

DIET Batch multi-jobs management system v.1.0

Daouda Traoré
daouda.traore@ens-lyon.fr

11 novembre 2010

Table des matières

| | | |
|----------|---|----------|
| 1 | Objectif | 2 |
| 2 | Présentation de l'ensemble logiciel "DIET batch management multi-jobs" | 2 |
| 2.1 | Implantation des différentes commandes | 3 |
| 2.2 | Pré-requis | 4 |
| 2.3 | Scénarios d'utilisation | 4 |
| 2.3.1 | Soumission d'un job ou d'un lot | 4 |
| 2.3.2 | Construction d'un lot à partir d'identifiants de tâches déjà soumises | 4 |
| 2.3.3 | Affichage de l'état des lots | 5 |
| 2.3.4 | Affichage du statut d'un lot | 5 |
| 2.3.5 | Suppression d'un lot | 6 |
| 2.3.6 | Récupération des résultats et des sorties d'erreurs | 6 |
| 3 | Logiciel et documents fournis | 7 |

Ce document détaille les objectifs et la mise en œuvre des différents logiciels composant l'ensemble logiciels *DIET batch management multi-jobs system*. La première section présente les objectifs tels qu'ils ont été définis avec le destinataire de cet ensemble logiciel. La seconde section présente la mise en œuvre effective des différents logiciels nécessaire à la bonne réalisation des objectifs définis dans la première section. La troisième et dernière section donne la liste des codes et documents fournis.

1 Objectif

Le service de gestion des batchs par l'intermédiaire de l'intergiciel DIET présenté dans ce document a pour objectifs :

- De permettre de gérer un très grand nombre de tâches élémentaires, de l'ordre de la dizaine de milliers jusqu'au million, soumises uniquement à partir de *clawi2q2*. Ces tâches peuvent être regroupées et soumises progressivement sur un ensemble de machines hétérogènes. Le module DIET développé doit contenir les fonctionnalités suivantes :
- Le lancement transparent, par l'utilisateur, d'applications paramétriques provoquant de grands nombres de tâches ;
- La soumission d'un lot (ensemble de tâches) par la liste des cartes (une carte représente ici la description d'une tâche) qui le constituent ;
- La construction d'un lot à partir d'identifiants de tâches déjà soumises aux ordonnanceurs (des ressources de calcul à haute performance) ;
- La fourniture, à l'utilisateur, d'un suivi personnalisé de ses tâches (nombre de tâches en cours, nombre de tâches terminées, pourcentage d'erreurs) ;
- La récupération des sorties de résultats et d'erreurs de ses applications ;
- La suppression d'un lot en supprimant l'ensemble des tâches qui le constituent.

2 Présentation de l'ensemble logiciel “DIET batch management multi-jobs”

L'intergiciel DIET développé par l'équipe GRAAL l'INRIA Rhône-Alpes a été utilisé pour construire le logiciel *batch management multi-jobs system* décrit dans ce document. Cet intergiciel est basé sur le paradigme client/serveur et d'appels de procédure distantes. Il donne la possibilité de définir des politiques d'ordonnancement avancées et permet d'utiliser un gestionnaire de données permettant la réplication et l'accès transparent aux données. Comme dans le document *batch management system*, le document présente aussi DIET par ces trois éléments principaux qui sont :

- Un client : c'est une application qui utilise DIET pour résoudre des problèmes. Plusieurs clients peuvent se connecter à DIET, ces clients peuvent être des applications web, des applications PSE (Problem Solving Environment) comme Matlab ou Scilab, ou d'un programme compilé.
- Un “Server Daemons” (SeD) : c'est un serveur qui permet de fournir les services permettant de résoudre les problèmes des clients. Un SeD encapsule un serveur de calcul. Il peut être par exemple placé sur le point d'entrée d'une machine parallèle (super-calculateur, cluster etc.) ou simplement sur une machine individuelle du réseau. Un SeD stocke la liste des services qu'il offre aux utilisateurs, mais aussi des informations sur sa charge et ses capacités matérielles (nombre de processeurs, charges des processeurs, quantité de mémoire installée etc.).
- Un agent : c'est un élément de DIET qui reçoit les requêtes des clients, et qui effectue la sélection du ou des serveurs utilisés pour l'exécution d'un service. les agents sont interconnectés suivant une topologie d'arbre dont le sommet est un

agent “maître” (Master Agent ou MA) qui sera seul contacté par les applications clientes à la recherche d’un service.

