ray

CLIENT

Scene Description

```
#include "colors.inc"
#include "shapes.inc"
camera {
  location <50.0, 55.0, -75.0>
  direction z
}
plane {y, 0.0 texture {pigment {RichBlue }}}
object { WineGlass translate -x*12.15}
light_source { <10.0,50.0,35.0> colour White }
...
```

Command Line Parameters

Include Files

Display



SERVER

Input Files

Output Image

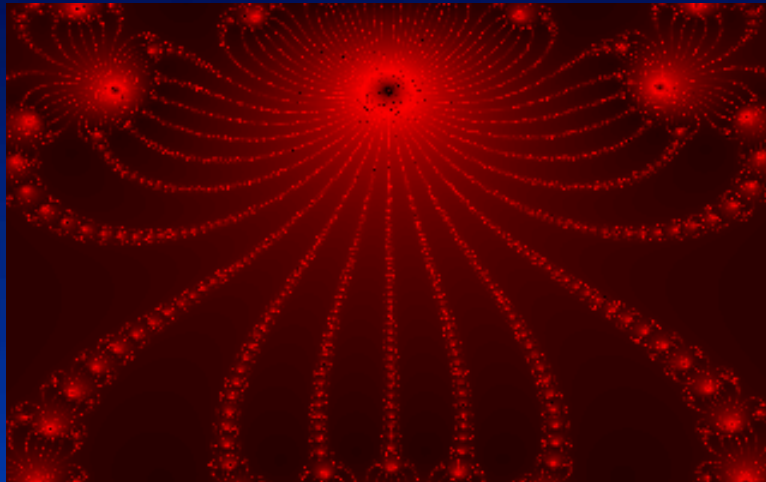
Libraries

POV-Ray Application

Job Directory (USpace)

Remote File System (XSpace)

