Soumission de jobs

M. Jouvin (LAL-Orsay)

Tutorial Utilisateur Grille
MRM Grille Paris Sud, LAL
2 Juin 2010

Information Society and Media

Agenda

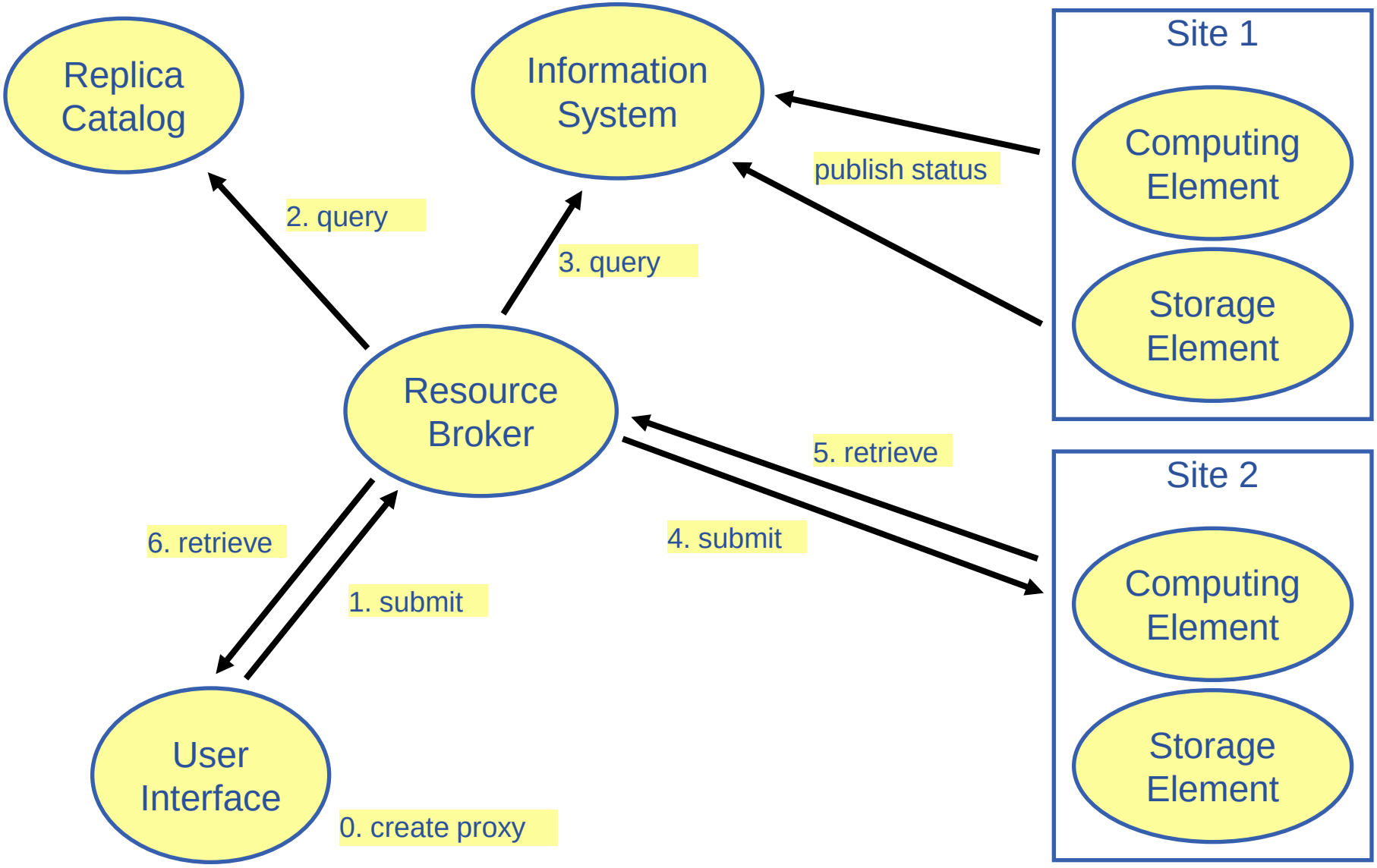
- Les différents composants de la gestion de jobs
- Les principales commandes
- Job Description Language (JDL)

