Introduction to CREAM

Massimo Sgaravatto

INFN Padova

On behalf of the Padova CREAM group
- CREAM service: Computing Resource Execution And Management service
- Service for job management operations at the Computing Element (CE) level
- Allows to submit, cancel, monitor, … jobs
- Web service interface
- Implemented and maintained by the Grid middleware Padova Group
CREAM usage scenario

- CREAM can be used:
  - through the gLite WMS
  - by a generic client willing to interact directly with the CE
    - We provide and maintain an “official” CREAM CLI
      - Very similar to the WMS CLI
    - Users can build their own clients using a Web Service framework
• **Job submission**
  - Submission of jobs to a CREAM based CE
  - Supported job types: normal (sequential batch jobs), MPI (guidelines of MPI-WG group were followed), sub-jobs of collection/parametric jobs submitted through the WMS
    ▪ So basically everything but sub-jobs of real DAGs
  - Job characteristics described via a JDL (Job Description Language) expression
    ▪ CREAM JDL is the same (a subset) JDL used by the Glite WMS

> `glite-ce-job-submit -a -r grid005.pd.infn.it:8443/cream-lsf-grid02 myjob1.jdl`

https://grid005.pd.infn.it/CREAM152328764
CREAM functionality

- **Job status**
  - To get status and other info (e.g. significant timestamps, worker node, failure reason, issued commands on the job, etc.) of submitted jobs
  - Different levels of verbosity
  - Also possible to apply filters on submission time and/or job status

```plaintext
> glite-ce-job-status -L 1 https://cream-02.pd.infn.it:8443/CREAM955790315

******

JobID=[https://cream-02.pd.infn.it:8443/CREAM955790315]
Current Status = [REALLY-RUNNING]
Grid JobID   = [N/A]

Job status changes:
-------------------
Status         = [REGISTERED] - [Fri 21 Dec 2007 17:49:38] (1198255778)
Status         = [PENDING] - [Fri 21 Dec 2007 17:49:38] (1198255778)
Status         = [REALLY-RUNNING] - [Fri 21 Dec 2007 17:50:04] (1198255804)

Issued Commands:
-------------------
*** Command Name = [JOB_REGISTER]
Command Category = [JOB_MANAGEMENT]
Command Status   = [SUCCESSFULL]

*** Command Name = [JOB_START]
Command Category = [JOB_MANAGEMENT]
Command Status   = [SUCCESSFULL]
```
**CREAM functionality**

- **Proxy delegation**
  - To delegate a proxy, which can be used by the job to do operations requiring security support (e.g. GridFTP file transfers)
  - Possibility to automatically delegate a proxy for each job submission
    ```
    > glite-ce-job-submit -a -r grid005.pd.infn.it:8443/cream-lsf-grid02 myjob.jdl
    ```
  - Possibility to delegate a proxy, and then using it for multiple job submissions
    ```
    > glite-ce-delegate-proxy -e grid005.pd.infn.it:8443 mydelid
    > glite-ce-job-submit -D mydelid -r grid005.pd.infn.it:8443/cream-lsf-grid02 myjob.jdl
    ```
  - Recommended approach wrt performance, since proxy delegation can be “expensive”
  - For submissions done via WMS, the proxy is delegated only when needed (i.e. only if the “same” proxy has not been delegated yet)

- **Job cancellation**
  - To cancel previously submitted jobs
    ```
    glite-ce-job-cancel https://cream-02.pd.infn.it:8443/CREAM955790315
    glite-ce-job-cancel -a -e cream-02.pd.infn.it:8443
    ```

- **Job list**
  - To get the identifiers of all your jobs submitted on a specific CREAM CE
    ```
    glite-ce-job-list cream-02.pd.infn.it:8443
    ```
CREAM functionality

- **Proxy renewal**
  - To renew proxies for previously submitted jobs
    > glite-ce-proxy-renew --e grid005.pd.infn.it:8443 mydelid
    All jobs with mydelid as delegationid will have their proxy renewed
  - For jobs submitted to CREAM via the WMS, proxy renewal is done automatically, if it has been enabled

- **Job suspension and job resume**
  - To hold and then restart jobs
    > glite-ce-job-suspend https://cream-02.pd.infn.it:8443/CREAM955790315
    > glite-ce-job-resume https://cream-02.pd.infn.it:8443/CREAM955790315