Demo 5: Hold and release a job

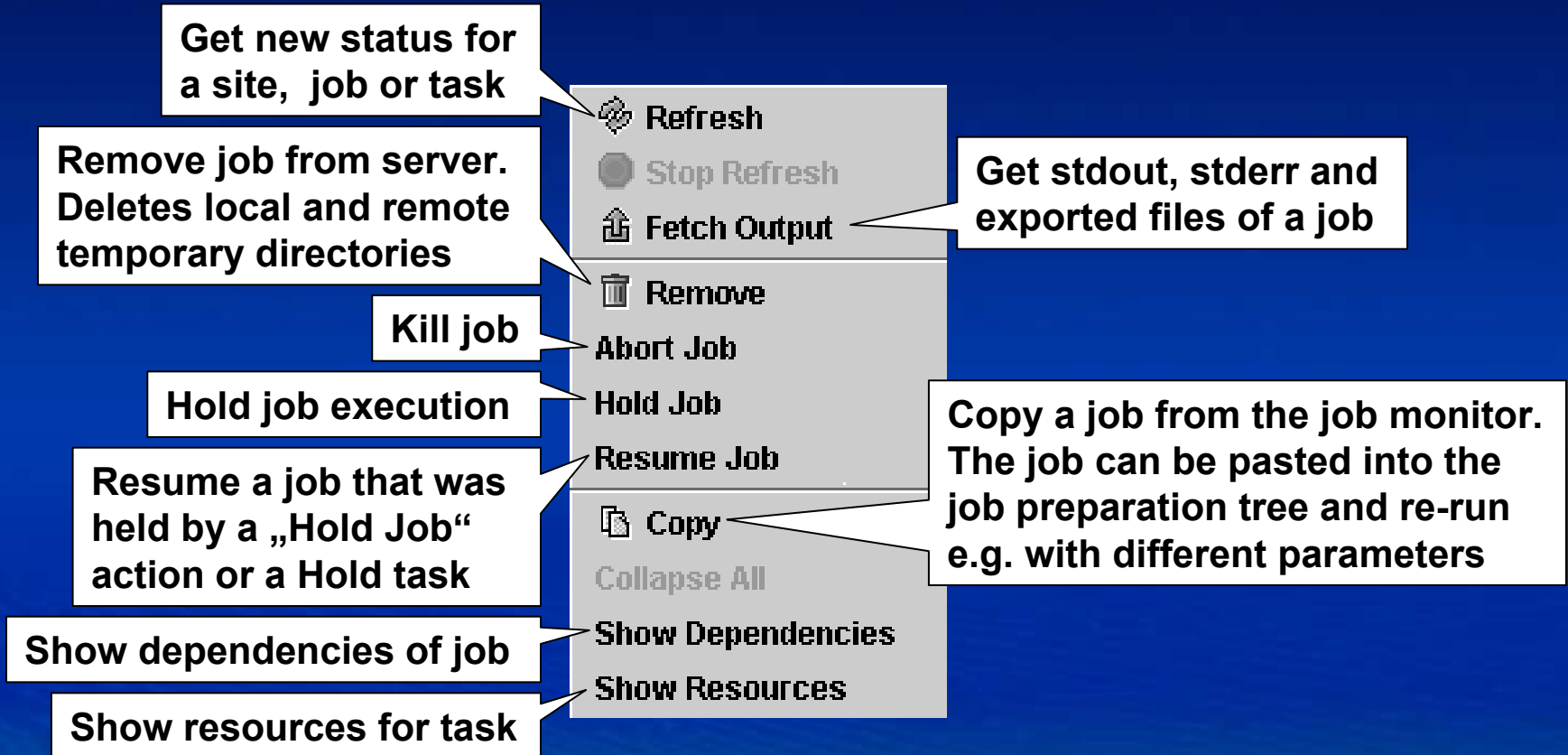


Demo Images from Pov-Ray Distribution

1. Render Background Image
2. Hold Job to check Image
3. Manually Resume Job Execution
4. Render Final Image

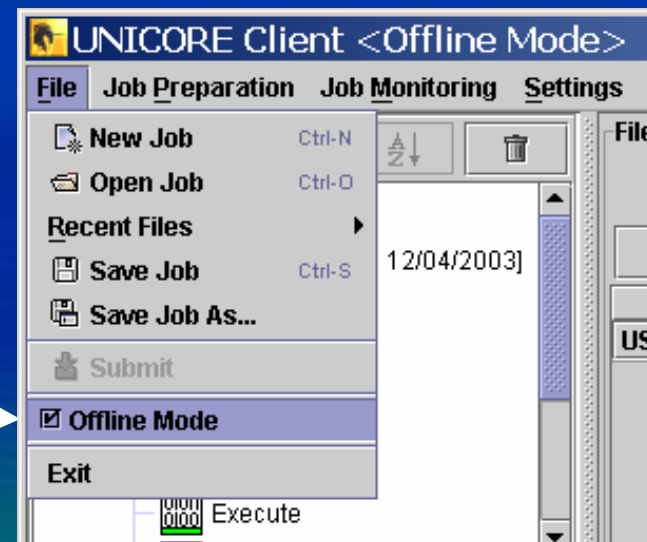


Job Monitor Actions



Caching Resource Information

- Client works on cached resource information
 - UNICORE Sites, Virtual Sites, available resources
- Resource cache will be updated on...
 - ... startup
 - ... refresh on „Job Monitoring“ tree node
- Client uses cached information in offline mode



