وزارة التعليم العالي والبحث العلمي

BADJI MOKHTAR UNIVERSITY -ANNABA-
UNIVERSITE BADJI MOKHTAR -ANNABA-

جامعة باجي مختار
- عنابة -

Faculté : Sciences de l'Ingénieur -Année 2010 -

Département : Informatique

MEMOIRE

Présentation en vue de l'obtention du diplôme de magister

« Plateforme à base de grid pour la gestion des connaissances (application sur les systèmes CBIR)».

Option

Texte Parole et Image

Par

Ahmed Dib

DIRECTEUR DE MEMOIRE: S.Mokhtar Professeur Univ. Annaba

DEVANT LES JURY

PRESIDENT: KHADIR Tarek	**Maitre de conférences**	**Univ. Annaba**
EXAMINATEURS:		
Dr. SERIDI Hassina	**Maitre de conférences**	**Univ. Annaba**
Dr. BELLEILI Habiba	**Maitre de conférences**	**Univ. Annaba**
Dr. FARAH Nadir	**Maitre de conférences**	**Univ. Annaba**

Remerciement

En préambule à ce mémoire, je souhaitais adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu’à la réussite de cette formidable année universitaire.

Je tiens à remercier sincèrement Monsieur Sellami, qui, en tant que Directeur de mémoire, s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'il a bien voulu me consacrer et sans qui ce mémoire n'aurait jamais vu le jour.

J'exprime ma gratitude à mes parents et ma femme pour leur contribution, leur soutien et leur patience.

Enfin, j'adresse mes plus sincères remerciements à tous mes proches et amis, qui m'ont toujours soutenue et encouragée au cours de la réalisation de ce mémoire.

Merci à tous et à toutes.

ملخص
الصور تمثل جزءا كبيرا من جميع البيانات الرقمية على الويب. حاليا ، قد تحتوي البنوك المعلوماتية على المليارات من الصور ، وتشغل عدة تيرابايت من مساحة القرص.

فهرسة عدد كبير من الصور في قواعد البيانات وتوزيعها في مختلف بيئات العمل يتطلب موارد حسابية و موارد تخزين كبيرة. لتحقيق أقل قدر من الوقت الشامل لعلاج الصور في نظام CBIR (استرجاع الصور القائم على المحتوى) ،نلجأ الى استخدام تكنولوجيا الشبكة.

في هذا العمل ، ركزنا جهودنا على دراسة انترجيسيال الشبكة ، غلوبوس الإصدار 4 ، درسنا هندسته من أجل تطوير وتثبيت أحد التطبيقات التي تلبي احتياجاتنا. هذا التطبيق يعتبر المنصة التي يمكننا من خلاله تخصيص معالجة ، فهرسة واسترجاع الصورعلى أساس المحتوى.

في هذا العمل، قمنا باستغلال موارد الشبكة ، تم تنفيذ هذه العملية عن طريق الآليات المنصوص عليها من طرف غلوبوس لتغطية كافة الاحتياجات من حيث الموارد الحاسوبية ، أو التخزين ، وبالتالي الإجابة على أسئلة عديدة من محركات البحث والصور ونظم CBIR بصفة عامة.

المصطلحات
الحوسبة الشبكية ، انترجيسيال ، غلوبوس ، خدمات الويب ، خدمات الشبكة ، WSRF، CBIR ، فهرسة ، تجزئة.

Résumé

Les Images représentent une grande partie de l'ensemble des données numériques sur le Web. Actuellement, les bases des images peuvent contenir une vingtaine de milliards d'images, et occupent plusieurs tera-octets d'espace disque.

L'indexation d'un nombre important des images dans des Bases de données réparties et dans des environnements hétérogènes, nécessite des ressources de calcul et de stockage importantes. Afin d'optimiser le temps de reponse des traitements de masse sur les images d'une plateforme CBIR (Content-Based Image Retrieval), on a fait recours à la technologie des grilles.

Dans ce travail, on a focalisé notre étude sur l'Intergitiel de grille, Globus Toolkit dans sa version 4, on a étudié son architecture et ses modules afin de pouvoir l'installer et développer une application qui répond à nos besoins. Cette application est considérée comme une plateforme sur laquelle on peut personnaliser les traitements des images, et bien entendu l'indexation et la recherche à base de contenu.

Avec ce travail, on a exploité les ressources de la grille de test installée, cette exploitation est réalisée via des mécanismes offerts par Globus afin de couvrir tous les besoins en terme de ressource de calcul, ou de stockage, et faire une indexation distribuée d'un nombre exponentiellement croissant des images avec des performances élevées, et donc la réponse à plusieurs questions des moteurs de recherches des images et des systèmes CBIR en général.

Mots clés

Grille de calcul, Intergitiel, Globus, Web service, service de grille, WSRF, CBIR, Indexation, Segmentation.

Abstract:

The images represent a large part of all digital data on the Web. Currently, the basis images may contain billions of images hold several terabytes of disk space.

The indexing of a large number of images in databases in distributed and heterogeneous environments requires computational resources and large storage. To improve the processing time of big data bases of images for a platform CBIR (Content-Based Image Retrieval), we made use of grid technology.

In this work, we focused our study on Intergitiel grid, Globus Toolkit version 4, we studied its architecture and its modules in order to develop and install an application that meets our needs. This application is considered as a platform on which we can customize the image processing, and of course the indexing and retrieval based on content.

With this work, we exploited the resources of the grid installed, this operation is performed via the mechanisms provided by Globus to cover all needs in terms of computing resource, or storage, and make a distributed indexing of an exponentially growing number of images with high performance, and therefore the answer to several questions of the search engines images and CBIR systems in general.

Keywords

Grid computing, Intergitiel, Globus, Web services, grid services, WSRF, CBIR, indexing, segmentation.

Table de matières

Introduction générale 2

1. La technologie des grilles 6

1.1 Introduction 6

1.2 Définition d'une grille 7

1.3 Différents types de grilles informatiques 8

1.3.1 Grilles d'information 8

1.3.2 Grilles de données 8

1.3.3 Grilles de calcul 8

1.4 Application des grilles de calcul 10

1.4.1 Supercalculateur distribué (Distributed Supercomputing) 10

1.4.2 Calcul haut débit (High-Throughput Computing) 10

1.4.3 Calcul à la demande (On-Demand Computing) 11

1.4.4 Calcul Collaboratif (Collaborative Computing) 11

1.4.5 Traitement massif de données (Data intensive Computing) 12

1.5 Évolution des technologies de la grille 13

1.5.1 Première génération 13

1.5.2 Seconde génération 13

1.5.3 Troisième génération 14

1.6 Notion d'organisation virtuelle (VO) 16

1.7 Architecture d'une grille 17

1.7.1 Couche fabrique 18

1.7.2 Couche connectivité 19

1.7.3 Couche ressources 20

1.7.4 Couche collectif 21

1.7.5 Couche application 22

1.8 Les Intergitiels 23

1.8.1 Globus 23

1.8.2 Unicore (UNiform Interface to Computer Resources): 24

1.8.3 Legion : 24

1.9 Open Grid Services Architecture (OGSA), Open Grid Services Infrastructure (OGSI), Web Services (WS) 25

1.9.1 Présentation 25

1.9.2 Services Web 26

A. Définitions 26

B. Architecture des Services Web : 27

C. Invocation des Services Web : 28

1.9.3 Architecture Orientée Service (SOA) 29

1.9.4 Web Services Resources Framework (WSRF) 30

1.9.5 Conclusion 33

2. Globus Toolkit 4 35

2. 1 Introduction 35

2. 2 Architecture de Globus 38

2. 3 Modules de Globus 39

2.3.1 Module de gestion de données 40

A. GridFTP 40

B. Reliable File Transfer (RFT) 41

C. Replica Location Service (RLS) 42

D. Data Replication Service (DRS) 43

2.3.2 Le module de sécurité (Globus security Infrastructure GSI) 44

2.3.3 Module de gestion d'exécution 46

2.3.4 Le module d'information (Monitoring and Discovery System MDS4 ou WS MDS) 48

A. Services de haut niveau (higher-level services) 49

B. Fournisseurs d'informations 49

C. Client WebMDS 50

2. 4 Conclusion 50

3. Introduction en traitement d'images 52

3.1 La matrice de Cooccurrence 52

3.2 Filtrage 53

3.2.1 Filtrage dans le domaine spatial 53

3.2.2 Filtrage dans le domaine fréquentiel 54

3.3 Segmentation 54

3.3.1 Méthodes statistiques 55

A. Histogrammes et Segmentation 55

B. Segmentation par sélection récursive sur histogrammes 55

3.3.2 Méthodes géométriques 55

A. Croissance de régions (Region growing) 55

B. Décomposition/fusion (Split & Merge) 55

3.3.3 Méthodes par optimisation 56
3.4 Classification 56
A. Méthodes de classification automatique 56
B. Méthodes de classification supervisée 56
C. Classification hybride 56
3.5 Conclusion 57
4. Systèmes CBIR à base de grilles 59
4.1 Introduction 59
4.2 Etat de l'art 59
4.3 Conception de plateforme CBIR basée sur Globus 76
4.3.1 Introduction 76
Interface de passage de message (MPI) 76
Architecture parallèle 77
Types de tâches dans une application parallèle 77
Décomposition de problème 77
4.3.2 Fonctionnalités de la plateforme 78
A. Traitement de masse 78
B. Reconnaissance, classification et indexation 79
C. Recherche 79
4.3.3 Structure de la plateforme 80
4.3.4 Architecture de la plateforme : 80
A. Ressources matériels et logiciels (La grille) 81
B. L'Intergitiel Globus 81

C. La couche traitement (Processing) 81

D. La couche contrôle 87

E. Application 92

4.4 Résultats 93

4.5 Conclusion 100

5. ANNEXE A 102

5.1 Manuel d'installation de l'Intergitiel Globus (GT 4.0.6) sous Linux Fedora core4 102

5.1.1 Quelques commandes Linux à connaître (Pré-requis): 102

5.1.2 Mise en place du réseau : 103

A. Installation du système Linux 103

B. Configuration du réseau 104

C. Outils nécessaires 104

5.1.3 Préparation de l'installation de globus toolkit 4.0.6 104

A. Création des comptes utilisateurs 105

B. Création des répertoires d'installation 105

C. Installation des outils apache Java, Apache-ant et Postgresql 105

5.1.4 L'installation de Globus Toolkit 107

A. Lancement du script d'installation 107

B. Installation de l'Autorité de Certification (CA): 109

C. Génération du certificat pour le hôte: 111

D. Génération du certificat pour l'utilisateur globus: 112

E. Génération du certificat pour l'utilisateur user: 113

F. Création du certificat du 'container': 115
G. Ajout des autorisations: 115
H. Vérification des certificats des utilisateurs 116
5.1.5 Installation de certificat pour plusieurs machines 117
5.1.6 Service 'gridFTP': 123
A. Configuration du service 'gridFTP': 123
B. Lancement du service 'gridFTP' 126
5.1.7 Lancement du container des services web: 126
5.1.8 Configuration du RFT 129
A. Création du fichier pg_hba.conf 129
B. Création d'un utilisateur globus sous postgres 129
C. Création de la base de données rftDatabase 130
D. Test de fonctionnement du RFT 131
5.1.9 Configuration du service GRAM 132
5.1.10 Configuration de gridFTP, RFT et GRAM sous les autres machines : 132
5.2 Interface Graphique de COG Kit 134
5.3 Conclusion 134
6. ANNEXE B 136
6.1 Traitement des images 136
6.2.1 Structure d'une région 136
6.2.2 Structure de fichier .seg 137
6.2.3 Calcul des caractéristiques visuelles d'une image 137
6.2 Application 140

6.2.1 Structure de la table des images indexées 140

6.2.2 Environnement de développement 140

6.2.3 Outil d'aide au développement 141

6.2.4 Etapes de développement de service de segmentation 142

A. Définition de l'interface du service 142

B. Implémentation de l'interface 146

C. Développement Côté client 150

D. Configuration de déploiement du service 152

E. Génération du fichier GAR 152

6.2.5 Conclusion 153

Liste de figures

Figure 1-1: Chronologie de l'évolution technologique 15
Figure 1-2: Architecture en couche des grilles 17
Figure 1-3: Eléments de base de l'architecture OGSA 25
Figure 1-4: Architecture des Services Web 28
Figure 1-5: Service-oriented architecture (SOA). 29
Figure 1-6: L'architecture OGSA implémente WSRF 32
Figure 1-7: Relation entre WSRF, OGSA et les Services Web 32
Figure 2-1: Relation entre GT4, WSRF, OGSA et les Services Web 36
Figure 2-2: Modules de Globus Toolkit 4 37
Figure 2-3: Vue Client/Serveur de l'architecture de Globus 4..... 38
Figure 2-4: Les quatre modules de Globus. 39
Figure 2-5: Architecture du service RFT 42
Figure 2-6: exemple d'un déploiement de RLS. 43
Figure 2-7: basique opération de GSI 45
Figure 2-8: Vue en couche de l'infrastructure GSI. 46
Figure 2-9: Architecture en couche de MPICH 48
Figure 2-10: framework d'agrégation 50
Figure 4-1: Composants utilisés avec le système Ganga pour la définition, la soumission et le contrôle de Jobs et le contrôle de *particle physics Grid* 60
Figure 4-2: domaines implémentés par Imense..... 61
Figure 4-3: Vue en couche de l'analyse des images et le processus de reconnaissance par Imense Ltd..... 62
Figure 4-4: analyse du visage humain. 62
Figure 4-5: interaction entre le T2 et EGEE via le plugin T2 gLite durant une soumission de Job 65
Figure 4-6: l'architecture de base de l'application gridifiée. 67
Figure 4-7: l'infrastructure utilisée pour le déploiement de l'application..... 67
Figure 4-8: DM2 interface entre les données médicales et la grille. 68
Figure 4-9: Interface Web de système MedGIFT. 70
Figure 4-10: Une description XRSL de Job pour l'exécution distribuée avec l'utilisation du middleware ARC. 71
Figure 4-11: structure en couche de AGIR. 73
Figure 4-12 : (a) gestion des informations et des images à travers les hôpitaux; (b) partage des informations et des images entre les hôpitaux et les différentes stations..... 74
Figure 4-13: plateforme et relation avec les autre technologies de grille. 80
Figure 4-14: Architecture en couche de la plateforme. 81
Figure 4-15 Ajout d'une nouvelle fonctionnalité au module IAM 82
Figure 4-16: interaction entre le module IAM et la couche supérieure..... 84
Figure 4-17 Informations sauvegardées dans la base de cas 85
Figure 4-18 : Interaction du module contrôle avec les autres modules de la plateforme 87
Figure 4-19: gestion des classes d'images 88
Figure 4-20 :L'opération de recherche. 89
Figure 4-21 : introduction du CBR dans la recherche. 90
Figure 4-22: collection d'information auprès de service MDS, vérification de l'existence de données et lancement de tâches sur les hôtes moins chargés. 92
Figure 4-23: Diagramme de comparaison. 94
Figure 4-24: Ordre chronologique des étapes de segmentation de 1000 images. 95
Figure 4-25: stabilisation de temps de segmentation après la troisième répitition. 96

Figure 4-26: les dix classe de Wang (Deselaers, 2003). 97
Figure 5-1: Une vue du fichier /etc/profile.d/jdk.sh 105
Figure 5-2: Une vue du fichier /etc/profile.d/ant.sh 106
Figure 5-3: Interface graphique COG. 134
Figure 6-1: Structure de la table Index des images. 140
Figure 6-2: environnement utilisé pour le développement. 140
Figure 6-3: structure générale de l'application. 141
Figure 6-4: structure de service de segmentation. 141
Figure 6-5: classe de service de segmentation. 147
Figure 6-6: classe SegmentationQName. 147
Figure 6-7: classe SegmentationRessource. 148
Figure 6-8: classe segmentationProcess. 148
Figure 6-9: classe SegmentationFactoryService.java 148
Figure 6-10: classe SegmentationRessourceHome 149
Figure 6-11: interaction entre classes de segmentation. 150
Figure 6-12: interface client pour effectuer une indexation d'une source de données. 151
Figure 6-13: interface graphique pour la recherche. 152

Liste de Tables

Tableau 1: machine composantes de la grille de test. 93
Tableau 2: comparaison des temps CPU de segmentation d’images sur un nombre variant de machine. 94
Tableau 3: résultats détaillés des principales opérations de segmentation par machine. 95
Tableau 4: résultats de l'indexation séquentielle sur 50 images. 97
Tableau 5: résultats d'indexation de 50 images par les services de la grille. 98

Introduction générale.

Introduction générale

Le traitement des images est un processus très utilisé en nombreux domaines d'application. Ainsi, on peut le rencontrer dans le domaine médical, les moteurs de recherche des images à base de contenu, les systèmes géographiques… Généralement, on appelle les systèmes qui traitent les images à base de leurs contenus, des systèmes CBIR (Content-Based Image Retrieval). On donne l'exemple où le traitement des images avec une grande capacité de calcul est indispensable, les services de radiologie modernes, qui sont de plus en plus numériques, et en même temps la quantité de données produite augmente [60]. Comme les images sont une partie importante du processus de diagnostic de maladies, de nombreuses applications d'imagerie médicale, ont été développées au cours des 20 dernières années. La plupart des applications ont porté sur un genre très spécifique d'images, d'une région anatomique, et souvent une maladie. Les Systèmes CBIR, ont pour but de permettre la récupération des images ou des cas similaires sur des collections d'images très hétérogènes [61, 62, 63] pour aider le processus de diagnostic. Avec les services de radiologie moderne produisant régulièrement des dizaines de milliers d'images par jour [64], il est apparu que les infrastructures basées sur la technologie des grilles de calcul sont nécessaires pour traiter cette masse de données.