- **Job purge**
  - To clear jobs from CREAM based CEs
    > glite-ce-job-purge https://cream-02.pd.infn.it:8443/CREAM955790315
    Then it will not be possible to “manage” anymore that job
CREAM functionality

- **Disable/enable new job submissions**
  - Can be used only by CREAM CE administrators
  - Useful for example for a scheduled shutdown of the CREAM CE
    ```
    > glite-ce-disable-submission grid005.pd.infn.it:8443
    > glite-ce-job-submit -a -r grid005.pd.infn.it:8443/cream-lsf-grid02 test.jdl
    > glite-ce-enable-submission grid005.pd.infn.it:8443
    ```
  - When submissions are disabled the other commands are still allowed
  - Submissions can be automatically disabled also when a certain condition (on the number of pending and/or idle and/or running jobs) specified in the CREAM conf file is met
    - E.g. a site administrator can decide to stop accepting new jobs when the site is already managing x jobs
- **Check if submissions are enabled**
  ```
  > glite-ce-allowed-submission grid005.pd.infn.it:8443
  Job Submission to this CREAM CE is disabled
  ```
CREAM architecture

• Runs as a Java-Axis servlet on Tomcat application server
• Web service interface
  – WS-I compliance
• Information related to jobs being managed by CREAM saved in the CREAM backend
  – Implemented via a RDBMS (Mysql)
• Interacts with CEMon (CREAM-JOB sensor)
  – CEMon: Web service responsible to collect and provide information, in synchronous or asynchronous mode
  – One or more sensors can be plugged into CEMon
    ▪ Each one of this sensor is responsible to "manage" a specific type of information
      • CE_MONITOR sensor: CE information
      • OSG_CE sensor: CE information, suitable for OSG needs
      • CREAM-JOB sensor: CREAM job information
      • GridICE sensor: GridICE monitoring information
  – Also part of VDT
“Simplified” view of CREAM architecture

- **Job submit (job register + job start), job cancel, etc.**
- **AuthN / AuthZ**
- **CREAM Interface**
- **Command Manager**
- **CEMon**
- **Notifications about CREAM job status changes**
- **Queue of async requests**
- **Async. commands (start, cancel, etc.)**
- **Sync. commands (register, status, etc.)**
- **Job Executor**
- **BLAH**
- **LRMS (PBS, LSF, ...)**
- **Glexec**

**CREAM JOB plugin**
Security

- **Authentication**
  - Implemented via trustmanager

- **Authorization**
  - Implemented via gJAF (Grid Java Authorization framework)
  - VOMS and gridmap PDPs
    - Possibility to enable VOs and/or single DN
  - gJAF not supported anymore by JRA1 security cluster in EGEE-III
    - We will maintain it
  - New authorization service being planned and implemented by JRA1 security people
  - A user can manage (e.g. cancel, monitor) only her jobs
    - Only CE admins can manage also jobs submitted by other users
    - CE admins (specified with their DN) defined in a specific file on the CREAM CE

- **Credential mapping**
  - To map Grid credentials on local accounts and execute commands on behalf of these local accounts
  - Implemented via glexec
    - Glexec uses LCMAPS and LCAS
The job submitted to the underlying batch system (via BLAH) is actually a job wrapper, very similar to the one considered in the submission to LCG-CEs.

- Besides running the user job (the one specified as *Executable* in the JDL), it is responsible for transferring the sandboxes, for logging to LB, etc.
- It also notifies CREAM about some job status changes:
  - Running, Really-Running, Done

In the submission to the LCG CE the job wrapper is created on the WMS:
- By the JobAdapter (Helper of WM)

In the CREAM CE the job wrapper is instead created on the CE by CREAM.
Support for ‘scattered’ sandboxes, as in the WMproxy
  - Input Sandbox files can be downloaded (by the job wrapper running on the WN) from several GridFTP/HTTPS servers

For staging input files from the client machine, CREAM foresees the deployment of a GridFTP Server (Icas/Icmaps aware) on the CREAM CE node

This is needed ONLY when submitting directly to CREAM and ONLY when files have to be staged from the client machine
  - E.g. not needed if files have been pre-staged on GridFTP/HTTPS servers
  - E.g. not needed for submissions to CREAM via the WMS, since the ISB files are pre-staged from the UI node into the GridFTP server running on the WMS node
The interaction with the underlying local resource management system (LRMS) is fully managed by BLAH

