



The UNICORE Command Line CLient

Rebecca Breu

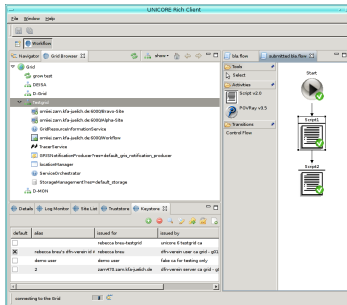
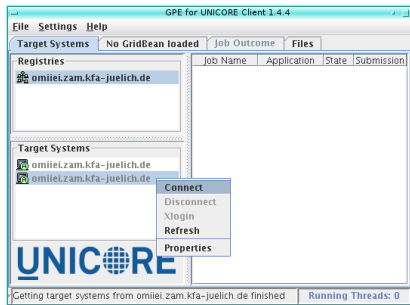
Jülich Supercomputing Centre
Forschungszentrum Jülich GmbH

08.07.2008

UNICORE Clients

Several clients for different purposes:

- ▶ **GPE Application Client:** Easy-to-use graphical client for simple tasks
- ▶ **Rich Client:** Eclipse-based graphical client for complex tasks
- ▶ **Portal Client:** Web based client to use with a browser
- ▶ **UCC:** Command line client



UNICORE Clients

Clients are ...

- ▶ intuitive
- ▶ Java based → platform independent
- ▶ easy to install

Prerequisites for users:

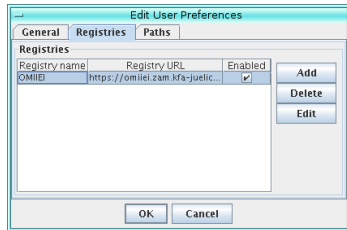
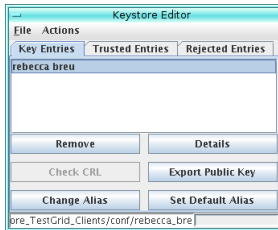
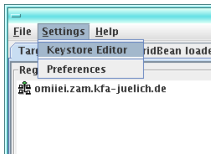
- ▶ Sun Java 1.5
- ▶ A user certificate

Outline

- ▶ This session: Single jobs via ucc
- ▶ Next session: Workflow jobs via Eclipse Client

Basic Client Configuration

- ▶ Create a password-protected **keystore** that holds all keys needed
- ▶ Add keys to keystore:
 - ▶ User's private and public key
 - ▶ Public keys of **trusted CAs** (must trust signer of involved gateway keys!)
- ▶ Add URI of **global registry**



Basic usage of UCC

Connect to UNICORE site:

```
> ucc connect  
You can access 1 target system(s).
```

For every ucc command, you can use -v for verbose output:

```
> ucc connect -v
```

Getting help:

```
> ucc help
```

Help for each ucc command with -h:

```
> ucc connect -h
```

Synchronous Job Submission

Use `run` command to submit a job:

```
> ucc run ucc-1.1/samples/date.u
RUNNING
SUCCESSFUL exit code: 0
/tmp/c61e1d7c-92d4-4be0-87f8-6a7dddc4c728.stdout
/tmp/c61e1d7c-92d4-4be0-87f8-6a7dddc4c728.stderr
/tmp/c61e1d7c-92d4-4be0-87f8-6a7dddc4c728.properties
```

- ▶ UCC waits until job is finished
- ▶ Output files are fetched automatically

Use `-a` option for asynchronous job submission → UCC returns prompt after submit

UCC job format: simple example

UCC jobs are given in **JSON** format.

Simple example: All-time favourite date application:

```
# Simple Date job
{
  ApplicationName: Date,
  Version: 1.0 # Optional
}
```

Date takes no input and produces stdout and stderr.

UCC job format: file imports and export

Example: Script job that produces additional file output.txt:

```
{  
  ApplicationName: Bash shell,  
  Environment: ["SOURCE=input.sh"],  
  Imports: [ {File: "/home/rbreu/myscript.sh",  
             To: input.sh} ],  
  # If script produces output (optional):  
  Exports: [ {File: output.txt,  
             To: output.txt} ]  
}
```

- ▶ File import from local machine to job's working directory
- ▶ File export from job's working directory to local machine

Specify Resources

```
{
  ApplicationName: Date,

  Resources: {
    Memory: 268435456, # per CPU in bytes
    Runtime: 86400, # per CPU in seconds
    Nodes: 3,
    CPUs: 64, # per node

    #Custom resources (site-dependent!):
    StackLimitPerThread : 262144
  }
}
```

Please be sparing if you try that here at ISSGC ;-)