L’ensemble logiciel de *DIET Batch management multi-jobs* présenté dans ce document se décompose en différentes catégories :

- Les logiciels *serveur* qui permettent d’offrir ou d’exécuter les différents services de batch de la machine sur laquelle ils sont exécutés.
- Les logiciels *client* qui permettent d’appeler de manière transparente les différents services proposés par le *batch management multi-jobs system* en choisissant la machine concernée par l’opération à effectuer.
- Un ensemble d’utilsitaires permettant l’installation et la configuration des différents éléments de l’ensemble logiciel. Il s’agit des fichiers de configuration DIET et des fichiers permettant la compilation de l’ensemble logiciel (*les makefile*).

2.1 Implantation des différentes commandes

L’ensemble logiciel *DIET batch management multi-jobs system* permet la réalisation générique et distante des opérations batch courantes. Ces opérations sont présentées dans le tableau 1.

| Commande générique DIET | Rôle de la commande |
|----------------------------|--|
| diet-submit-lot | permet de construire un lot à partir d’une liste de cartes (jobs) ou à partir d’un exécutable générant des cartes et de soumettre le lot construit. |
| diet-build-lot | permet de construire un lot à partir d’identifiants de tâches déjà soumises (des ressources de calcul à haute performance). |
| diet-list-lot | permet d’afficher l’état (nombre de total de tâches d’un lot, nombre de tâches en attente, pourcentage d’erreurs, pourcentage de tâches terminée ...) des lots |
| diet-status-lot | permet de suivre l’état courant d’un lot, de lister les tâches qui le constituent ainsi que leur état. |
| diet-cancel-lot | permet de supprimer un lot en supprimant l’ensemble des tâches qui le constituent. |
| diet-GetOutput-lot | permet de récupérer les résultats et les sorties d’erreurs d’une tâche |
| diet-OnlineOutput-lot | permet de récupérer les résultats et les sorties d’erreurs de toutes les tâches d’une machine |

TABLE 1 – Commandes fournies par l’ensemble logiciel *DIET batch multi-jobs management*

2.2 Pré-requis

Comme dans le document *batch management system*, les pré-requis identifiés au fonctionnement de ces opérations sont les suivants :

- L'installation sur les machines "hôtes" d'un *server Daemon* DIET développé à cet usage. Il s'agit ici du logiciel serveur DIET *diet-bmmd batch management multi-jobs daemon*. L'utilisation de ce logiciel requiert :
 - La disponibilité d'une installation de DIET fonctionnelle sur la machine hôte. Ceci comprend la présence des bibliothèques nécessaires à sa bonne exécution et la présence sur le réseau accessible à la machine hôte d'un bus CORBA géré par l'ORB *omniORB* ainsi que le service de nommage CORBA *omniNames*.
 - Des paramètres et fichiers de configuration fixés à cet effet.
- La possibilité d'effectuer les appels aux logiciels *client* avec les paramètres adéquats dépendant du type d'opérations à effectuer. Ces appels pourront être effectués en ligne de commande *shell* traditionnelle ou par l'intermédiaire d'une interface utilisateur permettant leur exécution avec des paramètres requis.

2.3 Scénarios d'utilisation

Cette section présente différents scénarios d'utilisation des commandes de l'ensemble logiciels DIET *batch management multi-jobs system*.

2.3.1 Soumission d'un job ou d'un lot

Le client souhaite soumettre un job ou un lot (ensemble de jobs fourni dans un fichier) sur une machine distante. Pour cela il utilise la commande **diet-submit-lot** en lui passant en paramètre le script de soumission ou le fichier contenant un ensemble de script (lot).

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande de soumission en donnant le chemin menant au script ou en donnant le fichier contenant l'ensemble des scripts à soumettre (pour la soumission d'un lot).
\$ **diet-submit-lot** <path_to_script> pour la soumission d'un simple script.
\$ **diet-submit-lot** <path_to_lot> pour la soumission d'un lot.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet le script ou le lot à soumettre au serveur.
4. Le serveur soumet le script ou le lot localement et transmet l'identifiant du lot soumis ou d'éventuelles au client.