L'objectif de la grille est en général d'avoir un très grand nombre de machines géographiquement distribuées qui peuvent être partagées pour réaliser des tâches de calcul intensif. Les systèmes ont besoin de gérer plusieurs ressources de nature hétérogènes, pour cela, la technologie des grilles est utilisée pour partager, synchroniser et exécuter les tâches requises. De nombreuses solutions ont proposé des approches techniques pour les grilles de calcul. L'origine des grilles remonte à la fin des années 1980 [65]. Grands et complexes FrameWorks ont apparu, tels que Globus [66] dans la fin des années 1990 qui a créé une base de développement middleware supplémentaire.

Actuellement, il existe une multitude d'Intergitiels de grille, par exemple gLite, UNICORE [67] et l'ARC (Advance Resource Connector) [68]. L'utilisation des grilles de calcul a été favorisée dans le domaine de la santé par l'initiative healthGrid1 en 2002. En général, la plupart des applications médicales basées sur les grilles se concentrent sur les problèmes de calcul intensif [69, 70]. La plupart de ces applications se concentrent également sur l'utilisation de grandes grappes.

En 2002, aux hôpitaux universitaires de Genève, un projet de grille a été lancé pour identifier les défis de cette technologie dans le domaine médical [71]. Le but était d'utiliser la technologie de grille avec un nombre important d'ordinateurs de bureau (6.000 machines) comme une ressource pour les projets de recherche. La plupart des hôpitaux ne disposent pas d'infrastructure de recherche en informatique, et pas de personnel pour maintenir une telle infrastructure. D'un autre côté, un problème de sécurité des données médicales a été posé. Des premières mesures concrètes pour une telle infrastructure ont été présentées dans [72]. Plusieurs autres auteurs favorisent l'utilisation des infrastructures basées sur la technologie des grilles [61, 73].

Le mémoire est organisé comme suit :

Le premier chapitre présente la technologie des grilles, son origine et l'évolution avec laquelle les technologies informatiques ont passé pour en arriver à la grille. De même, on présente l'architecture des grilles, les différentes couches qui les composent, les applications liées à la grille, quelques architectures qui ont fait la base des systèmes distribués connus jusqu'à maintenant. Cette partie fait aussi l'objet de présentation de quelques Intergitiels répendus, et la présentation de technologie des services Web. On termine le premier chapitre par la présentation du service à état et la notion des spécifications WSRF.

Le deuxième chapitre présente en détail l'Intergitiel Globus sur lequel est fondée cette étude. Dans un premier temps, on présente son architecture, et on se focalise sur les modules de cet Intergitiel.

Le troisième chapitre est une introduction en traitement d'image. Il présente les opérations de bases en imagerie.

Dans le quatrième chapitre, on commence par la présentation des systèmes CBIR, des notions dans le domaine des systèmes répartis, un état de l'art des systèmes CBIR basés sur les grilles. Ensuite, on va présenter l'architecture de la plateforme développée, ses services de grille, ainsi que sa gestion. A la fin, on donne les résultats obtenus de quatre scénarios qui portent sur l'exploitation des ressources de la grille.

Et pour finir, l'Annexe A porte sur l'installation de Globus Toolkit et la construction de la grille de test, les outils nécessaires, et un ensemble de tests pour vérifier le bon fonctionnement des modules installés. Dans l'Annexe B, on donne quelques bouts de codes Java et XML, utilisés dans un premier temps pour l'implémentation des algorithmes en imagerie (calcul de caractéristiques visuelles d'une image, la structure des régions…), et dans

un deuxième temps pour le développement et la configuration des services de grille développés. On donne aussi un aperçu de l'interface de l'application (indexation et recherche).

La technologie des grilles

1. La technologie des grilles

- **Introduction**

Les réseaux sont un assemblage d'équipements informatiques qui a pour but de supporter la communication entre différentes machines inter-connectées entre elles. En fait, le plus grand réseau connu jusqu'à maintenant est l'Internet, elle offre le moyen de mettre en œuvre un ensemble d'applications Web dans différents domaines notamment e-Business, e-Learning, online conference…

De nos jours, vu l'utilité de partage de ressources hétérogènes, une évolution des technologies Web est devenue de plus en plus demandée pour la réalisation de grands projets utilisant des ressources importantes et géographiquement distribuées, délocalisées et autonomes, ce qui implique un décroît des performances généralement coûteux, d'où l'apparition de la technologie des grilles informatiques qui répond aux besoins cités précédemment, elle fournit une puissance et une fiabilité qui permettent d'accroître les performances globales des environnements distribués. Les grilles informatiques ont prouvé elles mêmes leurs puissances, ils peuvent être à la base de développement des environnements qui demandent des performances élevées, tout en fournissant des outils pour gérer une multitude de ressources, et peuvent donner plus de confiance aux utilisateurs afin de manipuler et partager leurs ressources en toute sécurité.

- **Définition d'une grille**

Le terme de la grille est apparu en informatique dans les années 1990 [2], il s'inspire de la grille d'électricité [4].

Une grille peut être vue comme un assemblage de ressources logicielles et matérielles distribuées, hétérogènes et partageables telles que les unités de stockages, les unités de calcules, les ressources réseaux et les entités logiques comme les fichiers système distribués et les clusters.

Ainsi, ils existent plusieurs définitions de la grille ;

– « Une grille informatique est une infrastructure virtuelle, garantissant des qualités de services non triviales, et constituée d'un ensemble de ressources informatiques partagées, hétérogènes, coordonnées et non contrôlées. » [1]

– « il nous permet d'accéder à des ressources hétérogènes à partir de différentes organisations en fournissant un ensemble de protocoles, technologies, et méthodologies… » [3]

Une grille informatique est une infrastructure virtuelle constituée d'un grand nombre de ressources (cycles CPU, données, espace mémoire de stockage) qui ne possèdent pas d'une relation physique, mais juste de point de vue logique; et pour ce faire il est nécessaires de disposer d'une architecture matérielle permettant l'interconnexion entre ces ressources, et d'une architecture logicielle afin de gérer et contrôler cet ensemble.

- **Différents types de grilles informatiques**

On distingue trois types de grille :

1.3.1 Grilles d'information

La notion de grille d'information consiste en le partage de l'information à travers un réseau. Le net est basé sur ce principe de partage de connaissances, toute organisation peut publier les connaissances qu'elle possède afin de les rendre disponibles pour d'autres unités, en y fournissant un ensemble d'outils pour manipuler (rechercher, interroger, ..) ces connaissances. Ce genre de plateforme (grille) doit être doté de mécanismes pour gérer de grandes masses d'informations, les indexer, les classer et les trouver efficacement en cas de besoin.

1.3.2 Grilles de données

Ce type de grille (appelé caches distribués) permet le partage de données entre plusieurs noeuds, plus une donnée est demandée, plus elle sera répliquée pour la rendre plus disponible et facilement accessible, aucun calcul n'est effectué. Les grilles de données ont pour objectif de maximiser la récupération d'objets tout en assurant une cohérence des données dans un contexte multi-machines. Les réseaux d'échange Peer-To-Peer ("P2P") sont le cas le plus représentatif de ce concept, ils permettent l'accès aux données via un réseau de serveurs partageant un indexe, ce dernier est utilisé pour référencier les données est pour effectuer la recherche à travers des moteurs très performants.

Les données peuvent être divisées selon leurs volumes, identifiées avec une fonction de hachage, répliquées sur un nombre de noeuds variant selon leurs demandes.

1.3.3 Grilles de calcul

Les idées de partage de ressources de calcul et de stockage ont commencé dans les années 80 avec de grands projets tel que Condor [6], la demande de ce partage a eu lieu dans le domaine de la physique des particules, où de grandes masses de données sont produites.

Appelé également calcul distribué ou calcul partagé, ce type de grille permet de réaliser des calculs intensifs demandant un nombre important de ressources de calcul (processeurs, mémoires, ressources réseau, ..).

Une grille de calcul est constituée de ressources informatiques hétérogènes liées par un réseau, c'est une forme d'informatique distribuée, basée sur le partage dynamique des ressources afin d'augmenter les performances des environnements, et pour l'accomplissement d'une tâche gourmande en terme calcul.

Pour pouvoir coopérer un ensemble de ressources hétérogènes, il est nécessaire de se posséder d'une couche logicielle, généralement nommée Intergitiel, ce dernier masque toute sorte d'hétérogénéité au niveau matériel qu'au niveau logiciel, cette couche permet d'exploiter efficacement et en toute sécurité les différentes ressources disponibles sur la grille.

- **Application des grilles de calcul**

Vu la puissance offerte par les grilles de calcul, la disponibilité des ressources qu'elles peuvent garantir, ainsi la sécurité des transactions, elles peuvent êtres impliquées dans plusieurs domaines d'application qu'on peut les classer en cinq catégories principales:

1.4.1 Supercalculateur distribué (Distributed Supercomputing)

La grille de calcul peut contenir un nombre important de calculateurs, qui travaillent en synchronisation pour fournir la puissance de calcul demandée afin de réaliser une tâche gourmande. Les ressources agrégées peuvent être hétérogènes. Différentes technologies peuvent être incluses. Parmi les ressources qu'on peut trouver, on cite les stations de travail, les grappes et les clusters.

Dans le domaine de supercalculateur distribué, l'infrastructure doit être dotée de mécanismes et d'algorithmes capables de gérer l'ordonnancement à grande échelle, tout en garantissant des performances globales élevées qualifiantes l'utilisation des grilles de calcul dans une telle infrastructure hétérogène.

1.4.2 Calcul haut débit (High-Throughput Computing)

C'est un terme publié par le projet Condor (Miron Livny, University of Wisconsin). Utilise de puissants calculateurs distribués (potentially grid computing) pour affecter aux processus les ressources demandées afin de terminer le plus vite possible le traitement des lots de tâches. Les ressources de calcul qui peuvent êtres utilisées dans ce type d'application sont des serveurs centralisés comme les clusters et les Grappes, ou des PCs distribués. Les applications qu'on peut réaliser avec le calcul haut débit sont les applications de cryptographie, la simulation des molécules et l'analyse des ADN.

Plusieurs projets ont été élaborés dans ce contexte, on cite :

– **World Community Grid (WCG) [91]:** un projet lancé en Novembre 2004 par IBM, dans le but de créer la plus grande grille de calcul pour des projets de recherche. En utilisant des cycles de processeurs non utilisés des ordinateurs connectés à Internet dans le monde, le projet WCG de recherche a réussi d'analyser les aspects du génome humain, le VIH, la dystrophie musculaire, et le cancer.

– **Large Hadron Collider (LHC projet) [92] :** c'est le plus grand accélérateur de particules mis en opération fin 2008 par l'Organisation européenne pour la recherche nucléaire

(CERN). Destiné à s'opposer à des collisions de faisceaux de protons ou des ions de plomb. La grille du HLC est un assemblage de plus de 140 centres de calcul dans 33 pays, le principal objectif est de construire une infrastructure d'analyse et de stockage d'une dizaine de million de GO de données produites par le LHC.

1.4.3 Calcul à la demande (On-Demand Computing)

Dans ce type d'application, les utilisateurs de la grille veulent exploiter les ressources afin de se bénéficier à court terme d'une fonctionnalité qui n'est pas offerte en local, ou minimise la rentabilité du système, comme l'utilisation d'un capteur spécialisé, des cycles processeur, medias de stockages temporaires ou même des applications.

Les grilles de calcul à la demande ont été conçues pour surmonter le défi commun des entreprises d'être en mesure de répondre efficacement les fluctuations de la demande. Computer Associates, HP, IBM, Microsoft et Sun sont parmi les plus éminents dans la vente d'utilisation des ressources à la demande, ces sociétés se réfèrent à la demande de leurs produits et services par une variété d'entreprises utilisateurs.

Des notions telles que (Grid Computing, utility computing, autonomic computing, et adaptive management) sont très similaires à la notion de calcul à la demande qui englobe tous les autres termes cités précédemment, chacun d'eux signifie quelque chose de légèrement différent. Utility computing, par exemple, est une approche à la demande qui allie l'externalisation des ressources informatiques et la gestion des infrastructures à un usage à base de structure de paiement, (cette approche est aussi connue sous le nom de services de compteurs).

1.4.4 Calcul Collaboratif (Collaborative Computing)

Cette classe inclut des applications interactives dans le but de favoriser les échanges, la collaboration entre humains dans des environnements de simulation en temps réel et dans des espaces virtuels et partagés. On prend par exemple :

– **Le système d'information médicale (SIM)** qui est un exemple typique d'une application de l'informatique collaboratif dans lequel des médecins, des infirmières, des professeurs, des chercheurs, des personnels de l'assurance médicale, etc. partagent l'information des patients y compris les textes, les images et les données multimédia.

1.4.5 Traitement massif de données (Data intensive Computing)

Ce sont des applications qui collectent de grandes quantités de données auprès des bases de distribuées. Ce type d'application est très gourmand en terme de demande de ressources de calcul qu'aux ressources réseau et de stockages. Pour cette classe d'application, on peut citer :

– **Meteo-GRID [93]:** c'est l'une des trois grandes applications de l'EUROGRID, utilisée pour la prévision météorologique, elle baillis 10 km quatre fois par jour pour fournir des prévision pour le grand public. L'exécution d'un tel modèle de prévision demande beaucoup de ressources de calcul, elle nécessite jusqu'à 60 x 1012 opérations en virgule flottante (flop), et crée environ 20 Go de données de prévision à l'heure de sortie. En outre, les prévisions météorologiques à court terme sont une tâche critique qui doit être achevée en moins de deux heures. Ainsi, seuls les centres des ordinateurs de haute performance (HPC) sont en mesure d'exécuter de telles prévisions.

- **Évolution des technologies de la grille**

L'évolution des grilles a passé par trois étapes, et elle est toujours en cours d'avancement et d'implémentation de nouvelles technologies. La première génération inclut des systèmes qui ont tenté de lier des supercalculateurs pour bénéficier de leurs puissances réunisses. Dans la seconde génération, l'accent a été mis sur les middlewares qui supportent le calcul et la mise à jours des données à grande échelle. La troisième génération essai de réaliser des systèmes favorisants une collaboration mondiale et distribuée.

1.5.1 Première génération

Les années 1990 sont le début de l'apparition des environnements des grilles, l'objectif des systèmes de cette génération été juste de fournir les ressources de calcul nécessaires pour répondre aux besoins des applications à haute performance. Les deux projets marquants cette étape sont Fafner [7] et l'I-WAY [8]. Les problèmes rencontrés dans cette étape sont ceux liés à la communication, la gestion de ressources et le traitement distant de données. Les deux Fafner et de l'I-WAY sont très innovants et fructueux, chaque projet a contribué à ouvrir la voie à un grand nombre d'autres projets qui ont connus le succès dans le domaine des grilles, Fafner a été le précurseur de SETI @ home [9] et Distributed.Net [10]. I-WAY précède Globus [11] et Légion [12].

1.5.2 Seconde génération

Dans la seconde génération, plusieurs applications basées sur la grille qu'on cannait aujourd'hui ont été développées, vu l'évolution des technologies des réseaux, et l'adoption des standards connus à l'échelle mondiale, la grille de deuxième génération ait vu comme étant une infrastructure distribuée qui supporte des applications demandant des ressources distribuées à grande échelle.

Trois problématiques ont distingué cette étape [13]:

– **Hétérogénéité :** L'infrastructure de la grille implique l'introduction de multiples ressources de nature hétérogène. L'Intergitiel est la couche logicielle intermédiaire entre le système d'exploitation et les applications. Il permet de cacher l'hétérogénéité, et donne une vue d'homogénéité aux applications des utilisateurs en les fournissant une variété de services et d'interfaces standards.

– **Passage à l'échelle :** La grille peut utiliser des milliers voir million de ressources, ce qui provoque une dégradation des performances globales des plateformes. Les applications demandant des ressources géographiquement distribuées, doivent être tolérantes, et exploitent au maximum les ressources les plus optimales à acquérir en sur-mentant les problèmes d'hétérogénéités, les limites des politiques de sécurité et d'augmenter le passage à l'échelle. Ces applications peuvent être composées, ce qui croît la complexité des systèmes basés sur la grille.

– **Adaptabilité :** Afin de bénéficier le maximum des performances des services et ressources disponibles, les applications doivent adapter leurs comportements dynamiquement selon les ressources demandées.

Dans la deuxième génération, plusieurs technologies ont apparus et d'autres ont été évoluées. On peut citer (1) les Intergitiels tels que 'Globus' et 'Legion', ils ont constitué le corps des technologies de la seconde génération. (2) Courtiers des ressources de grille (3) Les systèmes paire à paire, (4) Les systèmes d'objets distribués et on a plusieurs exemples comme : (a) CORBA (Common Object Request Broker Architecture) [94] qui a été normalisé par l'OMG (Object Management Group), c'est une infrastructure qui fait partis du projet ODP-RM (Open Distributed Processing Référence Model) vise à mettre en place une norme d'architectures distribuées ouvertes dans le but de faire communiquer des applications dans un environnement hétérogène. (b) Jini [95] qui utilise le mécanisme RMI pour réaliser la communication entre applications. Il fournit toute une infrastructure logicielle destinée aux environnements de calcul distribué.

1.5.3 Troisième génération

C'est la génération la plus innovante, sa période a commencé en 2001 accompagnée de l'apparition de l'architecture OGSA (Open Grid Services Architecture), cette étape est marquée par la réalisation de la boite à outils Globus 3.0. Elle fait l'objet de l'intégration des technologies des grilles précédentes avec celles des Web services et la virtualisation des ressources [14]. Comme la technologie des Web services utilise des approches plus standards, les grilles peuvent les utiliser pour standardiser la façon d'accès aux différentes applications sur le réseau, et donc résoudre des problèmes persistants de la génération précédente. L'architecture OGSA détaillé par la suite, a fournit un framework avec lequel on peut construire des services de grille portables et inter-opérables. OGSA a permis aux grilles d'utiliser des protocoles normalisés, ce qui a permis d'augmenter la virtualisation des

ressources, multiplier les formes et méthodes de partage, et d'améliorer la qualité des services fournis avec un enrichissement de leurs fonctionnalités et l'amélioration des performances globales des plateformes basées sur les grilles. L'étape représentée par la période qui suit l'an 2003, est marquée par des évolutions très coûteuses en introduisant les réseaux sans fil et l'utilisation des capteurs pour une tentative de commercialisation des technologies des grilles.

