Integration of UNICORE6 at CEA

Xavier Delaruelle (CEA/DAM)
xavier.delaruelle@cea.fr

UNICORE and Supercomputing Workshop 2009
Deutscher Wetterdienst, Offenbach
March 18, 2009
Outline

● CEA Computing Complex
● Fulfill user's needs with UNICORE6
● Making UNICORE6 fit CCRT
● Plans and conclusion
CEA Computing Complex
Computing complex location

- Hosted by CEA/DIF center
- At Bruyères-le-Châtel, South of Paris
3 Computing Centres

- **TERA**
  - 64 Tflops (2005), Classified production
  - For CEA/DAM researchers

- **CCRT**
  - 52 Tflops (2007)
  - For CEA divisions, GENCI and industrial partners (EDF, ONERA, SAFRAN, ...)

- **OCRE**
  - Open research and development
  - Testbed for PRACE, DEISA and Ter@tec industrial partners
CCRT current architecture

BULL cluster
47.7 Tflops
Memory 23.2 To
Disks 420 To

6 * 10Gb/s Links

CCRT backbone
10 Gb/s

1 Gb/s

users

NFS servers

NEC Machines
2 Tflops
Memory 0.9 To
Disks 40 To

8 * 1Gb/s links

8 * 10Gb/s links

2*SGI Altix 450
+ DMF

Level 1: 1 PB
Sata DDN disks

Level 2: SUN
(9940, T10000)

Storage system
CCRT site update

- **Renewing architecture (since October 2008)**
  - Links with other French computing sites
  - CEA became associate partner of DEISA

- **New hardware**
  - Post-processing cluster
    - 38 HP graphical nodes
    - 100TB Lustre filesystem
  - Vector machine
    - Adding 3 Nec SX9 nodes
  - Nehalem-EP thin nodes cluster
    - 1068 Bull NoveScale R422 compute nodes (~100 Tflops)
    - 48 GPGPU Nvidia Tesla compute nodes (192 Tflops single precision)
    - 500TB Lustre filesystem with 20 GB/s of bandwidth
Fulfill user's needs with UNICORE6
CCRT user's needs

- Common computing needs
  - Submit jobs
  - Manage and transfer their data

- The way they want to do it
  - From their remote location
  - Be able to automate these actions with their own scripts to re-submit/re-transfer in case of failure
Matching UNICORE6 features

• Jobs and data are the middleware core
  ■ Job Management Service
  ■ Storage Management Service
  ■ Various file transfer mechanisms (OGSA-ByteIO, HTTP, etc)

• Remote management
  ■ Graphical and command-line clients to access the middleware
  ■ HiLA client API to program UNICORE applications
Making UNICORE6 fit CCRT
Batch scheduler candidate for CEA petaflop system

- Open Source solution
- Fit our scheduling needs

As of April 2008

- No TSI for SLURM publicly available
- Initial TSI SLURM developed at BSC

Initial SLURM TSI was

- Specific to BSC and DEISA environment
- Based of UNICORE5 TSI structure
● Work done
  ▪ Remove specific part from BSC's TSI
  ▪ Make it fit new TSI code architecture
  ▪ Submitted to UNICORE-devel
  Part of the UNICORE package since 6.1.2 (August 08)

● TSI development experience
  ▪ Seemed hard at first sight
    - More than 10 files in initial implementation
  ▪ It was easy in the end
    - Lot of code was dropped thanks to TSI SHARED
    - Resulting TSI is composed of 3 short files
    - TSI SLURM only defines how to call SLURM binaries, how to submit jobs and how to parse job status
• CEA users need Web proxy to access Internet resources

• Tests made
  - Check the Web proxy feature of all the clients
  - Bugs spotted and reported
Web proxy support (2/2)

- **Current situation**
  - UCC directly talks to the proxy in HTTPS
  - Whereas it expects HTTP Connect

- **Ongoing work**
  - How to handle “Connect” method with the HTTP Client library UNICORE is relying on
We like having every piece of software well integrated in our systems

- Our servers run Red Hat-like Linux system
- Our sysadmins expect using:
  - RPM to install or upgrade UNICORE components
  - syslog to manage log data
- Our sysadmins expect finding:
  - start/stop/status scripts in /etc/init.d
  - configuration files under /etc

As of March 2009

- No RPM package available
- Quickstart bundle not shipped with Red Hat-like service scripts
Linux system integration (2/2)

- **Planned work**
  - Develop one RPM package per component
  - With files stored in their regular paths
  - log4j configured to send log data to Syslog
  - No default configuration but examples in `/usr/share`

- **Make packaging work persistent**
  - Create RPM specfiles is good but if they are disconnected from the project management tool they will become useless in no time
  - Having specfiles and RPM built integrated in Maven's pom.xml files will make the work last longer
LDAP centralizes our users definition

- Our users are registered in a LDAP tree
- Their LDAP entry contains their certificate's DN
- And their grid authorization attributes

XUUDB is built from the LDAP

- No information are specific of the XUUDB
- For us, this is an extra component to manage
LDAP instead of XUUDB (2/2)

- **UNICORE/X to talk to LDAP instead**
  - Querying the LDAP instead of the XUUDB
  - No more need of XUUDB in our installation

- **Planned work**
  - Develop a LDAPAuthoriser class
  - Make LDAP fields configuration possible
High Availability (1/2)

• Complex multi-component middleware
  - If one component goes down, the whole service could be broken

• As of March 2009
  - No High Availability support in UNICORE
  - AJP13 support for Gateway and MySQL backend for persistence are good starting points
High Availability (2/2)

- Planned research
  - Create and assess a HA testbed by performing components redundancies
• Plans for Kerberos deployment
  - We plan to have Kerberos supported in our batch scheduler and our shared file system
  - We would like UNICORE to submit Kerberized jobs
  - So UNICORE has to get a Kerberos ticket for users

• How to get a Kerberos ticket
  - User's identity has to be submitted to Kerberos KDC in a format it understands
  - PKINIT Kerberos extension (RFC 4556) makes KDC understand X.509 certificate
  - UNICORE is currently able to forward proxy certificate the job working directory
**Planned research**

- Make UNICORE get a Kerberos ticket from KDC
- By presenting it, using PKINIT extension, user's proxy certificate
Plans and conclusion
Plans

• Summer 2009
  - Linux system integration
  - LDAP binding

• Autumn 2009
  - High availability support (*first results*)
  - PKINIT support (*first results*)
Conclusion

• UNICORE fits our needs
  - Most of our needs are available out of the box
  - For the rest, UNICORE is easy to adapt to make it fit our environment

• UNICORE is a good grid investment for us
  - Well spread middleware
  - With active development and support community
Questions?