Workload Management System

- **Essaie d'optimiser l'utilisation des ressources et d'exécuter les jobs des utilisateurs le plus rapidement possible**

- **Est composé des services suivants :**
  - *UI (User Interface)* : point d'accès pour les utilisateurs
  - *WMS* : le broker des ressources de la grille, responsable de trouver les « meilleures » ressources où soumettre les jobs.
    - *Anciennement appelé RB (Resource Broker)*
  - *LB (Logging and Bookeeping)* : stocke les infos concernant les états successifs.
  - *BDII (Information Index)* : un serveur LDAP qui collecte les informations concernant les ressources grille. Il est utilisé par le RB pour sélectionner les ressources

Soumission d'un job



L'Interface Utilisateur

- **UI : machine "en dehors" de la grille qui contient les outils pour interagir avec la grille**
  - Acquérir un proxy
  - Soumission et gestion des jobs
  - Transfert et gestion des données
- **Soumission de jobs**
  - Commandes : glite-wms-job-xxx
  - Commandes glite-job-xxx et edg-job-xxx ne sont plus supportées

Principales Commandes

- `glite-wms-job-submit [-d deleg_proxy|-a] fichier_jdl`
  Soumets un job
  Retourne le jobID
- `glite-wms-job-status [-v 1|2] jobID`
  Donne le statut du job
- `glite-wms-job-output jobID`
  Récupère les fichiers spécifiés dans l'attribut OutputSandbox
- `glite-wms-job-cancel jobID`
  Annule un job
- `Glite-wms-job-delegation-proxy –d identifier`
  Crée un *delegation proxy* à utiliser lors du submit
- `glite-wms-job-list-match fichier_jdl`
  Liste les ressources compatible avec la description du job
  Effectue le *matchmaking* sans soumettre le job
- `glite-wms-job-logging-info jobID`
  Donne des informations de *logging* sur les jobs soumis (tout les événements répertoriés par les divers composants du WMS)
- API C, C++ et Java disponibles pour toutes ces fonctions

La Soumission

- **Les commandes de soumission utilisent un fichier de description de job (JDL)**
  - On ne soumet pas directement son programme
  - L'application peut être préinstallée sur la grille
- **La soumission retourne un *job identifier***
  - Indispensable de le conserver pour pouvoir récupérer des informations et les résultats
  - Option –o permet de l'écrire dans un fichier
    - Utiliser l'option –i dans les autres commandes
    - Le fichier peut contenir une liste de jobid (pas écrasé à chaque fois)
- **Proxy delegation : nécessaire pour interagir avec le WMS**
  - Automatique : option –a, effectuée lors soumission
  - Explicite : glite-wms-job-delegate-proxy + -d à la soumission
    - Plus performant si on doit soumettre un grand nombre de jobs

Suivi du Job et Résultats

- **Pendant le job, on peut suivre son exécution avec la commande glite-wms-job-status**
  - Utiliser '-v 1' sinon verbosité insuffisante
  - Utiliser l'option '–i jobids_file' si le jobid enregistré dans un fichier (-o lors de la soumission)
  - Utiliser 'watch –n seconds..' pour avoir un suivi "temps réel"
    - Attention à ne pas utiliser des intervalles trop courts (minimum 30s)
    - Ne pas utiliser pour des jobs longs ou pour un grand nombre de jobs
  - '--all' permet de voir le status de tous les jobs de l'utilisateur
- **Récupération du status détaillé :**
  - **`glite-wms-job-logging-info [-v 2]`**
- **Récupération des résultats (stdout/stderr, output sandbox) : glite-wms-job-output**
  - A l'initiative de l'utilisateur
  - **`Conservé environ 3 semaines sur le WMS`**

Exemple d'un fichier JDL

- **JDL : Job Description Language**
  - le programme et ses arguments
  - les fichiers d'entrés et de sorties
  - les « Requirements » et « Rank »
  - Utilise une syntaxe «Condor ClassAd »

attribut job

attribut données

attributs ressources

```
Executable = "gridTest";
StdError = "stderr.log";
StdOutput = "stdout.log";
InputSandbox = {"/home/joda/test/gridTest"};
OutputSandbox = {"stderr.log", "stdout.log"};
InputData = "lfn:testbed0-00019";
DataAccessProtocol = "gridftp";
Requirements = other.Architecture=="INTEL" &&
               other.OpSys=="LINUX" && other.FreeCpus
               >=4;
Rank = "other.GlueHostBenchmarkSF00";
```