La figure 1-1 montre les étapes d'évolution de la technologie de la grille.

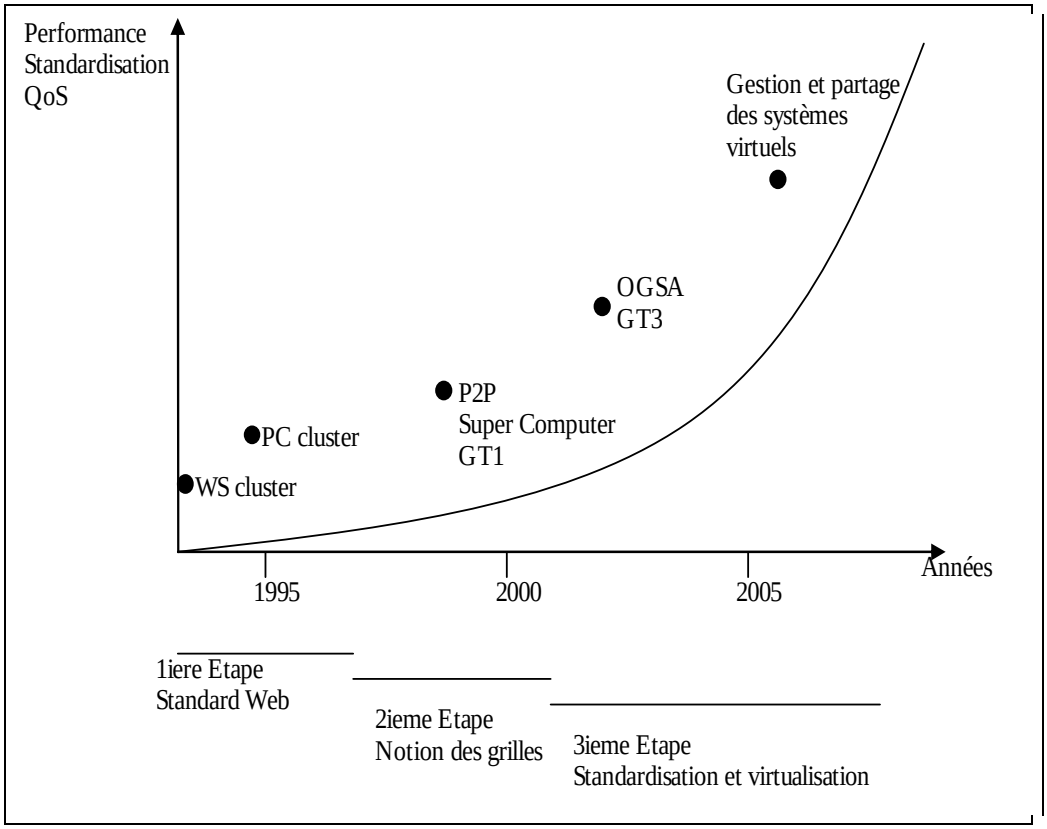


Figure 1-1: Chronologie de l'évolution technologique [5,13,14].

- **Notion d'organisation virtuelle (VO)**

L'organisation virtuelle est un concept qui signifie un ensemble d'individus et d'équipements informatiques, qui partagent des ressources et des services soumis à des politiques de sécurité spécifiant les autorisations de ce partage. On regroupe les utilisateurs qui ont les mêmes besoins au sein d'une VO, cette dernière possède des règles et des stratégies de sécurité adaptées le mieux à ce groupe. La VO peut grouper : une équipe qui travail sur un même projet ; un ensemble de chercheur dans un domaine commun ou avec une technologie commune ; un ensemble d'utilisateurs qui travaillent sur les mêmes ressources. La VO fournit une multitude de services selon les besoins de leurs utilisateurs et en suivant des stratégies de sécurité et des politiques définies par les concepteurs.

Un utilisateur peut être affecté à plus d'une organisation virtuelle, dans ce cas, son authentification se fait qu'auprès une seule, ensuite, il peut accéder au autres VO dont il est membre et donc a le droit de bénéficier des ressources et des services disponibles. La durée de vie des organisations virtuelles est variante, ainsi leurs créations et leurs gestions ouvrent la voie vers d'autres travaux de recherches qui incluent les projets de localisation, d'affectation et de partage de ressources dans une VO.

- **Architecture d'une grille**

La grille est une infrastructure équipée de matériels informatique, des calculateurs, des mécanismes de communication et d'un ensemble de services fournit par un Middleware, ce dernier a comme objectif le contrôle des activités et des interactions dans la grille…

Cette infrastructure peut être visualisée comme une architecture en couche qui illustre les principaux composants d'une grille. La figure 1-2 montre l'architecture en couche d'une grille.

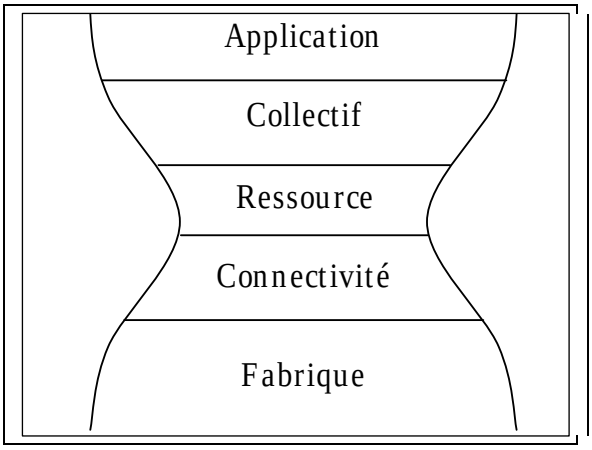


Figure 1-2: Architecture en couche des grilles [1,5].

Chaque couche représente une abstraction d'un ensemble de fonctionnalité de système. Chaque couche peut faire référence aux composants et aux services de n'importe quelle autre couche inférieure.

La couche application au sommet de l'hourglass, représente l'ensemble des applications qui seront exécutées sur la grille. Si ces applications ont été développées sur la structure logicielle de la grille, et adaptées selon ses spécificités, on dit qu'on parle de grid-aware.

La couche collective, se charge de la coordination des ressources. Elle se base sur les protocoles d'information, ces derniers fournissent des services d'annuaire, d'indexation et de recherche et d'autres services pour la gestion des ressources de la grille. La couche collective repose aussi sur les protocoles de gestion pour coordonner l'accès aux différentes ressources.

La couche ressource, possède un ensemble d'informations sur les ressources avec lesquelles elle contrôle l'accès aux différents services et aux ressources locales, elle les mise en disposition et gère leurs partages.

La couche connectivité, responsable de la communication et la sécurisation des différentes interactions sur la grille.

La couche fabrique, située le plus bas du modèle, représente l'infrastructure physique de la grille, elle inclut des ordinateurs, des medias de stockage, des réseaux, et tous autres matériels informatique ou industriels qui peuvent faire partie de la grille.

1.7.1 Couche fabrique

Cette couche est en relation directe avec le matériel qui constitue la grille, elle mis en disposition toutes les ressources partageables. On peut classer les ressources en deux catégories principales, physique contenant des calculateurs, clusters, ressources réseaux, capteurs, annuaires, bases de données, et la seconde catégorie regroupe des ressources logiques tels que des systèmes de fichiers distribués ou des serveurs virtuels, qui nécessitent l'utilisation de leurs propres protocoles de gestion internes indépendamment de la grille.

Lorsque l'une des couches supérieures fait référence à une ressource via une opération de partage comme par exemple la demande d'un emplacement mémoire, la couche fabrique s'interface pour répondre à cette demande, des composants logiciels de la couche agissent directement sur les ressources physiques et logiques et implémentent les opérations spécifiques selon la demande. Les ressources eux mêmes doivent implémenter un mécanisme d'introspection pour fournir les informations qui décrient leurs structures, leurs fonctionnalités et leurs états, ces informations sont nécessaires pour interroger les ressources. Des fonctionnalités intrinsèques de la couche peuvent être citées pour les différents types de ressources:

Ressources de calcul : ce type de ressource doit implémenter des fonctionnalités permettant de récupérer des informations sur les caractéristiques matériels et logiciels, tel que la charge de système, l'état des files d'attente. De même, ces ressources doivent être dotées d'un ensemble de mécanismes capables de lancer des programmes, de contrôler l'état des processus et de gérer les ressources à allouer.

Ressources de stockage : Ces ressources doivent pouvoir envoyer et récupérer des fichiers, lire et écrire des parties de ces fichiers, fournir un mécanisme de gestion de ressources allouées lors de transfert de données comme l'espace mémoire, la bande passante du disque, la bande passante réseau, CPU, charge du système. Des fonctions d'introspection sont utiles pour fournir des informations sur les caractéristiques logicielles et matérielles utilisées durant le transfert des données.

Ressources réseau : doivent fournir des fonctionnalités d'introspection pour déterminer les caractéristiques et la charge de réseau.

1.7.2 Couche connectivité

Fournit des protocoles nécessaires pour l'authentification et la communication des ressources à travers une grille, les protocoles de communication implémentés sont principalement ceux utilisés par Internet comme TCP/IP, UDP, HTTP, DNS, etc. Avec une hétérogénéité croissante des ressources des grilles, un nombre important de protocoles été implémentés, et il reste toujours la possibilité d'incrémenter ce nombre selon les besoins de la communauté et des nouvelles technologies. Les protocoles de communication implémentés permettent les échanges de données entre ressources au niveau de la couche Fabrique.

Les protocoles d'authentification sont implémentés avec les protocoles de communication, ils sont nécessaires pour identifier l'identité des ressources, leurs origines et leurs propriétaires. Des opérations de cryptographie sont possibles, ainsi d'autres techniques d'authentifications sont implémentées afin de sécuriser l'utilisation des ressources, et bénéficier au maximum des services de la grille sans risques.

Un défit est rencontré envers l'implémentation des protocoles dans la couche connectivité. Ainsi, une multitude de protocoles est mise en question, ce qui implique un besoin de standardisation en se basant sur ceux qui déjà existaient, plus précisément avec ceux implémentés par Internet, que se soit au niveau réseau, transport, sécurité ou les autres couches. On peut citer quelques protocoles utiles pour la sécurité des grilles, réutilisés ou basés sur d'autres standards d'Internet.

Single sign-on (SSO): c'est un mécanisme qui simplifie la procédure d'authentification pour l'utilisateur, au lieu d'être identifié à chaque utilisation d'un service de la grille, le SSO lui permet de ne s'authentifier qu'une seule fois. L'identifiant de l'utilisateur et ses attributs seront propagés. De même, certains mécanismes qui implémentent SSO, permettent de fermer toutes les sessions ouvertes de l'utilisateur après sa déconnection.

Délégation : les processus lancés par les utilisateurs, doivent porter les mêmes autorisations et le mêmes droits que leurs propriétaires, ainsi les autres applications et services interrogés par ces processus, doivent avoir les mêmes droits, ce qui est appelé 'délégation des privilèges'.

Relation d'approbation : si un processus fait appel à plusieurs ressources de différents sites, et il n'est authentifier qu'auprès d'un seule, la relation d'approbation assure l'accès à toutes les ressources demandées en faisant confiance aux politiques de sécurité du premier site qui a autorisé l'utilisateur.

Intégration : dans une grille, chaque site ou fournisseur de ressources implémente une solution interne pour sécuriser ses ressources, la politique de sécurité générale de la grille doit être capable de s'interfacer avec toutes les différentes solutions spécifiques adoptées par les sites.

1.7.3 Couche ressources

Cette couche réfère à utiliser tous les services et les protocoles qui nous permettent de gérer les ressources d'un point de vu individuel. Elle utilise les protocoles des deux couches précédentes pour collecter les informations sur les caractéristiques des ressources individuelles. La gestion des ressources implique l'utilisation des services de contrôle, d'énumération et d'initialisation des états des ressources. Dans cette couche, on n'essai pas de gérer les interactions globales entre les ressources de la grille, mais juste avec les ressources individuelles. Ils existent deux classes principales de protocoles implémentées dans cette couche :

Les protocoles d'informations : cette classe de protocoles nous permet d'accéder aux informations des ressources. Les informations à collecter peuvent être statiques comme celles concernant un calculateur, sa puissance, sa capacité de stockage, sa configuration, sa stratégie de sécurité, etc. comme elles peuvent êtres dynamiques comme la charge d'un système, le nombre de processus lancés, l'espace non occupé d'un media de stockage, etc.

Les protocoles de gestion : permettent de négocier l'accès aux ressources partagées, ils nous offrent la possibilité de spécifier des réservations, à voire les conditions de qualité des services et des opérations qui peuvent êtres effectuées comme la création de processus et l'accès aux données distantes. Cette classe de protocoles et de services prend en charge le monitoring des opérations (de surveillance et d'alarmes). Elle contrôle l'exécution des opérations et remonte les erreurs vers les services des couches supérieurs afin de les informer sur l'état des traitements lancés. Les protocoles de gestion de cette classe sont responsables de la bonne exploitation des ressources, tout en vérifiant les stratégies concernant l'utilisation des ressources partagées définies par le concepteur.

Il y a une multitude de protocoles qui peuvent êtres implémentés pour répondre à toutes les opérations concernées par cette couche, mais ils doivent êtres choisis, normalisés ou limités en nombre afin de faciliter leurs intégrations et leurs déploiements dans une variété de contexte et d'environnement.

1.7.4 Couche collectif

Contrairement à la couche précédente, cette couche possède une vue globale des ressources. Elle implémente des services et des protocoles capables de gérer un nombre important de ressources et gérer leurs interactions. Cette couche est responsable de l'ordonnancement et la co-allocation des ressources demandées, elle possède plusieurs informations globales sur les ressources, ces informations lui donnent la possibilité de choisir et d'allouer la ressource la plus appropriée pour la réalisation d'une tâche demandée. Elle s'occupe également de la réplication des données, la récupération et le traitement des alertes remontées par la couche ressources. En d'autre mot, la couche collectif se charge de l'orchestration des ressources disponibles sur la grille.

Dans cette couche, on trouve des services figés, utilisés pour la gestion des ressources. On cite :

– Annuaire : Le service d'annuaire fourni par cette couche joue un rôle important dans la gestion des ressources de la grille. Il est représenté par une base de données contenant toutes les caractéristiques des ressources. Avec ces informations, le courtier localise la ressource la plus appropriée pour répondre à la requête lancée, il consulte l'annuaire et transmit les informations nécessaires pour la localisation physique de la ressource.

– Services d'allocation et d'ordonnancement : afin d'allouer une ressource à un processus, le service allocateur consulte l'annuaire pour trouver celle la plus appropriée. L'Ordonnanceur prend l'initiative de décider quel processus doit être exécuté, et sur quelle ressource, et le moment exacte de son exécution selon la politique de l'Ordonnanceur adoptée.

– Services de contrôle et de diagnostique : ils donnent à l'utilisateur la possibilité de contrôler l'état des ressources et s'assurer de leurs bons fonctionnement.

– Services de gestion de données : les processus lancés ont besoin d'un ensemble de données pour accomplir leurs tâches. Les services de gestion de données permettent la récupération et le transfert de ces portions de données pour êtres exploitées.

– Services de réplication de données : leur objectif est de maximiser les performances de la grille, une réplication est effectuée afin de minimiser le coût et le temps d'accès à un bloc de données.

Plusieurs critères de choix de la ressource appropriée peuvent êtres cités, comme par exemple :

– Des spécifications logicielles et matérielles ont été fournies par la requête ;

– Des droits qui doivent êtres vérifiés pour accéder à une ressource ;

– Répondre à une politique d'équilibrage de charge, le courtier peut affecter une ressource à un processus, selon la charge globale de son organisation, ou selon le coût nécessaire pour son exploitation.

1.7.5 Couche application

Elle contient les applications disponibles sur la grille, cette couche peut interagir directement avec les autres couches inférieures. Les couches Collectif et Ressources sont sollicitées pour localiser les ressources demandées. La couche Connectivité sera l'outil d'authentification. Et la couche Fabrique pour y accéder.

- **Les Intergitiels**

L'Intergitiel est considéré comme l'infrastructure logicielle et l'outil convenable pour une meilleure exploitation d'une grille et une meilleur virtualisation des ressources. Son objectif est de masquer toute hétérogénéité et d'assurer l'interopérabilité, il fournit des applications, des services et des APIs (application programming interfaces), il utilise des standards pour offrir des fonctionnalités requises par la grille tel que la sécurisation des ressources et des utilisateurs, la communication entre entités dans une grille, la gestion des ressources (partage, transfert et exploitations), la découverte des ressources, ainsi que la gestion de leurs états à un instant donné ; d'autre fonctionnalités peuvent être rajoutées selon la nature de l'Intergitiel et le contexte dans lequel il est utilisé, les Intergitiels ont adopté la nature modulaire pour leurs composants afin d'offrir plus de souplesse durant leurs déploiements sur la grille, et pour n'utiliser que les modules nécessaires selon les besoins des utilisateurs. Toute amélioration ou évolution des fonctionnalités de la grille, nécessite une augmentation dans un ou plusieurs modules des Intergitiels. On cite les Intergitiels les plus connus :

1.8.1 Globus

Globus a été développé sur la base de I-WAY, en plus de supporter des applications demandant des performances, il peut gérer des organisations virtuelles. Développé aux Etats-Unis dans le laboratoire Nationale de l'Argonne par l'équipe de Ian Foster. Il fournit une infrastructure logicielle permettant aux applications de manipuler les ressources hétérogènes, géographiquement distribuées comme étant une machine virtuelle unique. La boite à outil Globus a fournit les services de base de grille comme les services de sécurité, les services d'informations, de communications et de gestion des données. La version 4 de la boite à outil (Toolkit) Globus a implémenté l'architecture OGSA pour permettre l'interopérabilité entre les ressources de la grille, les principaux modules de Globus sont :

- GSI (Globus Security Infrastructure) qui est un module d'authentification utilisant la politique clé publique et les certificats X.509, ainsi que plusieurs standards tel que SSO et TLS… ;
- MDS (Monitoring and Discovery Service) : responsable de la collection des informations des différentes ressources de la grille. Avec un langage de spécification, ce module est interrogé par plusieurs autres services;

– GRAM (Globus Resource Allocation Manager) : il est responsable d'allouer des ressources distantes pour y soumettre et surveille l'exécution des tâches. Ainsi, il fonctionne via un co-allocateur dynamique de ressources (DUROC) ;

– GridFTP (Grid File Transport Protocol) : responsable de la gestion des données, il permet le transfert sécurisé des fichiers sur la grille. Il est à la base de fonctionnement d'autres services telque RFT(Reliable File Transfert) et RLS (Replica Location Service).