- Implemented and maintained by INFN Milano group

BLAH used to submit, cancel, etc. jobs on the batch system

BLAH also used, via the BLParser, to notify CREAM about job status changes

- Actually CREAM knows about (some) job status changes also from the job wrapper running on the Worker Node

Two BLParser implementation models:

- Old one: works parsing the batch system log files
- New one: works referring to the batch system status/history commands
  - New model done also to facilitate the porting to new batch systems
• Currently supported batch systems in BLAH and therefore in CREAM:
  1. LSF: using old BLparser model
     ▪ Migration to new parser will be done
  2. Torque/PBS: using old BLParser model
     ▪ Migration to new parser will be done
  3. Condor: using new BLParser model
     ▪ Submission to CREAM based CE using Condor as resource management system has been proved (also via WMS), but not too many tests done
     ▪ More tests will be done by PIC people

• No changes needed in CREAM when moving from old BLParser implementation to new one
• Besides the legacy interface, CREAM exposes also a BES-compliant interface

• BES (Basic Execution Service): recent OGF specification for a standard interface for Grid execution services
  – Aim: favor interoperability between different Grids

• BES defines basic operations for job submission and management
  – BES itself does not mandate any specific security implementation
    ▪ E.g. proxy delegation is not part of the BES specification

• JSDL (Job Submission Description Language) used in BES to describe computational jobs
• **Legacy interface**
  - JobRegister
  - JobStart
  - JobCancel
  - JobList
  - JobLease
  - JobInfo
  - JobPurge
  - JobSignal
  - JobSuspend / JobResume
  - JobProxyRenew
  - GetInfo
  - GetCEMonURL
  - EnableAcceptJobSubmissions
  - DisableAcceptJobSubmissions
  - DoesAcceptJobSubmissions

• **BES Interface**
  - CreateActivity
  - TerminateActivities
  - GetActivityStatuses
  - GetActivityDocuments
  - GetFactoryAttributesDocument
  - StopAcceptingNewActivities
  - StartAcceptingNewActivities
Compliance to standards

• Standards addressed
  – Basic Execution Service (BES) v. 1.0
  – Job Submission Specification Language (JSDL) v. 1.0
  – HPC Basic Profile v. 1.0

• Interoperability between different Grid services with a BES compliant interface demonstrated at SC07
  – Other actors participating in this demo: UVA .NET, Microsoft HPC Group, UNICORE, GRIDSAM, Nordugrid/KnowARC, Platform Computing

• These BES related activities done so far in the domain of the OMII-EU project
  – BES support done on an old CREAM implementation
  – Work in progress to finalize BES support in the current CREAM implementation
WMS-CREAM integration

- WMS-CREAM integration implemented via ICE (Interface to CREAM Environment)
- Daemon running on the WMS node
- Basically has the role played by JC+LM+Condor in the submission to LCG CEs
- ICE takes the job management requests from its filelist and satisfies them
- ICE also monitors jobs submitted to CREAM CEs and take appropriate actions
ICE: detecting CREAM job status changes

- CEMon with CREAM-JOB plug-in coupled with CREAM
- ICE subscribes to CEMon to be informed about CREAM job status changes
- ICE receives these notifications from CEMon, detect CREAM job status changes and take the appropriate actions
- As a fail-safe mechanism, ICE also able to poll CREAM if the relevant notifications are not received via CEMon
ICE: some details

• Implemented in C++
• Multithreaded
• Uses cream-client-api-c
  – Use of gSOAP library
  – Used also by CREAM CLI
• Persistent saving of vital information
  – Berkeley DB
• Reliable lease based mechanisms in the submission protocol
  – To handle failure scenarios and avoid to leave “unmanaged” jobs (zombies)
  – General idea
    ▪ Each job has an attribute (the lease) which is basically the time to live of the job
    ▪ Leases are renewed by ICE as long as ICE and CREAM can talk to each other
    ▪ When the lease expires, the job is removed on both sides: CREAM and WMS
ICE tries to renew lease

CREAM crashes

Lease expires, job is removed from WMS

Cream restarts sees lease expired, purges jobs
• No differences wrt LCG-CE
  – Events logged by the job wrapper running on the Worker Node
    ▪ Running, Really-Running, Done events (Source = LRMS)
    ▪ Forwarded to LB server via the LB interlogger running on the CE node
  – Events logged by ICE
    ▪ The same logged by JC+LM when dealing with submissions to LCG-CEs
    ▪ Actually they appear with JobController / LogMonitor as Source
    ▪ You’ll never see ICE as Source in the LB events