Accessing other UNICORE Sites

Job Monitor Root

Performing a „Refresh“ on this node will reload UNICORE Sites

UNICORE Sites

will be read from an XML file
Can be a URL on the web

Virtual Sites

are configured at the UNICORE Site

UNICORE: User Defaults

General Paths System Settings User Interface Logging Settings

User Email: ralf.ratering@intel.com

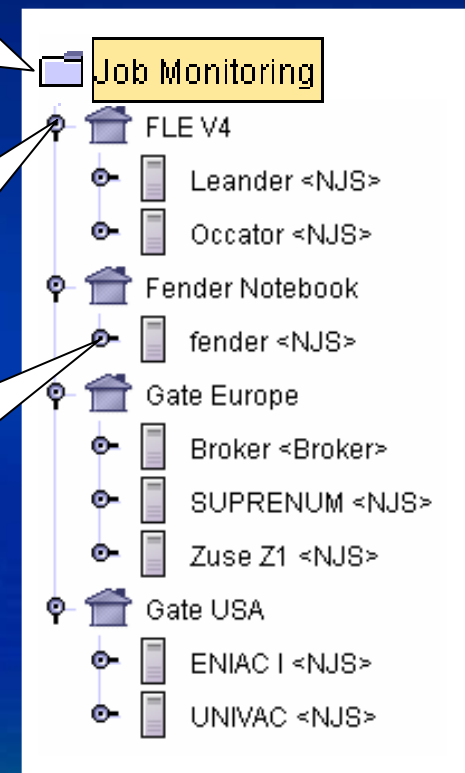
URL(s) for UNICORE Site Server: http://www.unicorepro.com/unicoreSites.xml
file://Z:/docs/unicorePro/unicoreHumpty.xml
file://Z:/docs/unicorePro/naregi.xml

Certificate Revocation List: https://unicore-ca.lrz-muenchen.de/unicore-ca.crl

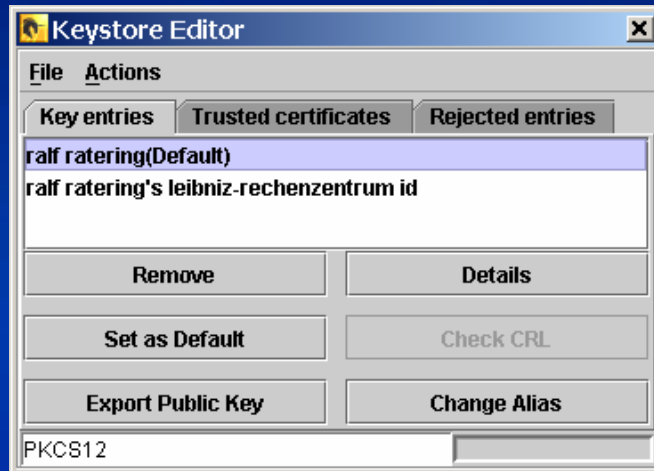
Proxy Server Address: Port: 1080 On/Off:

No proxy for:

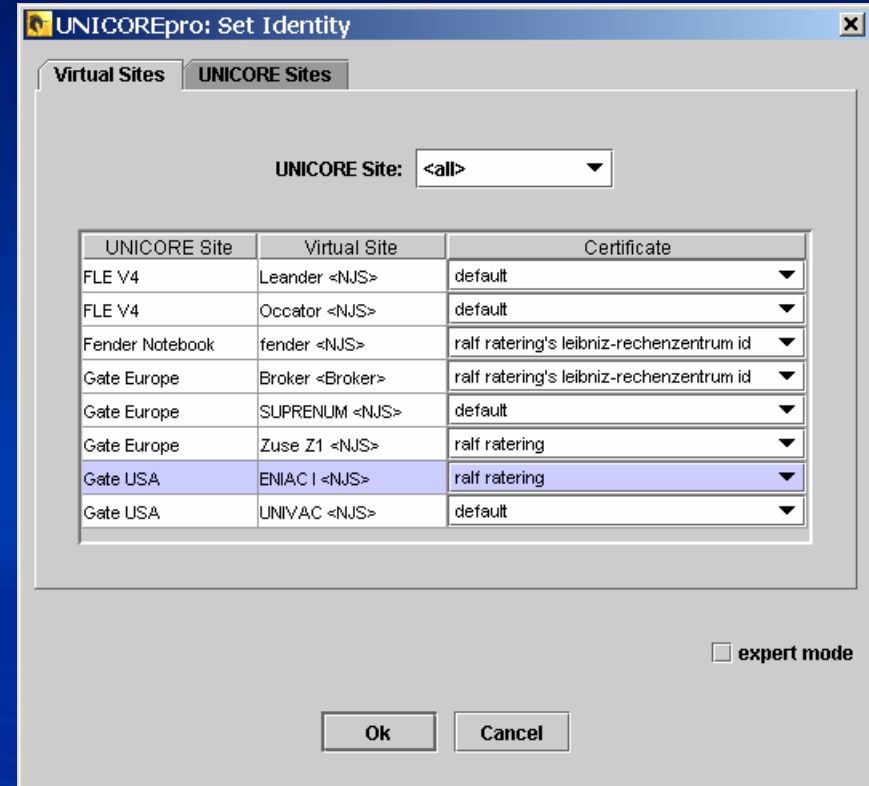
Ok Cancel



Configuration: Using Different Identities



Key entries: Who am I?