1.8.2 Unicore (UNiform Interface to Computer Resources)

C'est un projet Européen qui a pour objectif d'assurer un accès sécurisé et uniforme à des plateformes de calcul intensif [15], il fournit des fonctionnalités pour la gestion de données, la gestion des tâches, de sécurité et d'informations. L'Intergitiel implémente une architecture client serveur basée sur le modèle trois tierce, composée de trois couches [16]. Une couche utilisateur, représentée par un client UNICORE qui possède une interface graphique pour visualiser tous les services de la couche serveur, cette dernière utilise les AOJ (Abstract Job Objects) pour la communication avec la Couche client UNICORE. Un AOJ contient des données et des descriptions de tâches de calcul, les tâches sont lancées sur la troisième couche de l'Intergitiel, elle est représentée par les plateformes et les sites sur lesquels UNICORE est installé. L'envoi et la réception des AOJs se fait via la couche UPL (UNICORE Protocol Layer), elle assure la sécurité des interactions dans la grille. Pour lancer et gérer les tâches sur n'importe quelle site connecté à la grille, UNICORE fournit une interface de ligne de commande (CLI).

1.8.3 Legion

Un 'metasystem' à base d'objet, résolument paire à paire, développé à l'Université de Virginia. Il fournit aux utilisateurs de la grille une infrastructure logicielle permettant aux systèmes hétérogènes, distribués et performants d'interagir uniformément. Contrairement à Globus, l'Intergitiel Legion est vu comme étant un objet, et ses modules aussi sont considérés comme des objets, cela facilite beaucoup la manipulation des relations, tel que l'héritage, l'instanciation, le polymorphisme. Legion s'interface entre le système d'exploitation des utilisateurs et les ressources distribuées sur la grille. Chaque utilisateur aura l'impression de ne voir que ses propres ressources, mais en réalité, il accède aux différentes ressources réparties sur le réseau, tout en respectant les protocoles de sécurité et d'accès définis par l'Intergitiel.

- **Open Grid Services Architecture (OGSA), Open Grid Services Infrastructure (OGSI), Web Services (WS)**

1.9.1 Présentation

L'apparition de l'architecture OGSA a mené vers de grandes évolutions dans le domaine des grilles, OGSA est une norme adoptée en 2002 par Global Grid Forum (GGF). L'architecture OGSA se base principalement sur les technologies et les concepts des Services Web. Elle considère la grille comme un ensemble de services. Un service de grille est considéré comme un Service Web qui fournit un ensemble d'interfaces conformes aux standards utilisés [17]. Les principales composantes de l'architecture OGSA sont l'Open Grid Services Infrastructure (OGSI), les services et les schémas OGSA comme démontré dans la figure 1-3. OGSA repose sur les standards utilisés par les services Web et les paradigmes qui sont largement déployés dans le domaine des systèmes distribués, et qui fournissent des mécanismes de base pour invoquer et décrire les services de grille. Les services OGSA doivent êtres déployés tout au long de la grille, et doivent communiquer entre eux via des standards de communication.

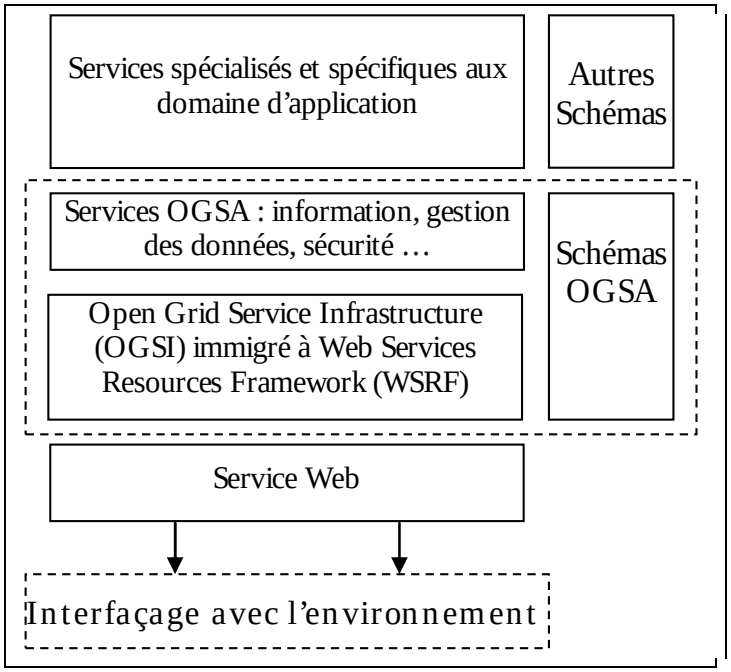


Figure 1-3: Eléments de base de l'architecture OGSA [5].

Les Standards utilisés par les Services Web ne fournissent pas les informations sémantiques sur les services tel que comment un service est créé, sa durée de vie, comment il gère les exceptions, etc. Cet ensemble de comportement des services doit être standardisé, et à partir de ce point, on peut voir l'utilité de OGSI. Un service qui suit les spécifications conformes à OGSI est appelé un Service de grille. OGSI définit un ensemble de composants pour les

systèmes distribués avec des interfaces standards, et des comportements qui décrivent les attributs sémantiques des services.

Pour qu'un Service Web se qualifie d'être un Service de grille, OGSA définit trois conditions [18] : (i) il doit être une instance d'une implémentation de service, (ii) il doit avoir un Grid Services Handle (GSH) qui est en général l'unique identificateur 'URI' d'une instance d'un Service de grille, c'est avec le GSH qu'on localise le 'Grid Service Reference' (GSR) qui est représenté par un document WSDL décrivant les propriétés de l'instance, (iii) il doit implémenter un port appelé 'Grid Service' pour pouvoir :

– Localier des informations de l'instance du Service de grille qui ne sont pas fournies par le GSR tel que l'état de service, l'environnement d'exécution de l'instance et d'autres informations sémantiques.

– Détruire l'instance de Service de grille.

– Définir la durée de vie d'une instance.

1.9.2 Services Web

A. Définitions

On trouve plusieurs définitions pour les services Web :

Définition fournie par IBM : Un service Web est une interface qui décrit une collection d'opérations accessibles sur un réseau via des messages standards au format XML. Des Services Web accomplissent une tâche spécifique ou un ensemble de tâches. Un Service Web est décrit en utilisant la notation XML formelle, appelée également le descripteur de Service qui fournit tous les détails nécessaires, y compris des formats de message (qui détaillent les opérations) et protocoles de transport.

La nature de l'interface masque les détails de mise en œuvre de service pour que ce dernier puisse être utilisé indépendamment du matériel ou du logiciel de la plateforme sur laquelle il est mis en œuvre, et indépendamment de la langue de programmation avec laquelle il est implémenté. Cela permet et encourage les applications basées sur les services Web d'être faiblement couplées, orientées composant, indépendantes des implémentations des technologies. Les services Web peuvent être utilisés seuls ou avec d'autres services.

Une des définitions de Microsoft : Un service Web est une unité logique d'application, il fournit des données et des services. Les applications accèdent aux services Web via des

protocoles web et des formats de données standards comme HTTP, XML et SOAP, sans besoin de savoir comment chaque service Web est mis en œuvre. Les Services Web combinent les meilleurs aspects de développement à base de composant et de Web, et ils sont la fondation des modèles de programmation de Microsoft .NET.

Définition fournit par Sun : Les Services Web sont des composants logiciels qui peuvent êtres découverts, combinés et recombinés pour fournir une solution du problème/demande de l'utilisateur. Le langage Java et XML sont les headers des technologies Services Web.

Les services Web définissent des applications en s'appuyant sur (XML) pour mapper des programmes, objets ou des Bases de données. En XML, Les Services Web définissent le format des messages à échanger, spécifient l'interface de la destination, décrit la procédure du mappage et définissent le mécanisme de déploiement et de découverte des services Web sur les serveurs.

Un service est un acteur logiciel qui accomplit certaines opérations bien définies (i.e. fournir des services), il peut être invoqué dans le contexte d'un grand nombre d'applications. Les utilisateurs sont concernés juste par les descriptions des interfaces que fournit le service. Le service possède des interfaces réseau et de communication, et il est accessible par des standards de protocoles et de formats de données [24].

Les services Web peuvent être définis selon plusieurs points de vue et selon leurs contextes d'utilisation. Les principales caractéristiques d'un service Web sont:

– Décrit en utilisant un langage de description de services WSDL ;

– Publié dans un registre de services pour être localisé ;

– Découvert et invoqué via des APIs et des standards ;

– Interagit avec d'autre Services Web tout en masquant les détails de leurs implémentations, ou les contraintes des plateformes hétérogènes sur lesquelles il est déployé.

B. Architecture des Services Web :

En reposant sur la notation XML, l'architecture des services Web est composée de quatre parties principales comme montrées dans la figure 1-4.

La première partie permet la découverte et l'agrégation des Services Web, elle définit un registre connu avec le nom UDDI (Universal Description, Discovery, and Integration (Wrox,

2001)) qui est un annuaire utilisé pour sauvegarder et localiser les interfaces de tous les services Web déployés.

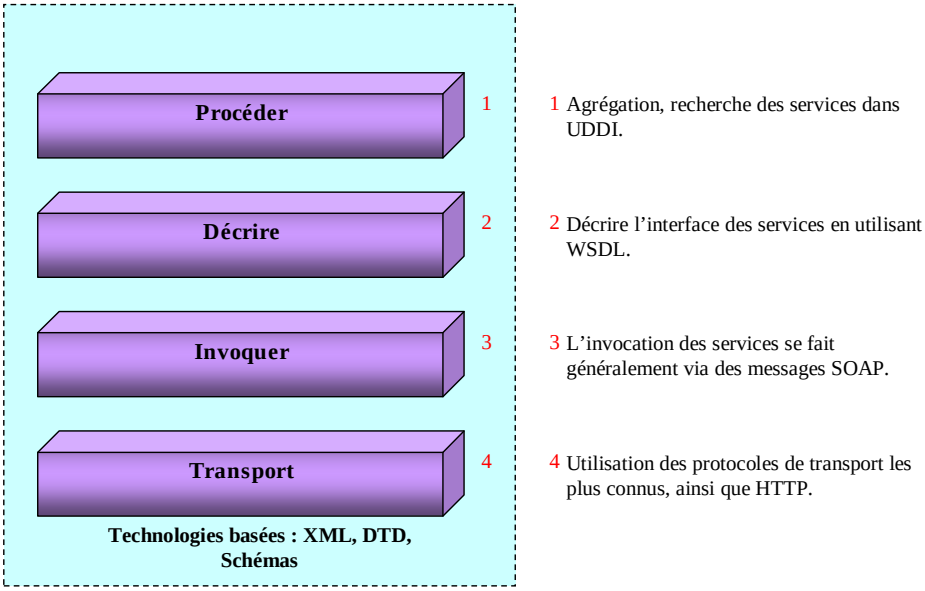


Figure 1-4:Architecture des Services Web [3,19].

La seconde partie de l'architecture utilise le standard WSDL (Web Services Description Language (OASIS, 2004)). WSDL fournit l'abstraction la plus fondamentale des services Web, il construit l'interface qui sera exposé aux autres services et à travers lequel il sera mappé.

La troisième partie de l'architecture implémente le standard SOAP (Simple Object Access Protocol (W3C, 2001)), c'est un protocole basé sur la notation formelle XML et qui définit la structure des messages à échanger entre les services.

La quatrième partie fournit les interfaces et les standards des protocoles les plus reconnus de la couche transport dans la technologie Web tel que HTTP, SMTP, FTP, JMS, etc.

C. Invocation des Services Web :

L'invocation d'un service Web passe par un ensemble d'étapes bien définies, elle se fait essentiellement via les standards SOAP et WSDL. Si on voit l'architecture des services Web comme une de type Client-Serveur, on énumère les étapes d'invocation suivantes :

– Le côté serveur contient un ensemble de services déployés, il possède d'un service conçu spécialement pour la découverte et la supervision des autres services. Le client peut être vu comme une application ou un autre service, il lance une requête au service de découverte pour demander l'accès au service demandé.

– Le service de découverte possède toutes les informations des autres services, il répond au client en l'envoyant l'URI du service demandé.

– Le client lance des requêtes au service demandé pour connaître toutes les fonctionnalités qu'il peut offrir, sa description et la méthode avec laquelle il peut être invoqué.

– Le service demandé fournit sa description et d'autres informations utiles en utilisant le langage de description WSDL.

– A présent, le client possède les informations qui lui permettent d'invoquer le service Web demandé, il lance des requêtes via des messages SOAP.

– Enfin, le service Web demandé repend aux requêtes du client, toujours en utilisant le format SOAP.

1.9.3 Architecture Orientée Service (SOA)

Service-Oriented Architectures SOA présente trois types de service, un demandeur, un fournisseur et un marqueur de services, comme montré dans la figure 1-5. Un service peut jouer plusieurs rôles, il peut être à la fois un demandeur et un fournisseur de services.

Le fournisseur de service crée les descriptions des WS, il les publie vers un ou plusieurs services marqueurs, et il reçoit les messages d'invocation venants des autres services demandeurs et répond à leurs demandes en leurs fournissant des références sur les WS publiés.

Le service demandeur est responsable de trouver une description de WS publié auprès d'un ou plusieurs marqueurs de service, ce dernier utilise ces descriptions pour invoquer les services indiqués par le service fournisseur.

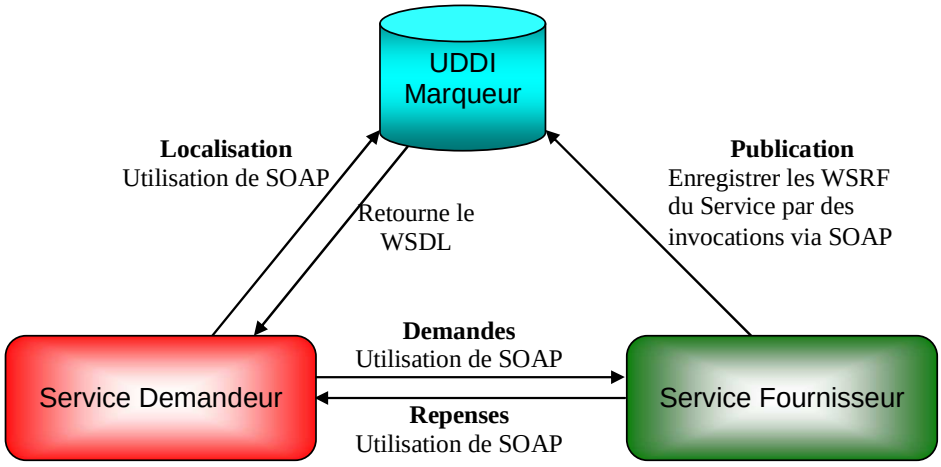


Figure 1-5: Service-oriented architecture (SOA).

Le service marqueur est responsable de publier les descriptions des Services Web qu'il marque, il donne la possibilité aux services demandeurs de chercher dans les collections contenants les descriptions des Services, il est l'intermédiaire entre les services demandeurs et les autres fournisseurs de services, une fois la relation est obtenue, l'interaction sera direct avec les demandeurs et les fournisseurs des services.

Les opérations possibles entre les trois types de services sont la publication, la recherche et la connexion des services. La publication est effectuée par les services fournisseurs, ils publient les caractéristiques, les descriptions et les fonctionnalités des WS dans le registre UDDI ; la seconde opération est la recherche ou la localisation, elle est effectuée par les services demandeurs pour localiser un WS fournissant des fonctionnalités précises ; la dernière opération est représentée par la connexion et l'interaction entre WS, la connexion est effectuée par les services demandeurs pour communiquer avec le WS trouvé et fournit par le service fournisseur, cette communication est faite via les standards SOAP et WSDL.

1.9.4 Web Services Resources Framework (WSRF)

C'est un ensemble de spécifications de WS, il introduit la notion des Ressources Services Web (Ressource-WS) pour gérer les informations sur l'état de la ressource. Une ressource à état associée à un WS est connue comme étant Ressource-WS [20]. Les spécifications Services Web sont :

Propriétés des Ressources-WS (WS-ResourceProperties): Elles décrivent le concept des Ressources-WS et comment des ressources à état sont associées à des Services Web. De même, elles décrivent comment les propriétés d'un Service sont trouvées, modifiées, et supprimées d'une ressource par l'utilisation des descriptions de ses données internes.

Le but des WS-ResourceProperties est de standardiser les concepts [21], terminologies, opérations. WSDL et XML sont utilisés pour définir la projection des propriétés des ressources, il est associé à l'interface du WS ainsi qu'aux messages définissants les demandes des services.

Durée de vie des Ressources-WS (WS-ResourceLifetime) : C'est la période entre la création de l'instance de Ressource-WS et sa destruction. Les spécifications WS-ResourceLifetime tentent à standardiser la façon avec laquelle l'instance de la ressources-WS est détruite et la façon de gérer sa durée de vie. Par contre, elle ne spécifie pas la façon dont elle est créée. La ressource est allouée à un service demandeur pour une période bien définie, et après l'écoulement du temps prévu, la destruction de cette instance de ressource

est effectuée par échange de messages comme décrit dans les spécifications fournies par WS-ResourceLifetime. Parfois l'instance de la ressource doit s'autodétruite dans le cas d'une déconnexion du client. Ainsi, les spécifications WS-ResourceLifetime décrivent la destruction de l'instance qui doit être faite automatiquement sans le besoin d'échanger des messages. Les spécifications décrivent aussi la manière dont une instance peut prolonger sa durée de vie pour répondre aux demandes des services.