Data management

List a remote directory:

```
> ucc ls u6://GILDA-CATANIA/home
.ssh
.bash_history
mydata
```

Copy files between local and remote machine:

```
> ucc get-file -s u6://GILDA-CATANIA/home/mydata \
-t mydata
> ucc put-file -s mydata2 \
-t u6://GILDA-CATANIA/home/mydata2
```

Batch Processing

Use `batch` command to submit all files within a directory:

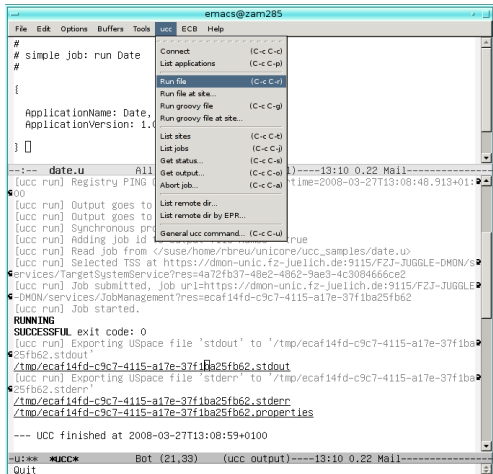
```
> ucc batch -i /my/dir/
```

- ▶ UCC returns when all jobs are finished
- ▶ Output files are fetched automatically

With `-f` option, `ucc` waits for more files to be copied in that directory:

```
> ucc batch -f -i /my/dir/
```

UCC from within Emacs



The screenshot shows the Emacs editor window titled 'emacs@zam285'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'ucc', 'ECB', and 'Help'. The 'ucc' menu is open, displaying the following options:

- Connect (C-c C-c)
- List applications (C-c C-p)
- Run file (C-c C-r)
- Run file at site...
- Run groovy file (C-c C-g)
- Run groovy file at site...
- List sites (C-c C-t)
- List jobs (C-c C-j)
- Get status... (C-c C-s)
- Get output... (C-c C-o)
- Abort job... (C-c C-a)
- List remote dr...
- List remote dr by EPR...
- General ucc command... (C-c C-u)

The main window contains a UCC script and its output. The script defines a job named 'date.u' with the following content:

```
# simple job: run Date
#
{
  ApplicationName: Date,
  ApplicationVersion: 1.0
} []
```

The output shows the job being executed successfully. Key lines include:

- Registry PING
- Output goes to
- Synchronous pro
- Adding job id
- Read job from </susse/home/rbreu/unicore/ucc_samples/date.u>
- Selected TSS at https://dmon-unic.fz-juelich.de:9115/FZJ-JUGGLE-DMON/s
- Job submitted, Job url=https://dmon-unic.fz-juelich.de:9115/FZJ-JUGGLE
- Job started.
- Exporting USpace file 'stdout' to /tmp/ecaf14fd-c9c7-4115-a17e-37f1ba25fb62.stdout
- Exporting USpace file 'stderr' to /tmp/ecaf14fd-c9c7-4115-a17e-37f1ba25fb62.stderr
- Exporting USpace file 'properties' to /tmp/ecaf14fd-c9c7-4115-a17e-37f1ba25fb62.properties
- UCC finished at 2008-03-27T13:08:59+0100

- ▶ UCC commands in Emacs
- ▶ Output highlighting
- ▶ Clickable links in output

See ucc-1.1.1/extras/

Advanced usage: Groovy scripting

Use groovy scripting to automatetasks, implement custom commands, ...

Simple example: Delete all your jobs

```
import de.fzj.unicore.uas.client.*;

def lister =
    new de.fzj.unicore.ucc.helpers.TargetSystemLister(registry)

lister.each {
    it.jobs.each{
        messageWriter.message "Job at "+it.address.stringValue
        new JobClient(it.address.stringValue, it,
                      securityProperties).destroy()
    }
}
```

Run with `ucc run-groovy -f delete.groovy`

Advanced usage: Groovy scripting

Predefined variables to use in Groovy scripts:

variable	description
registry	A preconfigured client for accessing the registry
securityProperties	Security configuration (keystore, etc)
registryURL	the URL of the registry
messageWriter	for writing messages to the user
commandLine	the command line
properties	defaults from the user's properties file

More information

- ▶ Information for the practical sessions:
www.fz-juelich.de/jsc/unicore/ISSGC08/
- ▶ UNICORE Website: www.unicore.eu/
- ▶ UCC Documentation:
www.unicore.eu/documentation/manuals/unicore6/ucc/
- ▶ Mailing list: unicore-support@lists.sf.net

More Groovy examples can be found in the ucc package.