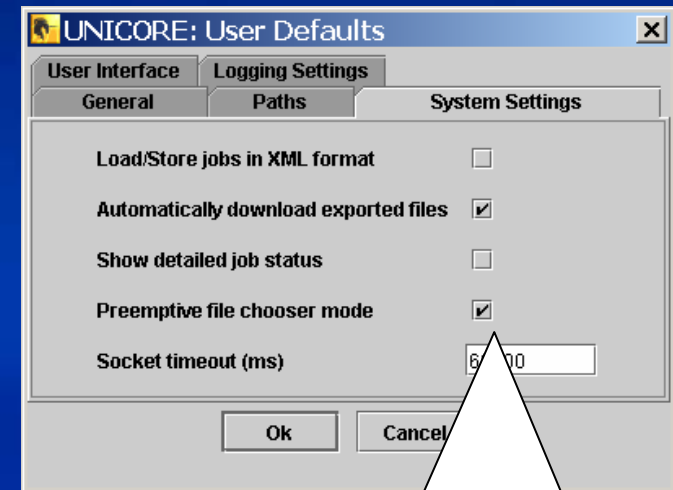
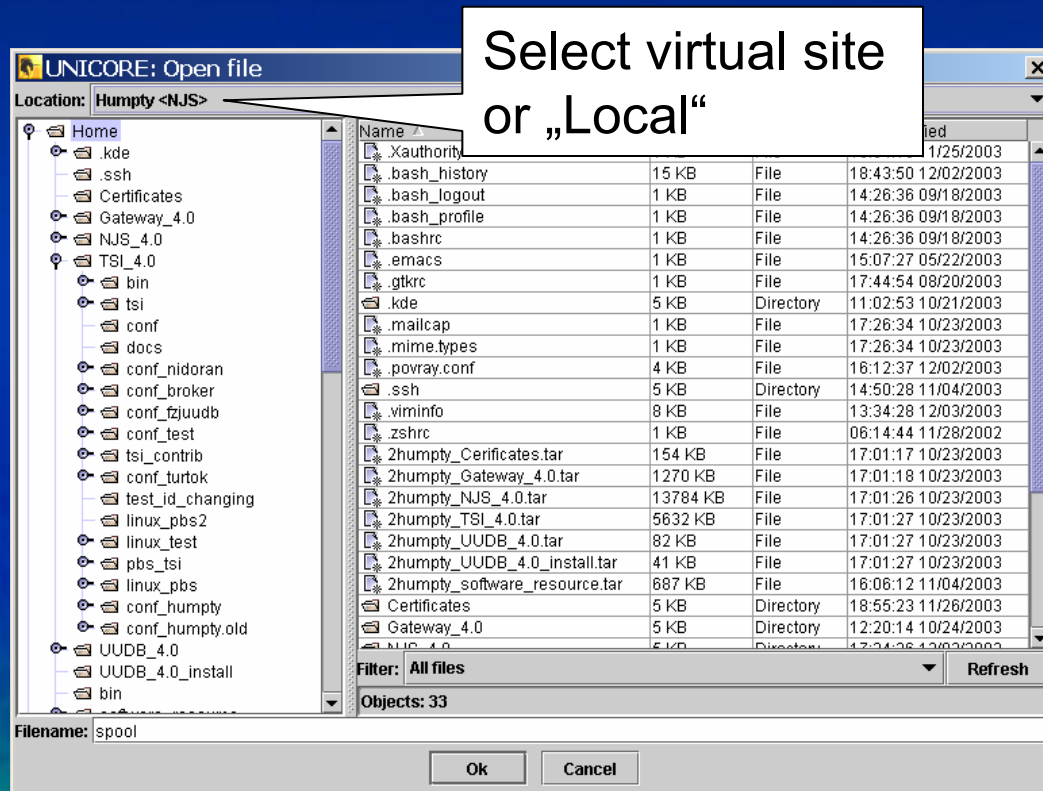


Using different identities

Browsing Remote File Systems

- **Remote File Chooser**

- Used in Script Task, Command Task, for File Imports, Exports, etc.