Notification-WS (WS-Notification) : C'est l'outil standard pour la communication inter-objets. Les WS se basent sur l'implémentation des échanges des messages standards. Cette spécification permet au WS d'être un producteur de notifications, et à certains d'autres d'être notifiés (consommateurs). De plus, elle fournit des standards pour notifier les consommateurs lors d'un changement dans une Ressource-WS.

Exceptions-WS (WS-BaseFaults) : C'est un ensemble de spécifications permettant de standardiser la base des exceptions-WS qui peuvent êtres produite dans l'environnement. Elles standardisent l'utilisation des concepts, terminologies, WSDL et XML par les interfaces des Services Web. Les messages communs des erreurs qui peuvent êtres produites simplifient la remonte, la compréhension et le traitement de l'exception par les demandeurs de services.

Groupe de Service Web (WS-ServiceGroup) : Afin de construire un service de haut niveau pour la gestion des autres services de l'environnement, on a besoins d'un ensemble de primitives pour effectuer des opérations basiques sur une collection de service, comme par exemple ajouter, supprimer ou trouver un service dans un groupe. Les spécifications WS-ServiceGroup définissent la façon avec laquelle les Services Web et les Ressources-WS peuvent être regroupés ensembles, un Groupe Service est une Ressource-WS qui représente une collection de Services Web.

Renouveler les références-WS (WS-RenewableReferences) : C'est un ensemble de spécifications, permettant de standardiser les mécanismes des Services Web avec lesquels ils renouvellent les références des instances (endpoint references) quand elles ne seront plus valides. Ces spécifications fournissent un mécanisme similaire à l'utilisation des GSH et GSR introduits par OGSI.

Les efforts du groupe GGF convergent vers l'adaptation de WSRF au lieu d'OGSI pour la construction des applications de grille orientées service. OGSA a utilisé l'infrastructure OGSI pour définir des spécifications techniques à implémenter par les interfaces des noyaux des

Services de grille, cette implémentation été mise en œuvre par la boite à outils Globus à partir de sa version 3 (GT3). Ci-dessous l'architecture OGSA implémentant WSRF.

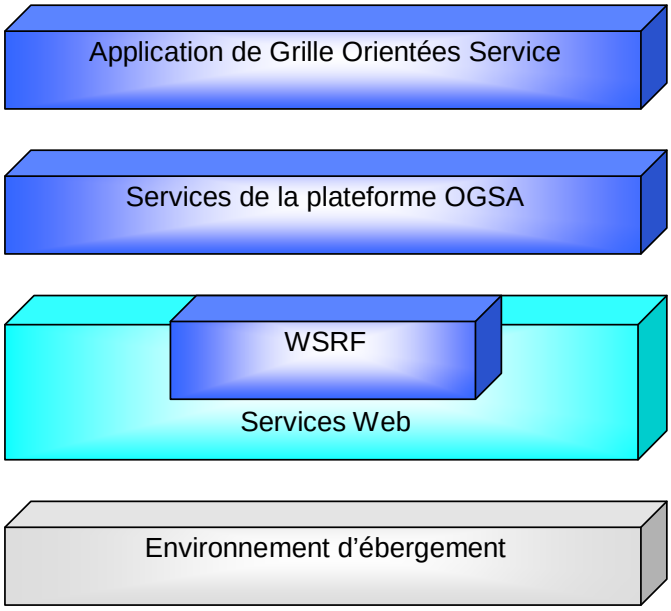


Figure 1-6: L'architecture OGSA implémente WSRF [5].

WSRF fournit des standards pour représenter les Services Web comme des ressources à états, OGSA l'a utilisé comme une infrastructure pour développer des Services de grille, et cette implémentation de WSRF été mise en œuvre par Globus qui sera le sujet de la partie suivante. Les services de OGSA, sont des services à base de grille, utilisés pour le lancement des jobs, authentification des utilisateurs, réplication et transfert et accès sécurisé aux données, les Services WSRF sont le noyau des Services de grille, ils sont utilisés pour la création, la destruction et la gestion des Ressources-WS à état. La figure 1-7 montre la relation entre WSRF, OGSA et les Services Web pour construire un Service Web à état.

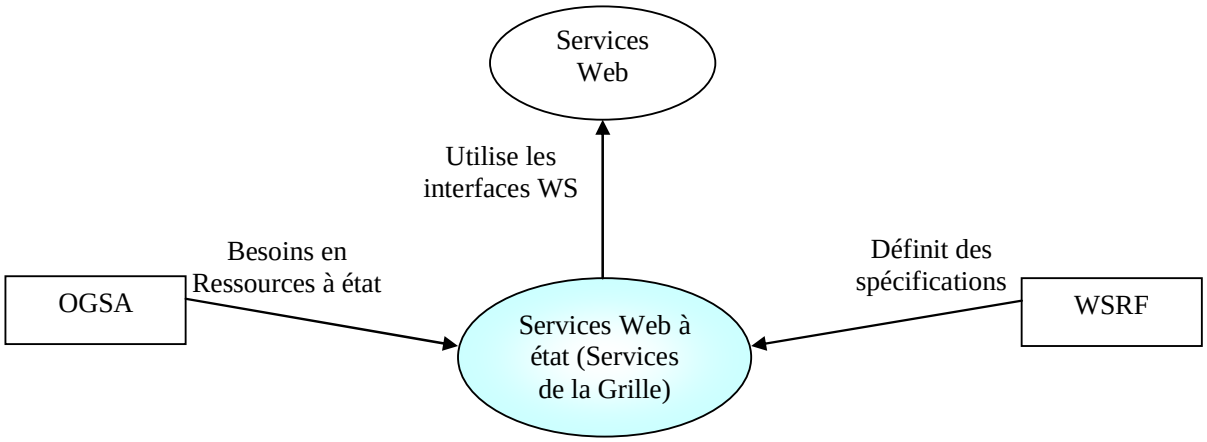


Figure 1-7:Relation entre WSRF, OGSA et les Services Web [3].

1.9.5 Conclusion

Les origines du Grid Computing sont assez floues, aux alentours des années 70. Certains disent que le précurseur des grilles de calcul est la société Apple, plus précisément l'entreprise NeXT. D'autres disent que l'idée serait venue de trois personnes, du Docteur en Mathématiques et en Informatique Ian Foster, de Monsieur Carl Kesselman chercheur en informatique et de Steve Tuecke ingénieur en informatique. Ces trois sont surnommées «fathers of the Grid » et sont à l'origine de « The Globus Alliance ».

Les grilles informatiques sont une technologie qui n'est pas récente, elles représentent une fabuleuse alternative au calcul intensif classique, réalisé par des clusters.

Toutefois, cette approche reste encore dans des phases de prototypage, bien que certains projets, comme Globus Alliance, essaient de surmonter ce problème ainsi que d'autres problèmes liés à la standardisation des protocoles et des politiques utilisés.

Certaines entreprises ont vite compris l'intérêt de ce concept. En effet, certaines sociétés commencent à entrevoir des possibilités de standardisation, tels que Oracle avec sa base 10G (G pour grille), IBM avec World Grid Community (qui pourrait, à terme, devenir un principe adoptable pour les entreprise aux vues de son succès et de sa stabilité).

Globus.

2. Globus Toolkit 4

2. 1 Introduction

L'alliance Globus est un membre actif dans le Global Grid Forum (GGF), ce dernier vise à standardiser les technologies de la grille. Globus Toolkit est une collection de solutions [26], destiné pour résoudre les problèmes fréquemment rencontrés dans la construction des applications collaboratives distribuées, les problèmes fréquemment rencontrés sont en relation avec l'hétérogénéité, la sécurité et la standardisation.

Globus Toolkit est une boite à outil avec une architecture ouverte, à base de communauté (community-based) et à code source libre (open source) [22]. Cela a pour but d'encourager les contributions et l'adoption du Toolkit, ainsi que pour des considérations à prendre dans la standardisation des solutions. Globus fournit un ensemble de services et de librairies logicielles [25] qui supportent les applications déployées sur la grille. Globus assure la communication et la gestion des données en toute sécurité, il fournit un système d'information pour le suivi de changement de l'état des ressources et pour la supervision de toutes interactions intra-grille, De même, il offre un mécanisme de gestion des exceptions. Le code source de Globus peut être réutilisé et adapté selon les besoins des utilisateurs et des applications, ses services sont regroupés dans des modules indépendants l'un à l'autre et personnalisés selon le cas de leur utilisation.

Globus est développé principalement pour répondre aux besoins de la collaboration. En faite, l'hétérogénéité été un obstacle pour collaborer un ensemble de machines hétérogènes au niveau matériel qu'au niveau logiciel. Les interfaces et les services que fournit Globus, rendent la collaboration plus facile, ils permettent aux utilisateurs d'accéder aux ressources distantes comme si elles sont en local tout en masquant l'hétérogénéité de l'environnement.

Globus a passé par des étapes durant son évolution, dans cette partie, on va présenter la boite à outils dans sa version 4. GT4 a utilisé OGSA et a remplacé l'architecture OGSI, ce changement été réalisé par l'intégration des spécifications WSRF afin de fournir des ressources à états (Ressources-WS). Ci-dessus, la figure 2-1 qui illustre la relation entre Globus, OGSA, WSRF et les services Web.

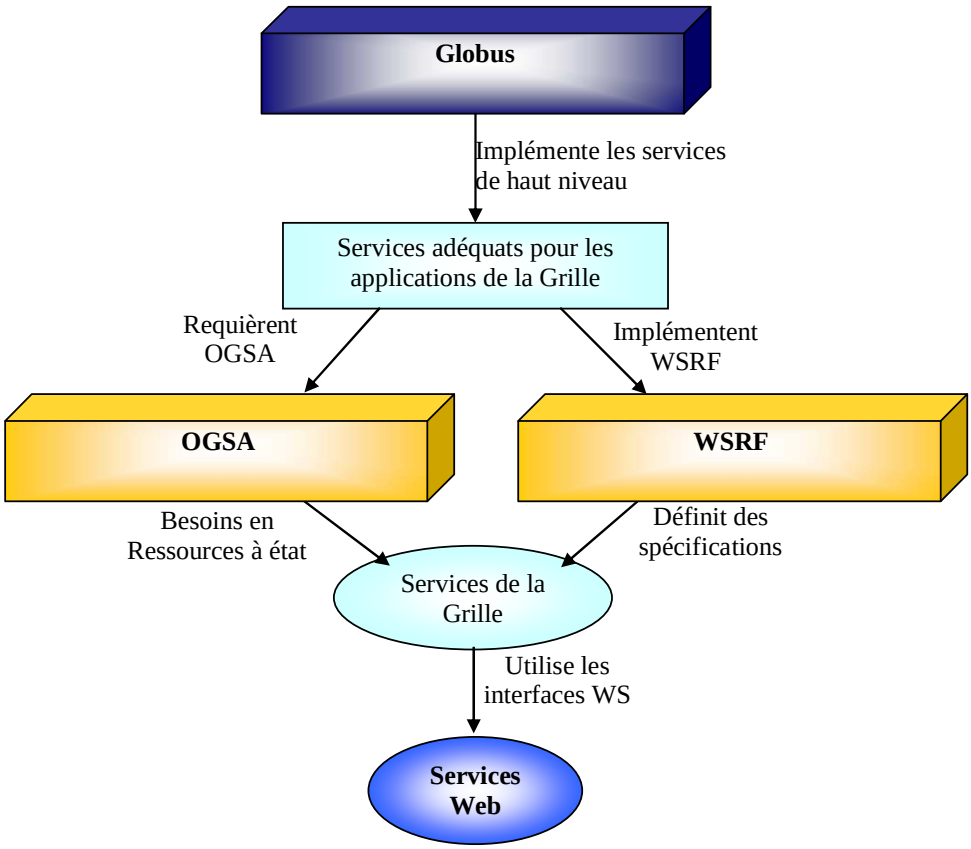


Figure 2-1:Relation entre GT4, WSRF, OGSA et les Services Web [3].

Le développement de Globus Toolkit a commencé à la fin des années 90, et il est maintenant dans sa version 4.2. Globus est un Intergitiel modulaire, où chaque module englobe des composants qui utilisent les spécifications WSRF (WS Component) ou pas (Non-WS Component). Tous les composants sont mis en œuvre pour fournir des services requis par la grille, ils suivent des standards dans leurs implémentations et durant leurs utilisations. Ces composants sont une implémentation de l'architecture OGSA en introduisant les spécifications WSRF. La figure 2-2 montre les cinq modules de Globus, qui sont:

a. Core runtime : Une infrastructure fournit des librairies et des outils pour la construction de nouveaux WS et Non-WS Services pour la grille. Les langages de programmation supportés sont :

– C avec le C WS CORE
– Python avec le Python WS CORE
– Java avec le Java WS CORE

Pour le langage Java, il existe l'outil Java Commodity Grid Kit (COG Kit) qui fournit un niveau d'abstraction à l'API de Globus Toolkit pour un usage plus facile. Il permet aux utilisateurs de grille, aux développeurs d'applications, aux administrateurs de grid d'utiliser, de

programmer et d'administrer les grilles à l'aide d'un Framework de plus haut niveau. Une capture d'écran de l'interface graphique de COG Kit installé pour des raisons de test, est montrée dans l'Annexe A.

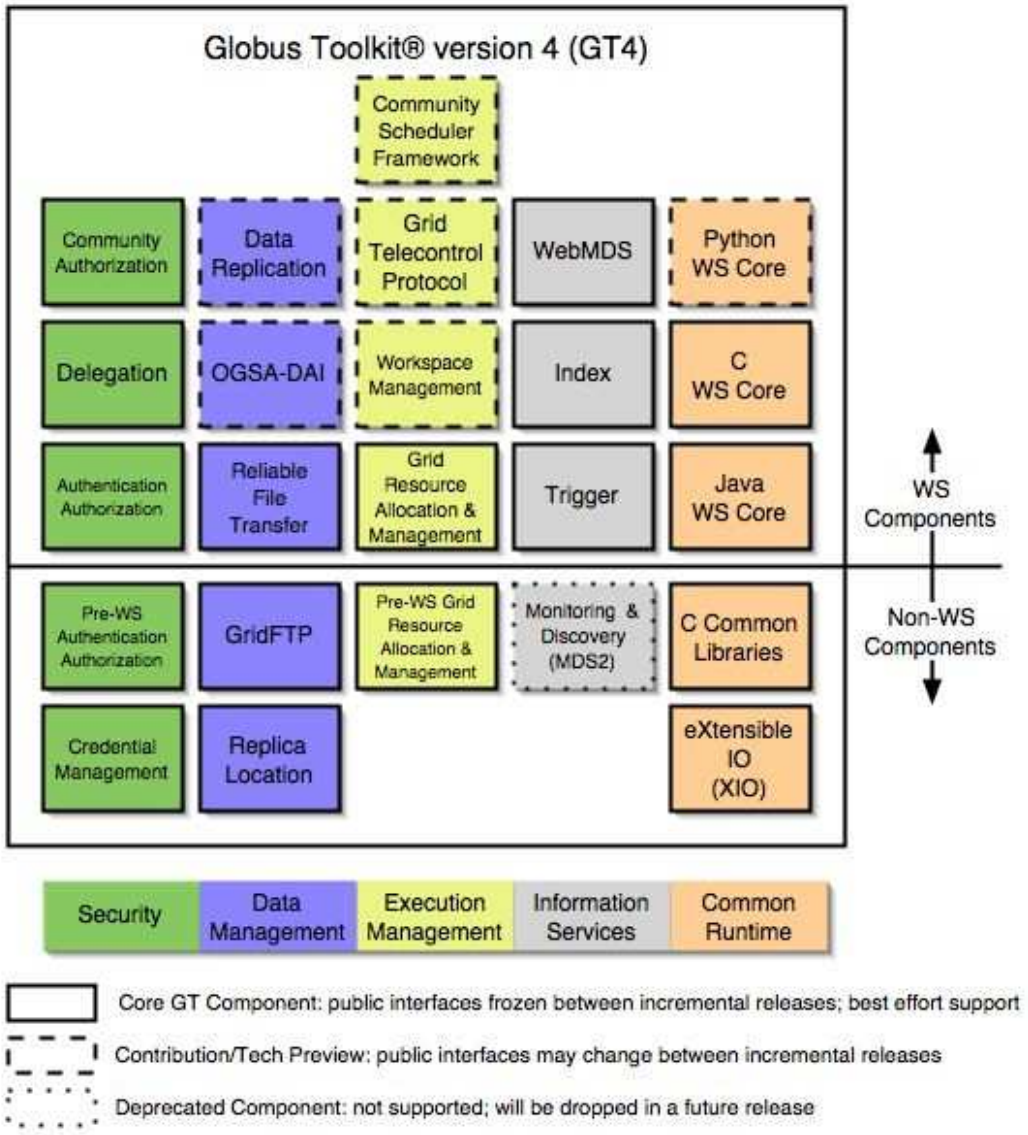


Figure 2-2 modules de Globus Toolkit 4 [23].

b. Sécurité : Un module qui Fournit des politiques uniformes pour différents systèmes de sécurité.

c. Gestion d'exécution : Son objectif est de gérer le déploiement et l'exécution des services sur la grille.

d. Gestion de données : Responsable de la découverte, le transfert, et l'accès aux données distantes.

e. Système d'information : utilisé pour la découverte et la supervision dynamique des services déployés ainsi que les différentes ressources de la grille.

2. 2 Architecture de Globus

Globus offre une collection de Service de grille qui suit les principes de l'architecture OGSA dans leurs implémentations. L'architecture de Globus a évolué avec l'évolution des technologies Web, ces derniers peuvent être utilisées pour améliorer et répondre aux besoins des grilles, avec l'introduction des spécifications WSRF, l'évolution de l'architecture de l'Intergitiel Globus a convergé vers celle implémentée par le Toolkit dans sa version 4.