  Event: DeQueued
  - Source = JobController
  - Timestamp = Mon May 12 15:01:39 2008 CEST

  Event: Transfer
  - Destination = LRMS
  - Result = OK
  - Source = LogMonitor
  - Timestamp = Mon May 12 15:01:55 2008 CEST
• No differences wrt LCG CE for what concerns the interaction with the Information Service
  – CREAM CE characteristics and status published in the BDII, according to Glue Schema
  – Same information providers used in the LCG CE
  – Only some differences in the configuration of the static ldif file (e.g. CEID format) wrt the LCG CE

• Querying the BDII, it is possible to check what is “flavour” of a specific CE checking the GlueCEImplementationName attribute
  – GlueCEImplementationName: CREAM
  – Can be used for matching CREAM CEs

• It is also possible to refer to the GlueCEUniqueID:
  – CREAM-CE → <host>:<port>/cream-<lrms>-<queue>
  – LCG-CE → <host>:<port>/jobmanager-lcg<lrms>-<queue>
Deployment scenario

• **A same Worker Node can be shared between LCG CEs and CREAM CEs**
  - No specific requirements on the WN imposed by CREAM

• **Possible deployment scenario**
  - A farm of WNs with one (or more) CREAM CE front-end and one (or more) LCG CE front-end
  - Some batch system queues used by CREAM CEs, and some other queues used by LCG CEs
    - Even if a same batch system queue could be shared between a LCG CE and a CREAM CE
Reliability tests

• Done in the context of the “Acceptance tests”
• Test 1 (without proxy renewal) results:
  – >8 days unattended running
  – ~90K jobs submitted via gLite WMS by 40 users
  – About ~5k jobs always in the batch system queue
  – No errors due to CREAM
  – No performance degradation observed
• Test 2 (with proxy renewal) results:
  – Jobs lasting 3-4 hours
  – Initial proxy valid 2 hours
  – About ~5k jobs always in the batch system queue
  – 5 days of unattended running
  – 64000 jobs submitted by 40 users
  – About 0.5% of failures, mainly due to “proxy expired”
    • Problem (with proxy renewal in BLAH) has then been understood and fixed
• Problems redoing tests with proxy renewal because problems with proxy renewal service
  – From time to time it doesn’t work
• **How the tests have been performed**
  - ICE turned OFF
  - Submission of 1000 jobs by 1 user (single delegation) to the WMS
  - When all the requests have been inserted in the ICE filelist, ICE turned ON

• **How the measurements have been performed**
  - $T_{\text{start}}$ = LB timestamp of first ICE dequeued event (i.e. request removed from the filelist, i.e. ICE started its work)
  - $T_{\text{stop}}$ = timestamp of last submission to batch system event in the BLAH accounting log file

• **Throughput to submit to LRMS** = $\#$ jobs / ($T_{\text{stop}}$ - $T_{\text{start}}$)
CREAM - ICE throughput test results

- Submission of 1000 jobs by 1 user (1 proxy delegation done to the WMS)
- UI at INFN-CNAF
- WMS ICE enabled at INFN-CNAF
  - ICE configured with 10 threads
- CREAM CE at INFN-PADOVA

<table>
<thead>
<tr>
<th>Try No.</th>
<th>% success</th>
<th>Throughput to LRMS (jobs/min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100 %</td>
<td>65.08</td>
</tr>
<tr>
<td>2</td>
<td>100 %</td>
<td>69.77</td>
</tr>
</tbody>
</table>
CREAM installation & configuration

- **YAIM based installation procedure available**
  - Works for PBS/Torque and LSF as LRMS
  - Support of Condor in yaim installation procedure to be done

- **Manual installation instructions available as well in the CREAM web site**

- **CREAM configuration**
  - Several CREAM attributes can be tuned editing the CREAM configuration file `opt/glite/etc/glite-ce-cream/cream-config.xml`
    - Even if the provided defaults are usually ok
    - The meaning of the configurable attributes explained in the CREAM web site
    - The most significant attributes configured via the yaim `siteinfo.def`
CREAM CLI and ICE installation & configuration

- **CREAM CLI installation**
  - Will be part of the UI node installation
  - CREAM CLI manual installation instructions available in the CREAM web site

- **ICE installation part of the WMS installation process**

- **ICE Configuration**
  - Specific ICE section in the WMS configuration file (/opt/glite/etc/glite_wms.conf)

```plaintext
[WorkloadManagerProxy = [
  ...
]
ICE = [
  logfile = "${GLITE_LOCATION_LOG}/ice.log";
  ...
]
...
]
```