The Client Log

- „clientlog.txt“ or „clientlog.xml“
- Used by developers to figure out problems

User Defaults->Paths:

Log directory:

User Defaults->Logging Settings:

Logging Level: **INFO** **INFO should be fine**

ALL
FINEST
FINER
FINE
CONFIG
INFO
WARNING
SEVERE

Logging Format: **PLAIN** **Use PLAIN**

XML
PLAIN
BOTH

Send client log to file:

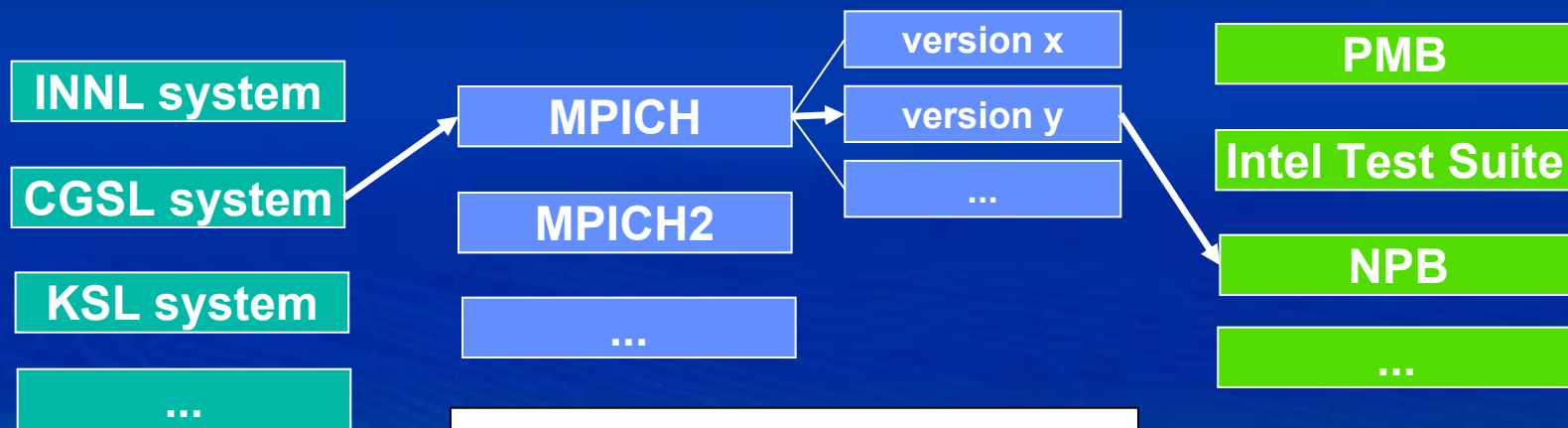
Enable under Windows,
when no console is used

Starting the client re-visited

- client.jar in lib directory
 - start with .exe (Windows) or run script (Unix/Linux)
 - or: „`java -jar client.jar`“
- Command line options
 - Choose an alternative configuration directory:
 - `-Dcom.pallas.unicore.configpath=<mypath>`
 - Enable the security manager:
 - `-Dcom.pallas.unicore.security.manager`
 - Enable SOCKS proxy:
 - `-DsocksProxyHost="socks-proxy.isw.intel.com"`
 - `-DsocksProxyPort="1080"`