On prend une vue Client-Serveur [25] de l'architecture de Globus comme montré par la figure 2-3, il est composé de trois ensembles de composants :

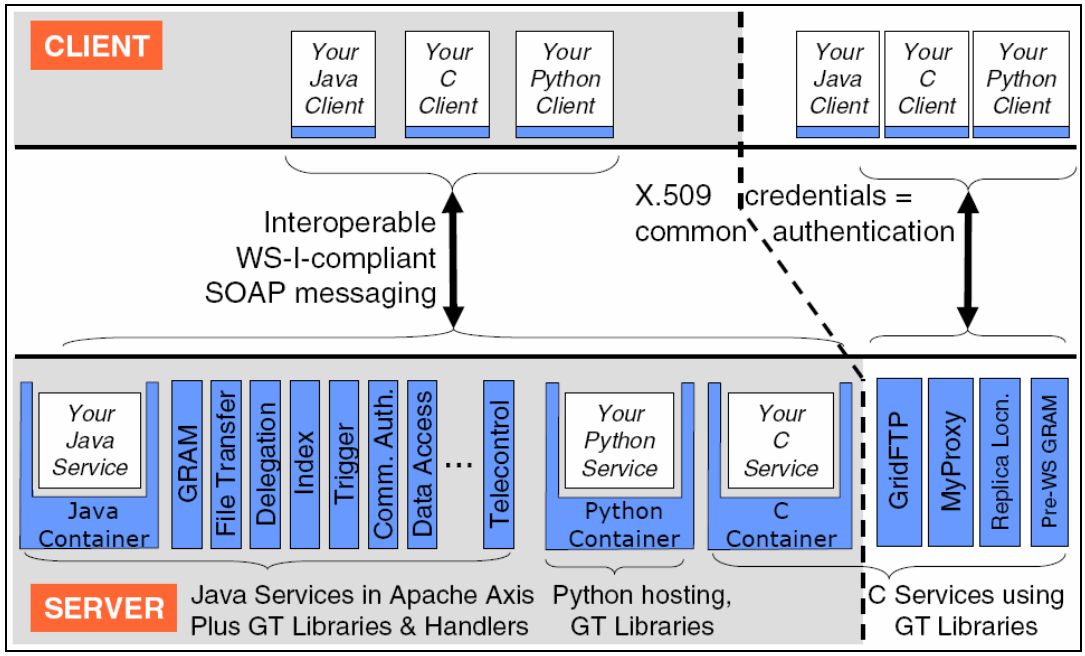


Figure 2-3 : Vue Client/Serveur de l'architecture de Globus 4[27].

i. La partie basse de l'architecture représente un ensemble d'implémentations de Services, ces derniers sont fournis pour la gestion de la grille, comme le service GRAM pour la gestion des exécutions, le service de sécurité (GSI), des services pour la gestion des données (GridFtp, RFT, RLS, DRS) et d'autres pour construire le système d'information de la grille, ils sont responsables de la découverte et la supervision des informations concernant tous les services déployés dans la grille comme le module d'information MDS4 qui fournit les services d'indexation et de recherche (Index, Trigger, WebMDS), gestion des certificats (MyProxy, Delegation, SimpleCA), et la gestion des instruments (GTCP). La plupart des services cités sont développés en Java.

ii. Trois conteneurs pour héberger les services des utilisateurs écrits en Java, Python et le langage C. ces conteneurs fournissent des implémentations de sécurité, de gestion, et des mécanismes requis dans la construction des Services.

iii. Un ensemble de librairies offertes au client pour lui permettre de programmer en Java, C et Python, afin d'invoquer les opérations fournies par Globus ou par d'autres services développés par l'utilisateur.

L'utilisateur peut interroger les services avec des abstractions et des mécanismes uniformes, cette uniformité facilite la construction des systèmes complexes et inter-opérables, les principales règles sont (1) l'utilisation des messages SOAP pour la communication entre les services. (2) mécanisme de sécurité uniforme et une infrastructure de messages permettant l'interopérabilité entre les différentes applications et les services. (3) un système de certification uniforme, solide et à base de standards permettant d'assurer l'accès sécurisé aux différents systèmes d'autorisation. (4) les conteneurs et les services implémentent des comportements et des interfaces standards pour la représentation de l'état, l'accès, l'inscription, ils facilitent la découverte et la supervision des services déployés. (5) l'implémentation d'abstractions et d'interfaces communs pour la gestion des durées de vie et la destruction des services.

2. 3 Modules de Globus

Dans cette section, on va présenter les quatre modules principaux de Globus concernant la gestion de sécurité, la gestion des données, le système d'information et la gestion des exécutions. La figure 2-4, montre les quatre modules de Globus, où celui de la sécurité représente leur base.

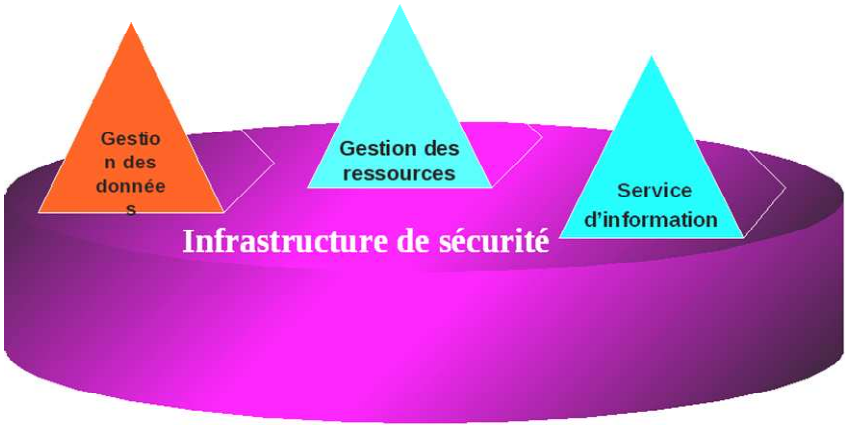


Figure 2-4: Les quatre modules de Globus.

2.3.1 Module de gestion de données

La gestion de données dans les grilles est un problème complexe, un seule service ne peut absolument pas résoudre ce problème, le Toolkit Globus a développé un ensemble de services qui peuvent être utilisés seuls ou en combinaison avec d'autres services.

A. **GridFTP**

extension du protocole File Transfert Protocol (FTP), il fournit des outils et des librairies pour effectuer un transfert à niveau élevé de performance (memory-to-memory and disk-to-disk), transfert sécurisé et consistant, ainsi que nouvelles fonctionnalités de sécurité (GridFTP qui supporte GSS-API Grid Security Infrastructure (GSI) et l'authentification Kerberos et les mécanismes de sécurité SHH [29]) ont été ajoutées dans la couche connectivité, et encore pour gérer les transferts en parallèle sur plusieurs canaux pour maximiser les débits. Cinq raisons [28] pour lesquelles le protocole GridFTP est adopté par le projet Globus : (1) il permet l'introduction d'un troisième élément dans les transferts de données (third-party transfers) qui se traduit par la réalisation des transferts entre deux hôtes finaux (end hosts) à partir d'un troisième hôte, (2) il utilise des standards qui le qualifie d'être inter-opérable avec une variété de protocoles connus par L'Internet Engineering Task Force (IETF), Il fournit une architecture bien définie pour les extensions des protocoles tel que FTP, (4) une grande partie des protocoles définis par l'IETF ont été utilisés, (5) il ajout des nouvelles fonctionnalités qui convient au mieux au transfert distribué dans les grilles :

- Contrôle de transfert de données Third-party: il permet à un utilisateur ou une application dans un site donné d'effectuer, gérer et contrôler le transfert de données entre deux hôtes distants de différents sites.
- Authentification, intégrité de données, confidentialité de données : Grid-FTP supporte Generic Security Services (GSS) et des API d'authentification de contrôle de canaux de transfert, il supporte aussi le contrôle utilisateur du niveau d'intégrité et de confidentialité.
- Transfert ligné de données (Striped data transfer): les données doivent être transférées en plusieurs reprises entre les serveurs. Outre, GridFTP définit des extensions de protocoles qui supportent le transfert entre plusieurs serveurs de données fractionnées.
- Le transfert parallèle de données : GridFTP supporte le parallélisme via des commandes FTP ou par des extensions canaux.

- Transfert partiel des fichiers : le transfert partiel des fichiers est d'une importance élevée, il est utile dans le cas où une application veut accéder à un fichier volumineux, FTP permet le transfert d'une partie d'un fichier, cette possibilité permet au GridFTP de supporter le transfert régional des fichiers.
- Négociation automatique *TCP buffer/window sizes*: la négociation automatique (cache TCP/ tailles fichiers) augmente les performances de transfert des données, ce qui est faux pour une négociation manuelle qui n'est pas banale et difficile pour un utilisateur qui n'est pas expert, GridFTP utilise des commandes FTP et des protocoles canaux pour supporter la négociation manuelle et automatique de la taille du cache TCP par rapport à la taille du fichier à transférer.
- Supporte le transfert consistant de données et le rétablissement de l'état de transfert : GridFTP utilise le protocole FTP qui permet de redémarrer les transferts échoués pour implémenter la notion de rétablissement des états des transferts de données, ce mécanisme doit mettre en place une solide infrastructure pour gérer les exceptions qui peuvent se produire durant un transfert.

B. Reliable File Transfer (RFT)

Utilisé pour gérer plusieurs transferts consistants et effectuer des transferts third-party via GridFTP. Il utilise une base de données pour sauvegarder les états de tous les transferts afin de pouvoir restaurer et rétablir l'état d'un transfert dans le cas d'une exception remontée.

Le RFT fonctionne sur la base de GridFTP, il utilise leur librairie et hérite de ses performances et de ses fonctionnalités tel que le redémarrage de transfert. Ainsi, le transfert de fichier avec GridFTP nécessite que le client reste actif jusqu'à la fin de transfert, cet inconvénient a motivé le développement du RFT qui n'est pas basé sur la présence de l'utilisateur, ce dernier peut lancer le transfert d'un ou plusieurs fichiers, et en cas de déconnexion du client ou une exception, le RFT reprend les transferts à partir d'un état sauvegardé dans sa base de données. La figure 2-5 montre l'architecture du service RFT.

Niveau application de RFT: le client lance le transfert des fichiers via le RFT. Le client possède un thread, lorsque ce dernier détecte l'échec ou l'altération du transfert, il le relance à partir de l'état sauvegardé dans la Base. L'application du client peut interroger directement le service GridFTP, pour mettre fin au processus de transfert, ou pour configurer la taille du cache de TCP.

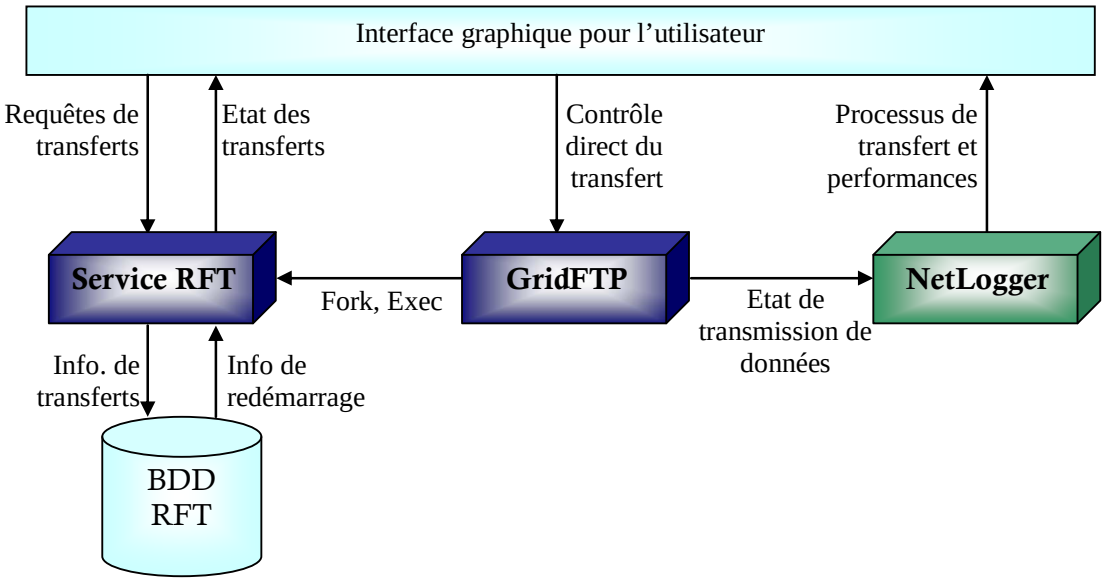


Figure 2-5: Architecture du service RFT [30].

Niveau réseau de RFT: le niveau réseau reçoit les informations à partir de TCP qui est responsable de la transmission des paquets de données, le TCP envoi les paquets et attend un accusé de réception de l'autre côté de connexion (la destination), quand il ne reçoit pas l'accusé, il retransmet le paquet jusqu'à son envoi. Dans le cas d'une panne dans le réseau, le TCP ne peut pas sauvegarder l'état de transfert des paquets, et pour cela, RFT relance le transfert à partir du dernier état sauvegardé. La définition de la taille du cache de TCP peut être crucial pour augmenter les performances du transfert. Ainsi le RFT utilise la fonctionnalité de négociation automatique ou manuelle offerte par le GridFTP afin de définir la taille du cache de TCP utilisé pour le transfert.

Niveau système de RFT : Le système contrôle les transferts de données, il peut détecter la source de l'exception qui a interrompu le transfert, ces exceptions peuvent être produites au niveau de la source, la destination ou au niveau du troisième élément de transfert. Le système peut redémarrer le processus de transfert à partir de son dernier état sauvegardé lors du rétablissement de la connexion entre les hôtes.

C. Replica Location Service (RLS)

La réplication de données joue un rôle important dans la gestion des données, elle permet d'augmenter les performances globales des grilles. On réplique pour diminuer la charge des nœuds contenants les données. Ainsi, elle sécurise les données en les répliquant, et donc créer des copies de secoures pour les données qu'on peut perdre facilement dans un environnement dynamique tel que les grilles. La réplication des données sur plusieurs

nœuds de la grille crée des serveurs de données qui peuvent être utilisés pour diminuer le temps d'accès vers un serveur distant, et donc élever les performances des transferts.

RLS est un composant du module de gestion de données dans Globus, il fournit un mécanisme pour répliquer et retrouver les réplications des données dans la grille. Chacune des donnée possède un nom logique et un ou plusieurs noms physiques [31], l'association entres le nom logique et les noms physiques se fait par le RLS, il utilise un catalogue pour sauvegarder et fournir des informations concernant les réplications de données (nom logique, noms physiques, nombre de copies, emplacement, etc.).

RLS est composé de deux catalogues, un libellé *Local Replica Catalog* (LRC), et *Replica Location Index* (RLI), le premier maintient l'association entre le nom logique de la donnée répliquée et les noms physiques. Ainsi, il supporte environ 600 mise à jour et 2000 requêtes d'accès par seconde [23]. Une nouvelle entrée dans le LRC est crée à chaque publication d'une nouvelle copie d'une donnée. Le deuxième registre RLI utilisé par RLS est utile dans un déploiement distribué, i.e. l'existence de plusieurs LRC dans la grille. Il collecte toutes les informations des noms logiques existants, ainsi, lors d'un lancement d'une requête cherchant un nom logique d'une donnée, le LRI retourne comme résultat la liste de tous les LRC possédant ce nom logique. La figure 2-6 montre un déploiement possible de RLS.

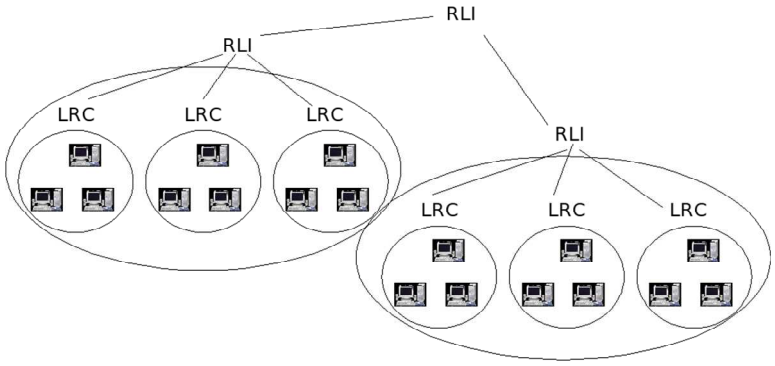


Figure 2-6: exemple d'un déploiement de RLS.

D. Data Replication Service (DRS)

C'est un WS-Service de haut niveau, il est responsable de prendre en charge les requêtes de réplications, il enregistre les nouvelles copies dans le RLS. Ainsi, il vérifie l'existence physiques des données dans la grille. Il interroge le service d'information pour choisir le meilleur chemin de réplication, il bénéficie des prédictions fournit par le SI sur les performances du réseau et des médias de stockage pour effectuer la réplication [32].

La requête de la réplication lancée par le client représente la ressource à état de DRS, le DRS fournit les informations de la WS-Ressource qui inclut [33] :

- Etat de la réplication (Status) : indique si la requête de réplication est en attente, active, suspendue, achevée ou détruite.
- Type d'action (Stage) : indique si l'opération est un transfert, recherche ou registration.
- Résultat (Result) : indique le résultat de l'opération, achevé, aucun, échec ou exception.
- Statistiques (Count) : fournit le nombre total des fichiers transférés avec succès et le nombre des échecs.

Le DRS fonctionne sur la base de deux Non-WS Services (RLS et GridFTP), et sur la base des deux autres WS-Services qui sont le RFT et le service de délégation fournit par le module de sécurité, utilisé comme une base de fonctionnement de plusieurs services dans Globus.