Les Attributs JDL

- ***Attributs du job***
  - Défini le job lui-même
- ***Attributs sur les ressources***
  - pris en compte par le WMS et utilisé par l'algorithme de *matchmaking* (choix du site)
  - ressources de calcul (CE et clusters associés) principalement
- **Attributs sur les données**
  - Complémentaires des ressources
  - Prise en compte de la localisation des données pour le choix du site

Définition des Jobs

- Executable ***(obligatoire)***
  - le nom de la commande
- Arguments ***(optionnel)***
  - arguments de la ligne de commande du job
- StdInput, StdOutput, StdError ***(optionnel)***
  - standard input/output/error du job
- Environment ***(optionnel)***
  - liste de variables d'environnement
- InputSandbox ***(optionnel)***
  - liste de fichiers sur le disque local de l'UI nécessaire lors de l'exécution du job
    - Taille maximum définie par site : en général < 1MB
  - les fichiers listés sont envoyés depuis l'UI sur le CE
- OutputSandbox ***(optionnel)***
  - liste des fichiers, générés par le job, qui seront récupérés
  - Taille maximum généralement limitée à 10 MB (dépend du site)

Attributs sur les Ressources

- Requirements
  - besoin du job en ressource de calcul
  - spécifié en utilisant les attributs des ressources publiées dans le système d'information (BDII)
  - si non spécifié, la valeur par défaut définie dans le fichier de configuration de l'UI est utilisé
    - Site dependent
  - Les requirements ne sont pas passés au batch scheduler
    - Va devenir possible avec le CREAM CE
  - Possibilité d'expression complexe (syntaxe Condor ClassAds)
- Rank
  - exprime la préférence (comment classer les ressources qui ont rempli les conditions de l'attribut Requirements)
  - spécifié en utilisant les attributs des ressources publiées dans le système d'information
  - si non spécifié, la valeur par défaut définie dans le fichier de configuration de l'UI est considérée

Attributs pour les données

- InputData ***(optionnel)***
  - Données se trouvant sur la grille utilisées en entrée : elles sont généralement publiées dans le catalogue LFC de la VO
    - Généralement LFN mais éventuellement PFN possible
  - Site (CE) sélectionné doit avoir pour *close SE* un des SE détenant un des replica
  - A ne pas confondre avec l'input sandbox…
    - Fichiers (hors grille) qui doivent être envoyés avec le job

- DataAccessProtocol ***(obligatoire si InputData specifié)***
  - Le protocole ou la liste des protocoles avec lesquels l'application est susceptible d'accéder aux ***InputData*** sur un SE donné

Job Collections

- **But : faire une seule soumission pour plusieurs jobs**
  - Collection arbitraire : 1 JDL par job
    - Option '--collection directory'
  - DAG (Direct Acyclic Graph) : enchainement de jobs
    - Option '--dag directory'
  - Jobs paramétriques : 1 job avec plusieurs valeurs pour 1 paramètre
    - Obsolète : remplacé par les job collections
  - Pour DAG/collections, le répertoire contient le JDL des sous-jobs
    - Paramètre 'JDL file' ne doit pas être utilisé
- **Un seul job id pour l'ensemble des jobs permettant de vérifier le status, récupérer les outputs, arrêter…**
  - Output : 1 répertoire par sous-job + 1 fichier ids_nodes.map décrivant l'association entre les sous-jobs et les répertoires
    - Aussi 1 job id par sous-job
  - Status de la collection reflète celui de l'ensemble des jobs

Job Perusal

- **But : permettre d'examiner les fichiers produits par un job pendant son exécution**
  - Peut s'appliquer à tout fichier
  - Requiert 2 lignes supplémentaires dans le JDL :

```
PerusalFileEnable = true;
PerusalTimeInterval = 120;     # In seconds, not too low
```