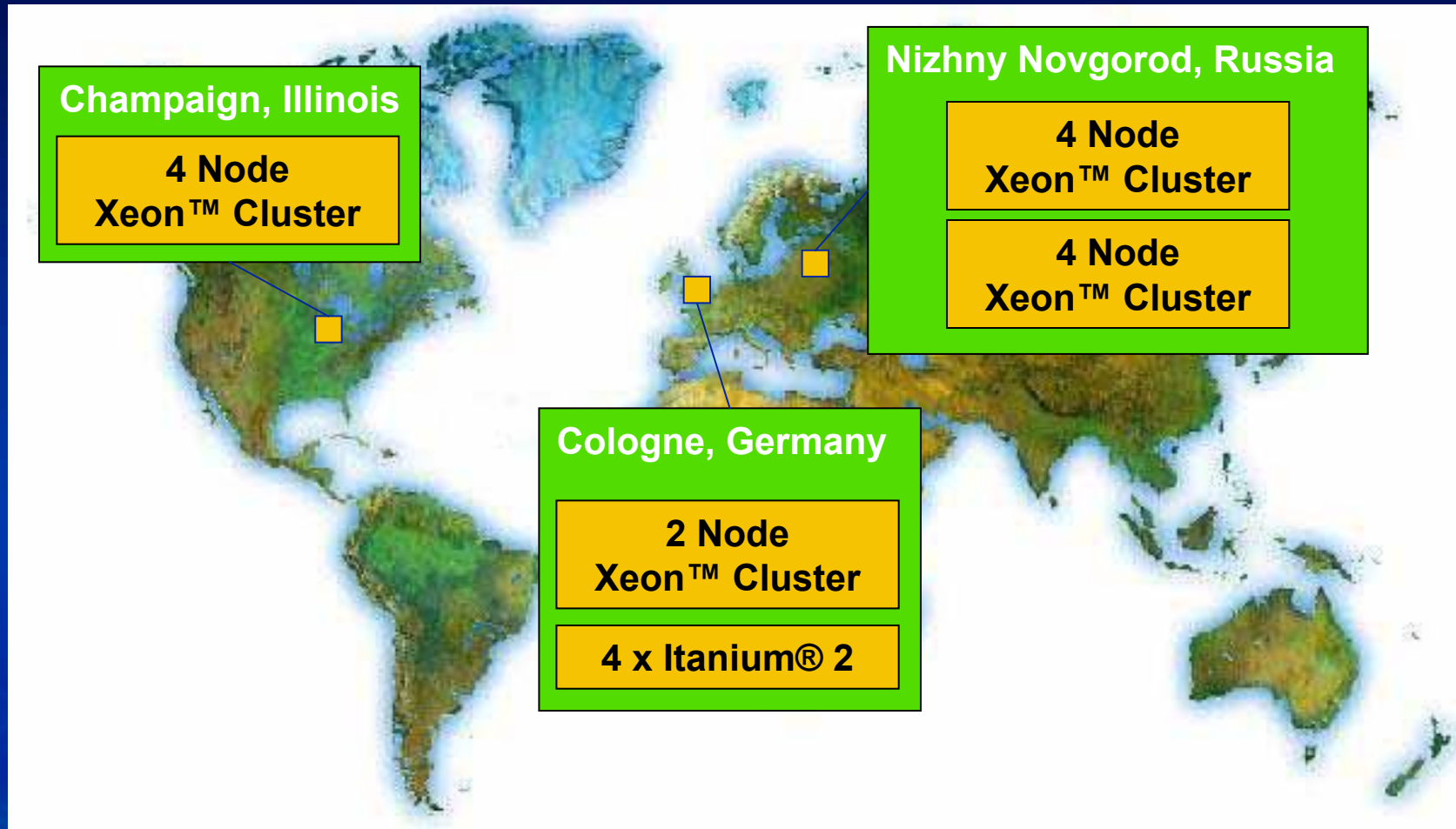
A real world Enterprise application: UNICORE inside Intel

- Software testing at Parallel and Distributed Solutions Division (PDSD)
 - Windows TSI port on server side
 - Complex existing testing environment



1. **Build with parameters**
2. **Run with parameters**
3. **Get result files**

Intel PDSD Grid



- UNICORE makes testing different versions on distributed systems a lot easier

Lessons learned...

- Security is negligible within intranet
 - Systems are protected by firewall
- Firewalls in the Intranet are a problem
 - Administrators have to open ports for every new NJS to the Gateways
- Users come and go
 - Managing user database and logins too complex
- Solutions
 - Open port range in firewalls
 - All testers use the same user certificate!!!

Summary

- Intel UNICORE Client offers an intuitive user interface to UNICORE Grids
- Client can be downloaded as Open Source at unicore.sourceforge.net
- Client functionality can be extended through plug-in interface