Le Client énumère les transferts qu'il veut effectuer, il identifie les données par leurs noms logiques et spécifie les destinations et les sources de transfert dans le cas de l'existence de third party dans le transfert. Une fois les spécifications ont été définies, le client envoi le fichier décrivant les transferts au DRS qui vérifie le certificat du client, et interroge les RLIs pour localiser les LRCs contenant les noms logiques des fichiers demandés, à présent, les LRCs sont localisés, le DRS récupère les noms physiques des fichiers à transférer et procède proprement à la phase de transfert, il utilise une instance du RFT et démarre le transfert. Le DRS récupère les états des transferts fournis par le RFT afin d'associer un état à chacun d'eux. Dans la phase suivante du fonctionnement, le DRS met à jour les informations des réplications existantes dans la grille, cette mise à jour est faite au niveau du registre LRC qui à son tour met à jour les LRIs correspondants.

2.3.2 Le module de sécurité (Globus security Infrastructure GSI)

Basé sur la cryptographie à clé public, il sert à l'authentification de l'utilisateur, une fois ce dernier authentifié auprès d'un service Globus, il sera authentifié pour tous les autres domaines sur lesquels Globus est installé et pour lesquels l'utilisateur est autorisé à accéder [34]. GSI utilise des standards de communications et garantit la sécurité dans les environnements à base de grille, il assure une communication sécurisée et certifiée, la confidentialité des informations, la non répudiation et l'intégrité des données transférées.

GSI inclut:

- Certificat: utilise le standard X.509v3 comme clé privée, il représente l'identité d'un utilisateur, d'une ressource ou d'un programme, ce certificat contient des informations nécessaires pour l'identification des entités, il est transmis d'une façon sécurisée via un tiers de confiance appelé autorité de certification (certification authority AC), ce dernier est utilisé comme une clé publique pour signer le certificat.
- Un algorithme d'authentification qui est défini par le protocole Secure Socket Layer (SSL) Renommé Transport Layer Security (TLS) par l'IETF, cet algorithme sert à l'identification des entités. Le TLS est un protocole de sécurisation des échanges effectués via Internet, il repose sur le procédé de cryptographie par clé publique et crée ce qui est connu par 'canal de communication sécurisé'.
- Mandataire et délégation (proxy) : se traduit par le besoin d'une seule authentification pour accéder à plusieurs applications, le SSO qui fait référence au mécanisme de délégation permet de répondre à ce besoin. L'utilisateur crée le Proxy qui va agir en son nom.
- Chaque ressource peut spécifier ses règles afin de pouvoir y accéder.
- GSI convertit les informations des identificateurs en un nom de sujet local (local subject name). Pour se faire, GSI utilise une liste de contrôles d'accès 'mapfile' qui définit les rapports entre le nom global et le nom en local.

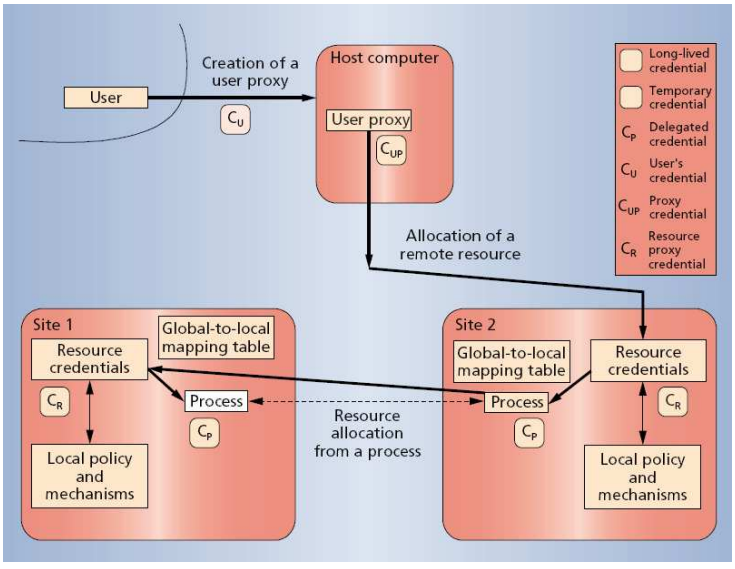


Figure 2-7: basique opération de GSI [35].

Lorsqu'un utilisateur veut accéder à une ressource, application ou un service qui peuvent êtres hébergés dans un site distant comme montré dans la figure 2-7, il crée un proxy via un mécanisme d'authentification à clé publique appliqué sur la licence de l'utilisateur (user's credential CU) suivi par la création de licence proxy temporaire d'utilisateur (temporary user

proxy credential CUP), et des demandes envoyées vers la ressource distante représentée par sa licence (resource proxies) qui lance le proxy de la ressource (resource proxy credentials CR) pour vérifier les droits et les conditions d'utilisation de la ressource.

L'accès à une ressource distante implique l'utilisation des fichiers mapfile incluant les licences déléguées (CP), et l'utilisation du protocole SAML qui sert à l'accès aux services d'autorisation. La figure 2-8 illustre une vue en couche du module GSI.

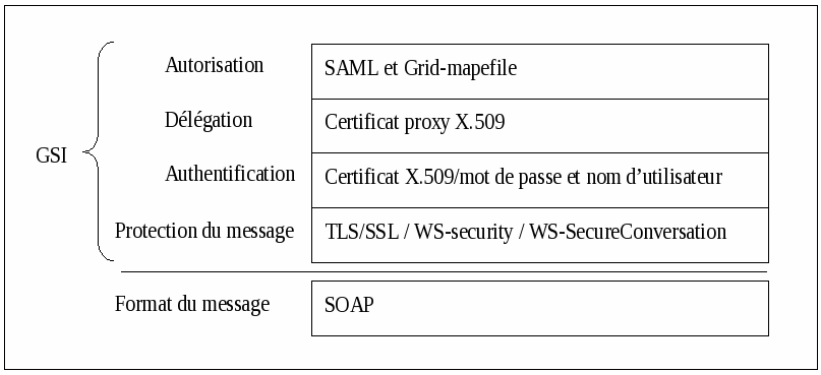


Figure 2-8: Vue en couche de l'infrastructure GSI.

2.3.3 Module de gestion d'exécution

Un module responsable du lancement, de la synchronisation et la supervision distante des jobs. Le service responsable de la gestion des exécutions est nommé Web Services Grid Resource Allocation and Management (GRAM). Dans la version 4.2 du Toolkit, GRAM a connu des améliorations [36] par rapport aux autres implémentés dans des versions précédentes de Globus. GRAM4 définit des mécanismes pour l'exécution et la supervision des jobs, ainsi que de récupérer les résultats de ces exécutions. GRAM4 est une extension de GRAM2 utilisé dans la version 4 de Globus et qui a été introduit à partir GT2.

L'objectif de GRAM est de lancer des jobs à distance, en parallèle, avec consistance, en sécurité et en gérant les états des exécutions. GRAM4 inclut des services gestionnaires de jobs, des adaptateurs locaux de système et d'autres services de GT4.2 [23].

GRAM4 fournit [23,36]:

a. Collection de services GRAM pour la gestion des jobs :

- Un service gestionnaire de job pour contrôler le cycle de vie d'un job.
- Un service de transfert de fichiers qui support l'implication des fichiers dans les ressources utilisées durant les calculs. Pour cela il s'interface avec le RFT pour un transfert performant des fichiers avant et après l'exécution du job.

- Des services de gestion de certificat utilisés pour contrôler la délégation des droits issus des applications des utilisateurs.

b. Adaptateurs d'Ordonnanceur : GRAM fournit une architecture plug-in pour permettre l'extension avec les adaptateurs afin de s'interfacer avec une multitude d'Ordonnanceur.

Pour la gestion de la sécurité des exécution, Contrairement à GRAM2 qui utilise le service GSI et des sockets, GRAM4 assure la sécurité des opérations en utilisant des fonctionnalités WSRF afin d'authentifier des requêtes utilisées pour la gestion des jobs et les requêtes des jobs elles-mêmes; il met en place un système local de sécurisation des domaines qui permet d'exécuter les jobs dans un contexte de sécurité local, les mécanismes de sécurité utilisés minimisent les privilèges donnés aux jobs et donc minimise les risques des programmes malicieux ; GRAM4 utilise la délégation des droits du client pour accéder aux ressources désirées, le client doit aussi déléguer les droits aux jobs eux-mêmes, cette délégation de droits est utilisée autrement dans GRAM2, ce dernier donne la délégation au service GRAM et aux jobs au même temps, ce qui ne permet pas le partage des droits en cas de besoins. Le certificat peut prendre trois valeurs qui traduisent son opportunité [37].

- Aucune (None) : aucune délégation de certificat ;
- Partagé (Shared) : une seule délégation de certificat est utilisée par tous les jobs ;
- Par job (Per Job) : délégation utilisée pour un seul job.

GRAM4 effectue des exécutions parallèles en permettant la notion des rendez-vous entre les processus.

GRAM4 peut gérer efficacement l'Ordonnancement des processus, mais pour des besoins d'utilisation à grande échelle, il utilise des composants externes comme les Ordonnanceurs Fork, Condor, PBS, LSF, Loadleveler, etc. De même, il utilise SUDO pour avoir plus de privilèges en cas de besoins, la version GRAM2 utilise tous les privilèges (root) ce qui donne une permission excessive aux jobs et augmente le risque de violation de sécurité.

Les composants internes utilisés par GRAM4 sont : (1) un générateur d'événement d'Ordonnanceur (Scheduler Event Generator), il donne la possibilité de contrôler les exécutions des processus, et il peut s'interfacer avec les autres Ordonnanceurs connus. (2) Fork Starter qui lance et contrôle l'exécution des jobs.

GRAM4 utilise le langage de description des jobs (Job Description Document (JDD)) qui est une extension du Resource Specification language (RSL), il est basé sur le langage XML.

Avec JDD, on peut spécifier le fichier exécutable et les ressources vérifiant les conditions exprimées en RSL, tel qu'une taille de mémoire vive précise, ou un type de ressource matérielle ou un espace disque de sauvegarde etc.

Pour la communication entre les services dans la grille, et pour standardiser l'invocation des services de middleware, Globus utilise MPICH-G2 dont l'architecture repose sur l'implémentation de MPI (Message Passing Interface), l'architecture en couche de MPICH est illustrée par la figure 2-9 :

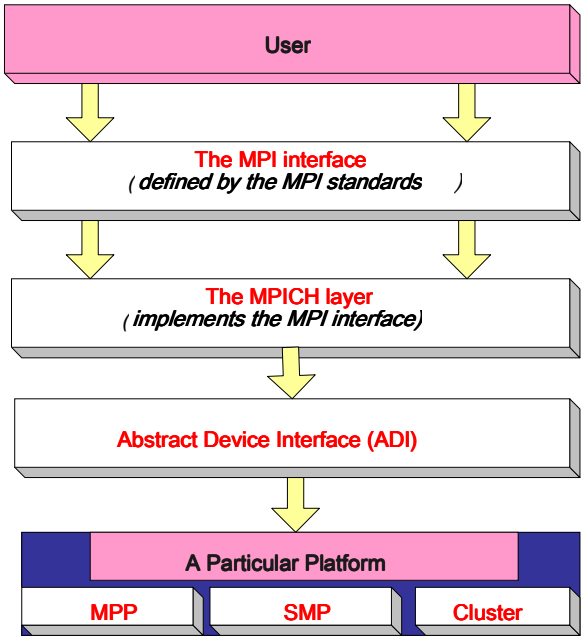


Figure 2-9: Architecture en couche de MPICH [84]

MPI est un standard permettant la communication entres services via l'échange de message, MPI fait recours aux méthodes et aux spécifications de l'Abstract Device Interface (ADI) [85].

2.3.4 Le module d'information (Monitoring and Discovery System MDS4 ou WS MDS)

Les mécanismes de contrôle et de découverte d'informations sont concernés par l'obtention, la distribution, l'indexation, l'archivage et par d'autres opérations sur des informations liées aux configurations et aux états des ressources distribuées dans la grille [38]. Ce collecte d'informations sert au contrôle et à la supervision de l'état globale de la grille, toutes les informations collectées sont accessibles via le (Monitoring and Discovery System) MDS.

Le MDS regroupe un ensemble de WS-Services pour contrôler et découvrir les services et les ressources de la grille, MDS4 inclut MDS2 de la version 2 du Toolkit, il est composé d'un service d'indexation, un service trigger, un framework d'agrégation, des fournisseurs

d'informations et des clients WebMDS (Web Service Data Browser) visionneur d'informations.

A. Services de haut niveau (higher-level services)

- *WS MDS Index Service*: une extension du registre UDDI, c'est un service de groupe utilisé pour la collection dynamique et la publication des informations sur les ressources de la grille, les programmes des clients lancent des requêtes après inscription et authentification pour retrouver les informations via le service index. Ce dernier est implémenté en utilisant le Framework d'agrégation. Les informations peuvent êtres ajoutées aux services par différentes sources d'agrégation qui sont utilisées pour fournir les informations des ressources au service index. Les informations collectées par ce service peuvent êtres examinées en utilisant des requêtes, par exemple les requêtes XPath [23].
- *WS MDS Trigger Service* : il permet la collecte des informations des ressources, il peut être configuré pour lancer des actions tel que notifier l'administrateur par envoi de mail, ou par la création d'un fichier log si l'espace disque est inférieure à un certain seuil (ressources de stockage d'un nœud distant).
- *Aggregator Framework* : il implique les deux services précédemment cités (index, trigger). Il est utilisé pour construire des services de collection et d'agrégation de données. Il fournit des mécanismes et des interfaces communs pour assurer les interactions avec les sources de données [39]. Ce framework peut être interrogé par plusieurs méthodes, par des requêtes lancées ou par des notifications requises par des WS-Services, ou par des programmes pour exécuter des actions lancées après vérification de certaines conditions. La figure 2-10 illustre le framework d'agrégation.

B. Fournisseurs d'informations

Utilisés pour la collection des informations des ressources spécifiques tel que des informations sur le host (nom, ID), processeur, taille mémoire, nom et version d'OS ou du système de fichier. Les informations qui peuvent êtres fournies par ces fournisseurs sont ceux concernant le contrôle des clusters (Ganglia cluster monitor, Hawkeye cluster monitor…), la gestion des ressources (GT4 WS GRAM) qui fournit les états des tâches, GT4 Reliable File Transfer (RFT) service qui fournit des informations sur les actions de transferts des fichiers, GT4 Replica Location Service (RLS) pour fournir des informations sur les catalogues impliqués dans la réplications des données… et les Ordonnanceurs (PBS scheduler, LSF scheduler…).

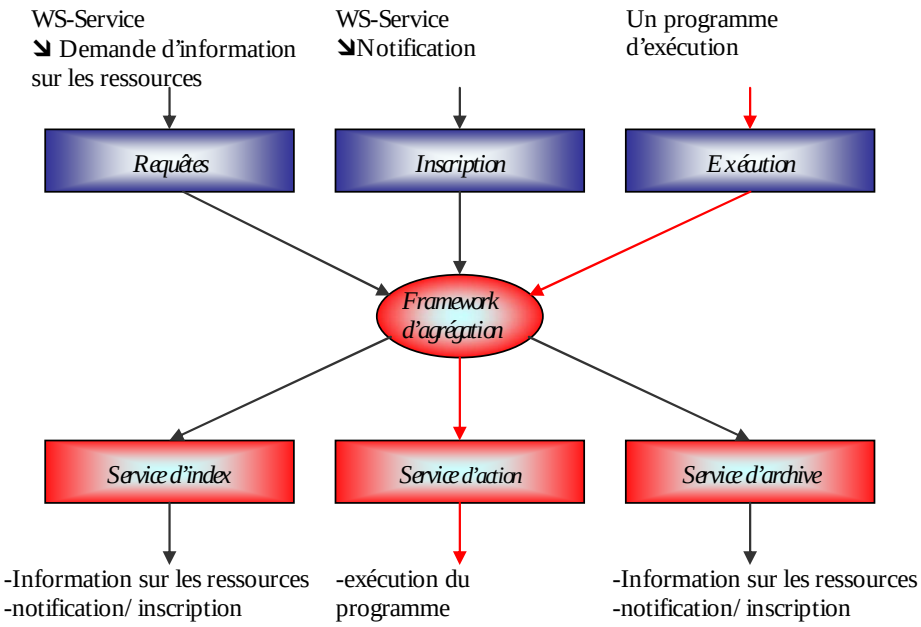


Figure 2-10: framework d'agrégation [23].

C. Client WebMDS

C'est une interface (front-end) du service d'indexation, basé-Web, utilisée pour visualiser les propriétés des WS-Ressources en lançant des requêtes (standard resource property requests) pour interroger les informations des propriétés des ressources, et affiche les résultats du format XML ou XSLT sous formes compréhensibles par l'utilisateur.

2. 4 Conclusion

Comme tous les Intergitiels, Globus est une technologie fondamentale pour les grilles, il permet de partager en toute sécurité la puissance de calcul, les bases de données distribuées et d'autres outils.

Le Toolkit Globus fournit des services et des librairies de programmation qui implémentent des standards pour la gestion de sécurité, gestion de données, découverte et surveillance de ressources et pour la gestion des tâches.

En plus des outils fournis, Globus offre une couche Application qui permet aux programmeurs de créer leurs propres applications et leurs propres services en utilisant différents langages de programmation (C, Java et Python).

Introduction en traitement des images.

3. Introduction en traitement d'images

3.1 La matrice de Cooccurrence

C'est une méthode statistique d'ordre supérieur proposée par Haralick [40], elle permet une analyse plus précise, elle est largement utilisée dans l'analyse de la texture des images, et donne des résultats satisfaisants pour différents types d'images [41]. Elle est plus simple à mettre en œuvre, elle permet de déterminer la fréquence d'apparition d'un motif formé par deux pixels, elle utilise deux paramètres, 'd' la distance entre les pixels et Θ l'angle de la droite reliant ces 2 pixels par rapport à l'horizontale. Une image avec un niveau de gris N est définie par une matrice de cooccurrence φ(d, θ) de taille N.

La matrice de cooccurrence extraite de l'image n'est pas utilisée directement, elle est exploitée à partir des informations qu'elle peut contenir. Haralick a définit 14 caractéristiques statistiques qui permettent une meilleure discrimination entre les différents types de textures, et d'estimer la similarité entre les matrices de cooccurrence. Les caractéristiques de Haralick les plus utilisées sont : l'énergie, l'entropie, le contraste et le moment inverse de différence.