- **Définition et récupération des fichiers à examiner : glite-wms-job-perusal [--set|--get|--unset] –f file jobid**
  - --set définit les fichiers à examiner
  - --get récupère la différence avec la version précédente
    - --all force la récupération de tous les fichiers
    - --nodisplay stocke le fichier plutot que de l'afficher
    - --unset : annule l'examen (la récupération périodique) du fichier
- **A utiliser avec modération : peut avoir un impact sur les performances du WMS**
  - Ne surtout pas utiliser en phase de production

VO Software Area

- **Chaque VO dispose d'un espace spécifique pour installer ses applications sur un CE**
  - Espace partagé par le CE et ses WNs
  - Référencé à travers une variable d'environnement :

    VO_VONAME_SW_DIR
  - VONAME est le nom de la VO avec les '.' remplacés par des '_'
- **Droit d'écriture restreint au seul VO Software Manager**
  - Accessible en lecture à tout le monde (toutes les VOs)
  - Software Manager définit avec un rôle VOMS (au choix de la VO)
- **Mise à jour de la SW area effectuée en soumettant des jobs avec le rôle Software Manager**
- **Contenu de la SW area peut être publié en définissant des tags depuis 1 UI ou 1 WN (job)**

  Lcg-ManageVOTag –host CE –vo voname …

Soumission Directe (CREAM CE)

- **CE : porte d'entrée sur le cluster d'un site**
  - Historiquement LCG CE basé sur Globus : utilisation direct rare
- **Nouvelle génération : CREAM CE**
  - En cours de déploiement : pas disponible sur tous les sites
  - Architecture très différente du LCG CE, plus performant
  - Même structure de commande que le WMS
    - Commandes : glite-ce-xxx
    - Pas (encore) de commande glite-ce-output : peut être complexe
- **Plusieurs mode de soumissions**
  - Via un WMS : pas de différence avec le LCG CE
  - Soumission directe : plus efficace mais pré-sélection du site
  - (aussi possibilité de soumission depuis Condor)
- **Job est décrit au moyen d'un fichier JDL**
  - Identique au WMS mais seulement section « job attributes »

lcg-infosites

- **Commande permettant de connaître la liste des ressources (CE, SE…) accessibles à une VO**
  - Utilise les informations dynamiques publiées par les sites (BDII)
  - Pas besoin d'être authentifié
- **Syntaxe très simple (*man lcg-infosites*)**

  lcg-infosites --vo biomed ce|se|lfc|wms
- **Exemple**

```
lx2/jouvin % lcg-infosites --vo biomed ce | grep lal.in2p3.fr:
1632    710     19          19      0    grid36.lal.in2p3.fr:8443/cream-pbs-biomed
1836    914     0           0       0    grid10.lal.in2p3.fr:2119/jobmanager-pbs-sdj
1632    710     19          19      0    grid10.lal.in2p3.fr:2119/jobmanager-pbs-biomed
```

- **Aussi possible de connaître l'état d'un site**
  - https://sam-fr-roc.cern.ch/myegee

Conclusions

- **Le grille fonctionne comme un grand système de batch**
  - Le composant principal est le WMS.
  - Son rôle est de trouver la meilleure ressource à partir des contraintes (Requirements et Rank) données par l'utilisateur
  - Il utilise le système d'information (BDII) pour sélectionner le site
- **Le service WMS simplifie la soumission de job:**
  - « Bulk » soumission
  - Jobs paramètriques, job collections, DAG jobs…
  - Swallow resubmission, Fuzzy Ranking…
  - VOMS proxy renewal (y compris les attributs VOMS)
  - Peut soumettre plus de 20 kjobs/jour/WMS
- **Pas la seule solution :**
  - Autres brokers comme GridWay, DIRAC, PanDA…
  - « Workflow managers » comme TAVERNA, MOTEUR, …
  - Soumission directe (CREAM CE) pour des utilisations avancées

Liens Utiles

- **Pour l'utilisation de la plateforme « desktop grid » EDGeS/DGHEP/XtremWeb**
  - http://dghep.lal.in2p3.fr/lal/doc/xwhep.html#SEC45
  - Très différent de l'utilisation de la grille basée sur gLite