- Meaning of the configurable attributes (and their default values) explained in the CREAM web site
• Available in the CREAM web site
  – http://grid.pd.infn.it/cream
• Release notes
• Administrator guides
  – Installation and configuration instructions for CREAM (manual and via yaim), CREAM-CLI and ICE
• User guides
  – CREAM User’s guide
  – CREAM JDL specification
• Info useful for troubleshooting
  – For users and admins
  – Log files to check
  – Some preliminary FAQs
    ▪ Relevant error messages and their meaning
    ▪ More work needed
• Papers and presentations
• Test results
  – Actually many on them done on old CREAM implementations
• CREAM ready for certification
  – Patch #1755: CREAM server side
  – Patch #1790: CREAM CLI
  – For what concerns the WMS, as agreed we made available a list of RPMs suitable to install a WMS ICE enabled, to test the submission to CREAM
    ▪ A WMS patch will be prepared as soon as some other WMS bugs (not related with ICE-CREAM) will be fixed

• Need to have a close collaboration with CREAM certifier (Di, GRNET) to make this process as efficient as possible
Some items for the future

- Clearly future CREAM related activities will be driven by the certification process
  - We will have to promptly address all the issues that will be found
- Clearly new functionality/improvements will have to be negotiated with the TMB
- At any rate we have already identified some items that should be addressed
  - Submission to CREAM by Condor
    - Some work already done with an “old” CREAM implementation
  - Submission of multiple jobs by the WMS to CREAM with a single call
    - E.g. if in the bulk matchmaking of a job collection n sub-jobs got matched to a certain CREAM CE, these n sub-jobs should be submitted all together to that CREAM
    - CREAM is already able to manage such scenario
    - Modifications are needed in the WMS
  - New development model for CREAM and WMS job wrapper
    - CREAM and WMS (the one used for LCG-CE) job wrappers have many common parts
    - Not good and dangerous to have duplicated code
Some items for the future

- **High availability/scalable CE**
  - CREAM CE front end and pool of CREAM machines doing the work
  - Main needed functionality already in place
    - Multiple CREAM CEs sharing the same backend (same DB)
    - E.g. a job can be submitted to a CREAM CE, and can then be cancelled on another CE
  - Still some issues to address

- **Better integration between CREAM and LB**
  - CREAM able to log information to LB
    - Enhance LB events with further information
  - Use of LB tools to monitor CREAM jobs
    - Also for the non WMS-jobs (i.e. the ones submitted directly to CREAM)
  - Discussions already started with CESNET people

- **CREAM used also to access a relational DB**
  - Requested by some G-DSE people
  - CREAM is already a general purpose command executor
  - So it is just a matter of implementing and plug an executor to access a RDBMS
    - This will be done in the context of e-NMR

- **Switch to new authorization service, when ready**
Specific WDL for job management i.e. CREAM, BES

WSDL job interf

WSDL cmd interf

WSDL db interf

Generic WDL for cmd execution

Specific WDL for DB management i.e. OGSA-DAI

async job cmd

job cmd queue DB

thread pool

AbsJobExecutor

BLAHExecutor

LRMS

DBExecutor

DB

async db cmd
db cmd queue DB

thread pool

INFSO-RI-508833
Contact us: jra1-pd@pd.infn.it

- Paolo Andreetto (EGEE-III SA3, formerly OMII-EUROPE and EGEE JRA1)
  - Internal release manager, Etics confs, CREAM pre-certification
- Sara Bertocco (EGEE-III SA3, formerly EGEE-II SA1)
  - CREAM integration, yaim, CREAM pre-certification
- Alvise Dorigo (EGEE-* JRA1)
  - CREAM and CEMon C clients, ICE
- Eric Frizziere (e-NMR, formerly Cyclops)
  - CREAM and CEMon, CREAM-GDSE integration
- Alessio Gianelle (EGEE-III SA3, formerly EGEE-II SA3 and EGEE JRA1)
  - CREAM pre-certification
- Moreno Marzolla (EGEE-III JRA1, formerly OMII-EUROPE and EGEE JRA1)
  - CREAM and CEMon C clients, ICE, CREAM-BES
- Massimo Sgaravatto (EGEE* JRA1)
  - Testing, overall coordination
- Luigi Zangrando (EGEE* JRA1)
  - CREAM and CEMon