2.3.2 Construction d'un lot à partir d'identifiants de tâches déjà soumises

Le client souhaite construire un lot à partir d'identifiants de tâches déjà soumises aux ordonnanceurs. Pour cela il utilise la commande **diet-build-lot** en lui passant en

paramètre l'adresse de la machine distante et la liste des identifiants des tâches avec lesquelles il veut construire un lot.

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande de construction précisant la machine concernée et les identifiants des jobs. \$ **diet-build-lot** <hote> *ID1 ID2 ... IDi*.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet la demande au serveur.
4. Le serveur transmet l'identifiant du lot formé ou d'éventuelles erreurs au client.

2.3.3 Affichage de l'état des lots

Le client souhaite afficher l'état (nombre de tâches en attente, terminée et pourcentage d'erreurs ...) d'un lot en cours d'exécution sur une machine distante. Pour cela il utilise la commande **diet-list-lot** en lui passant en paramètre l'adresse de la machine distante.

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande d'affichage en précisant l'adresse de la machine concernée. \$ **diet-list-lot** <hote>.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet la demande d'affichage au serveur.
4. Le serveur exécute la commande d'affichage et transmet les résultats au client.

2.3.4 Affichage du statut d'un lot

Le client souhaite suivre l'état courant d'un lot, lister les tâches qui le constituent, ainsi que leur état. Pour cela il utilise la commande **diet-status-lot** en lui passant en paramètre l'identifiant du lot et l'adresse de la machine distante.

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande en précisant l'identification du job et l'adresse de la machine concernée. \$ **diet-status-lot** <LotID> <hote>.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet la demande avec l'identification du job au serveur.
4. Le serveur exécute la commande et transmet les résultats au client.

2.3.5 Suppression d'un lot

Le client souhaite supprimer d'un lot (la suppression d'un lot supprime aussi l'ensemble des tâches qui le constituent). Pour cela il utilise la commande **diet-cancel-lot** en lui passant en paramètre l'identifiant du lot et l'adresse de la machine distante.

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande en précisant l'identification du job et l'adresse de la machine concernée. \$ **diet-cancel-lot** <LotID> <hote>.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet la demande avec l'identification du job au serveur.
4. Le serveur exécute la commande et transmet les résultats au client.

2.3.6 Récupération des résultats et des sorties d'erreurs

Le client souhaite récupérer les résultats et les sorties d'erreurs de son lot. Pour cela il utilise la commande **diet-GetOutput-lot** en lui passant en paramètre l'identifiant du lot et l'adresse de la machine distante.

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande en précisant l'identification du job et l'adresse de la machine concernée. \$ **diet-GetOutput-lot** <LotID> <hote>.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet la demande avec l'identification du job au serveur.
4. Le serveur exécute la commande et transmet les résultats au client.

Pour récupérer en continue les résultats et les sorties d'erreurs des lots déjà terminés, le client doit lancer la commande **diet-OnlineOutput-lot** en tâche de fond avec l'adresse de la machine distante.

L'opération se déroule en plusieurs étapes transparentes pour l'utilisateur :

1. Le client entre la commande en précisant l'identification du job et l'adresse de la machine concernée. \$ **diet-OnlineOutput-lot** <hote> &.
2. L'application cliente contacte le master agent (MA) afin d'obtenir une référence au SeD de l'hôte concernée.
3. Le logiciel client transmet la demande avec l'identification du job au serveur.
4. Le serveur exécute la commande et transmet les résultats en continue au client.

3 Logiciel et documents fournis

Le logiciel *batch management multi-jobs system* décrit dans ce document comprend :

- Le code source de l'ensemble des logiciels fournis :
 - Le code source du serveur DIET *batch management multi-jobs system* permettant la soumission, la consultation et l'annulation des jobs groupés.
 - Le code source des clients DIET *batch management multi-jobs system* permettant les exécutions des différentes commandes.
 - Les fichiers *Makefile* permettant la compilation des exécutables *client* et *server*.
 - Le code source de la librairie utilitaire utilisé par le serveur et l'ensemble des clients.
- La documentation des logiciels fournis comprend :
 - Un guide de compilation/installation des logiciels.
 - Un guide de configuration.
 - Un guide d'utilisation.