L'énergie (T1) : $\sum_i \sum_j p_d^2(i, j)$ (1)

L'entropie (T2): $-\sum_i \sum_j p_d(i, j) \log p_d(i, j)$ (2)

Le contraste (T3): $\sum_i \sum_j (i-j)^2 p_d(i, j)$ (3)

Le moment inverse de différence (T4): $\sum_i \sum_j p_d(i, j) \Big/ (i-j)^2$, i≠j (4)

Le code Java de chacune des caractéristiques est écrit dans la partie ANNEXE B.

La distance D entre deux images Img1 et Img2 est définie par le calcul de distance entre les caractéristiques de Haralick :

$$D_{Img1,\ Img2} = \sqrt{(T1_{Img1} - T1_{Img2})^2 + (T2_{Img1} - T2_{Img2})^2 + (T3_{Img1} - T3_{Img2})^2 + (T4_{Img1} - T4_{Img2})^2}$$

Où

$T1_{Img1}$: L'énergie de l'image Img1.

$T1_{Img2}$: L'énergie de l'image Img2.

$T2_{Img1}$: L'entropie de l'image Img1.

$T2_{Img2}$: L'entropie de l'image Img2.

$T3_{Img1}$: Le contraste de l'image Img1.

$T3_{Img2}$: Le contraste de l'image Img2.

$T4_{Img1}$: Le moment inverse de différence de l'image Img1.

$T4_{Img2}$: Le moment inverse de différence de l'image Img2.

Une seconde approche qui se fond sur l'utilisation des matrices de cooccurrence, appelée (Color Co-occurrence Matrix [44] (CCM)), cette méthode est utilisée pour capturer la variation des couleurs dans une image. CCM est représentée comme une matrice tridimensionnelle, où la paire coloré des pixels p et Np est située dans la première et la seconde dimension de la matrice, et la distance spatiale 'd' entre ces deux pixels est située dans la troisième dimension. La méthode CCM peut être utilisée avec l'espace de couleur HSV [45] ou avec l'espace RGB. Dans les deux cas, on obtient six matrices de cooccurrence, multipliées par le nombre de paramètres de Haralick choisi, on aura comme résultat, une trentaine de paramètres qui aident à la reconnaissance et la classification des images.

3.2 Filtrage

Le filtrage est une opération de voisinage appliquée à un pixel, où sa nouvelle valeur est remplacée par le résultat d'une fonction appliquée sur lui ansi que sur ses voisins. L'objectif de cette étape est l'amélioration de l'image numérique pour augmenter la qualité de son rendu visuel et à sa simplification pour faciliter les opérations d'analyses ultérieures, ils existent plusieurs types de filtres :

3.2.1 Filtrage dans le domaine spatial

Les méthodes de filtrage dans le domaine spatial portent sur les pixels de l'image : $g(x,y) = T[f(x,y)]$ où $f(x,y)$ est l'image d'entrée, $g(x,y)$ est l'image de sortie et T représente l'opérateur sur f.

Le processus de filtrage dans le domaine spatial consiste à avancer un masque de filtre d'un pixel de l'image à un autre, les filtres dans ce domaine sont regroupés en deux catégories :

Les filtres lisseurs : comme le filtre Gaussien et le filtre moyenneur dans lequel l'image de sortie est représentée par la moyenne des pixels de voisinage du masque appliqué.

Les filtres d'accentuation des bords : telque le filtre de Sobel et Laplace, leur objectif est de mettre en évidence les détails dans une image, ou d'augmenter le détail qui a été troublé par erreur ou comme un effet naturel d'une méthode particulière d'acquisition d'image [42].

3.2.2 Filtrage dans le domaine fréquentiel

Utilisé sur les images au niveau de gris et les images binaires pour enlever le bruit [43]. Le principe du filtrage en fréquences d'une image est de prendre la TF (Transformée de Fourier) de l'image à filtrer, multiplier le spectre obtenu par la fonction de transfert du filtre, puis prendre la TF inverse pour produire l'image filtrée.

Le lissage par réduction du contenu hautes fréquences, ou le rehaussement de contours par augmentation des composantes hautes fréquences vis à vis des basses fréquences, proviennent de concept directement reliés à la transformée de Fourier.

En effet, l'idée de filtrage linéaire est beaucoup plus intuitive dans le domaine fréquentiel. En pratique, les masques spatiaux sont utilisés beaucoup plus que la TF du fait de leur simplicité d'implémentation et de leur rapidité. Mais la compréhension des phénomènes dans le domaine fréquentiel est indispensable pour résoudre des problèmes difficilement appréhendables avec des techniques spatiales.

3.3 Segmentation

La segmentation des images est l'une des étapes les plus importante en traitement d'images, elle consiste à partitionner l'image en un ensemble de régions connexes. L'intérêt de la segmentation est de pouvoir manipuler les régions via des traitements de haut niveau pour extraire des caractéristiques de formes, de position, de taille, etc.

La segmentation d'images est un traitement de bas niveau, il consiste à créer des partitions homogènes dans l'image afin d'y extraire les informations utiles pour son interprétation. Il existe une multitude de techniques de segmentation, le choix d'une technique dépend des opérations situées en aval, des primitives à extraire ou des contraintes d'exploitation.

Le problème est évidemment très mal posé, car on ne sait jamais dire quelle est la segmentation idéale. On peut dégager des propriétés les plus raisonnables qu'on cherche à obtenir dans un algorithme de segmentation, en particulier:

- Stabilité : la segmentation obtenue ne doit pas varier beaucoup lorsque les conditions d'acquisition varient légèrement (bruit, illumination, point de vue,...)

- Régularité : les régions obtenues doivent être simples à manipuler (taille suffisante, forme régulière,...)

3.3.1 Méthodes statistiques

A. Histogrammes et Segmentation

Les méthodes par histogrammes sont généralement rapides à calculer et peu sensibles au bruit, mais elles n'intègrent pas (ou peu) d'information géométrique sur les régions. Ce sont des méthodes globales, au sens où la décision d'appartenance d'un pixel à une région dépend toujours de l'image entière.

B. Segmentation par sélection récursive sur histogrammes

L'histogramme d'une image représente la répartition des valeurs des composantes couleurs des pixels formant l'image. Ainsi, des régions ayant des caractéristiques communes dans l'image couleur font apparaître des modes délimités par des seuils dans les histogrammes couleurs. La segmentation par l'analyse de ces histogrammes consiste à détecter les modes représentatifs de ces régions, et à déterminer les seuils à appliquer.

3.3.2 Méthodes géométriques

Ces algorithmes intègrent naturellement les propriétés topologiques, et aussi parfois géométriques des régions. Ils ont comme inconvénients de mal déterminer les frontières entre régions.

A. Croissance de régions (Region growing)

Consiste à regrouper les pixels vérifiant un critère d'homogénéité, ce critère peut être de différentes natures, le plus simple étant la comparaison des niveaux de gris des pixels selon un seuil. La méthode est développée initialement par Muerle et Allen en 1968. Cette approche consiste à choisir, d'abord, des germes de régions, puis à faire croître chaque germe en intégrant progressivement des pixels voisins à ces germes. Dans cette méthode, deux facteurs agissent sur le résultat de la segmentation : le premier est le choix des germes initiaux, le second est la condition d'arrêt de la croissance de régions.

B. Décomposition/fusion (Split & Merge)

L'idée des algorithmes de ce type est de produire automatiquement une partition initiale de l'image, représentée par des petites régions (Split), qui vont ensuite croître en se regroupant (Merge). La partition initiale (Split) est réalisée en appliquant un algorithme récursif pour le

partitionnement de l'image en régions, la fusion se fait entre les régions adjacentes dont l'union respecte des critères d'homogénéité.

Ce type de méthode fait souvent appel à la théorie des graphes, on y retrouve les exemples : partitionnement de Voronoï, arbre quaternaire, approches pyramidales.

3.3.3 Méthodes par optimisation

Dans les méthodes par optimisation, le problème de la segmentation est formalisé par l'estimation d'une fonction *f* bidimensionnelle, en minimisant une fonctionnelle de coût K qui va dépendre de l'image analysée. La fonction *f* représente l'image segmentée par ses restrictions f_i sur chaque région R_i.

3.4 Classification

La classification consiste à extraire les différentes caractéristiques de l'image, ces derniers sont liées à la géométrie de l'image et à ses colorimétries (nombre de couleurs, pourcentage de couleurs saturées, variations brutales de couleur, histogramme, …). Les caractéristiques extraites sont fournies à un classifieur pour arranger les images dans une ou plusieurs classes qui leurs sont appropriées. Ils existent trois modes de classification : supervisé, non supervisé, et hybride.

A. Méthodes de classification automatique

Appelée aussi classification non supervisée, segmentation ou également clustérisation, elle consiste à rechercher des groupes homogènes inconnues au départ dans une population d'individus représentée par une ou plusieurs variables. Le DataMining propose plusieurs méthodes de classification automatique telle que la classification ascendante hiérarchique, la classification descendante hiérarchique, la méthode des centres mobiles...etc.

B. Méthodes de classification supervisée

La classification supervisée cherche à déterminer l'appartenance d'un événement à des classes préalablement identifiées par segmentation [47].

C. Classification hybride

Utilise les deux techniques précédentes (classification supervisée et automatique) pour rendre le processus plus efficaces et plus précises.

3.5 Conclusion

L’imagerie permet de modifier et de manipuler le contenu des images afin de tirer l'information utile pour une application particulière. Beaucoup de recherches sont portées sur l’analyse des images y compris le filtrage et l’indexation, et d’autres sur l’extraction de caractéristiques.

Jusqu’à maintenant, avec l’évolution constante de l’information, et avec la richesse des images en nombre et en qualité, il n’y a pas de méthode générale pour extraire la connaissance d’une image. Ils existent des algorithmes qui restent toujours liés à leur contexte, et qui fournissent de bons résultats dans leurs domaines de fonctionnement. Ainsi, ils existent certains outils qui nous offrent des moyens pour implémenter les algorithmes de traitement d’images tel que Matlab, .net, etc.

Systèmes CBIR à base de grilles.

4. Systèmes CBIR à base de grilles

4.1 Introduction

Les Systèmes de recherche des images à base de leurs contenus sont largement utilisés dans le domaine médical. Ainsi, ils ont été introduits dans plusieurs plateformes tel que les moteurs de recherche des images comme le cas de Google, les plateformes d'analyse des images satellitaires, et dans le domaine militaire. Généralement, les systèmes CBIR peuvent résoudrent les problèmes liés avec le temps d'achèvement d'une opération critique, ou avec ceux qui traitent une grande quantité d'images. Dans le domaine médical, les systèmes CBIR aident les médecins à prendre des décisions à base des cas précédents. Pour les moteurs de recherche des images, les systèmes CBIR rend l'opération de la recherche plus précise, elle ne s'appuiera plus sur la description textuelle, mais elle repose sur les images elles mêmes.

L'utilisation des grilles augmente d'une façon perceptible les performances des systèmes CBIR, elles fournissent la puissance de calcul requise et la mémoire vive ou de stockage nécessaire. L'analyse de millier d'images consomme beaucoup de temps, l'exécution parallèle de cette tâche est le seul chemin pour remédier ce problème, les grilles offrent la solution la moins coûteuse, on n'a pas besoin de ressources très puissantes, car la puissance des grilles réside dans la combinaison des ressources existantes.

4.2 Etat de l'art

Plusieurs travaux sont portés sur l'utilisation des grilles pour l'augmentation des performances des systèmes CBIR, leurs implications dans les moteurs de recherche des images ont été élaborés par le projet IMENSE [49]. Il utilise la puissance de la grille pour mettre en œuvre des techniques et des méthodes CBIR sur un nombre important des images par distribution de la tâche requise sur des milliers de nœuds. Avec seulement deux serveurs de stockage et de soumission de job, Imense permet l'analyse et l'indexation d'environ 25 millions d'images à hautes résolutions.

Le système CBIR développé par Imense Ltd est basé sur une analyse automatisée et de reconnaissance de contenu des images avec l'utilisation d'une ontologie sémantique, il est dotée aussi par des modules d'analyse pour effectuer la segmentation, la classification à base de régions, l'analyse de scène, la détection d'objet et implémente des méthodes pour la reconnaissance des visages.

Le système CBIR d'Imense est déployé par l'*international particle physics community* dans une puissante grille (plus de 120,000 CPU) connue sous le nom 'GridPP', qui est une collaboration entre la compagnie Imense Ltd à Cambridge UK, et *Cambridge University eScience Centre*. Pour la gestion des processus et la soumission des Jobs, Imense utilise le système Ganga [48] montré dans la figure 4-1, et qui est personnalisable et supporte une multitude de points d'arrivés.

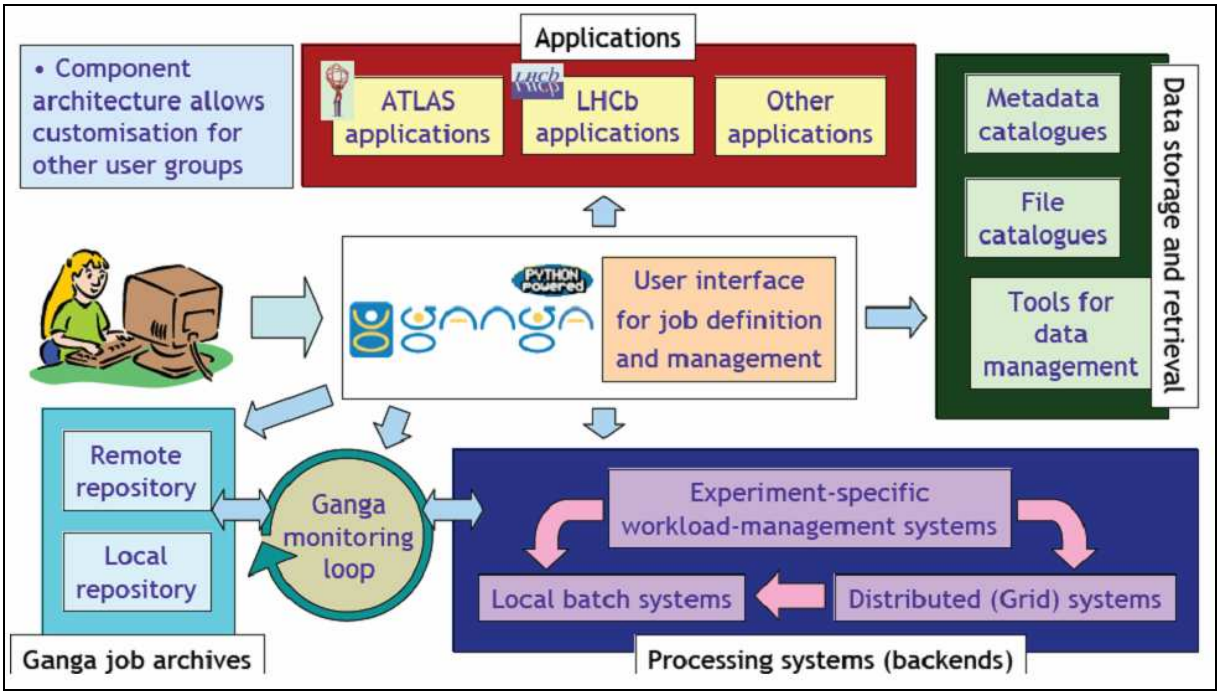


Figure 4-1: Composants utilisés avec le système Ganga pour la définition, la soumission et le contrôle des Jobs et le contrôle de *particle physics Grid* [48].

Dans des résultats présentés dans [49], plus de 500 Jobs s'exécutent en parallèle traduits par 500,000 images traitées en même temps avec la moyenne de 70 GB ou 290,000 images durant deux heures.

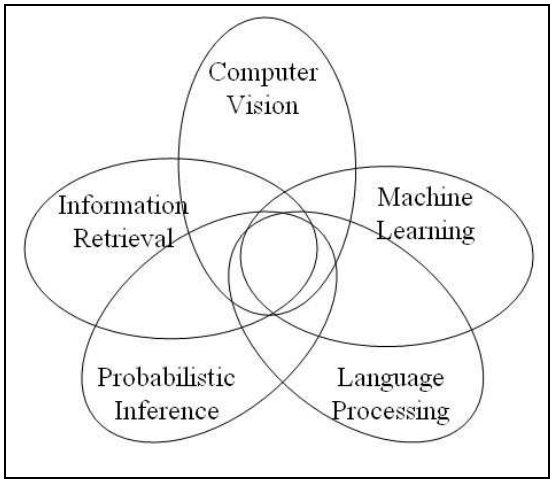


Figure 4-2: domaines implémentés par Imense.

Imense Ltd combine plusieurs domaines de traitement d'images comme montré dans la figure 4-2, il implémente un système original de recherche d'images basé sur l'analyse automatisée et la reconnaissance du contenu des images, ainsi qu'un langage de requêtes basé sur les ontologies. Cette méthode d'analyse inclut la reconnaissance des propriétés visuelles tel que la couleur, la texture et les formes ; la reconnaissance de matériels tels que le gazon, le ciel ; et la classification des scènes à base de leurs contenus, par exemple la plage, la forêt, le couché du soleil.

Le système utilise les relations linguistiques et sémantiques entre les objets pour interpréter les requêtes des utilisateurs, et aussi pour chercher les images selon le résultat de l'analyse sémantique de la requête précédente. Comme le système est extensible, l'intégration de nouveaux modules pour la classification, l'analyse des images ou des *metadata* est facilement réalisable.

Comme montré par la figure 4-3, Imense Ltd possède plusieurs modules dans le but d'analyser une image selon son contenu. Dans le but d'identifier les parties importantes de l'images et qui correspondent aux objets ou aux parties de l'objet recherché, l'image est automatiquement segmentée en un ensemble non chevauché de régions, et d'un ensemble de propriétés calculé pour chacune de régions tels que la taille, la couleur, les formes et la texture. Le nombre de régions segmentées dépend de la taille de l'image et sa complexité visuelle.

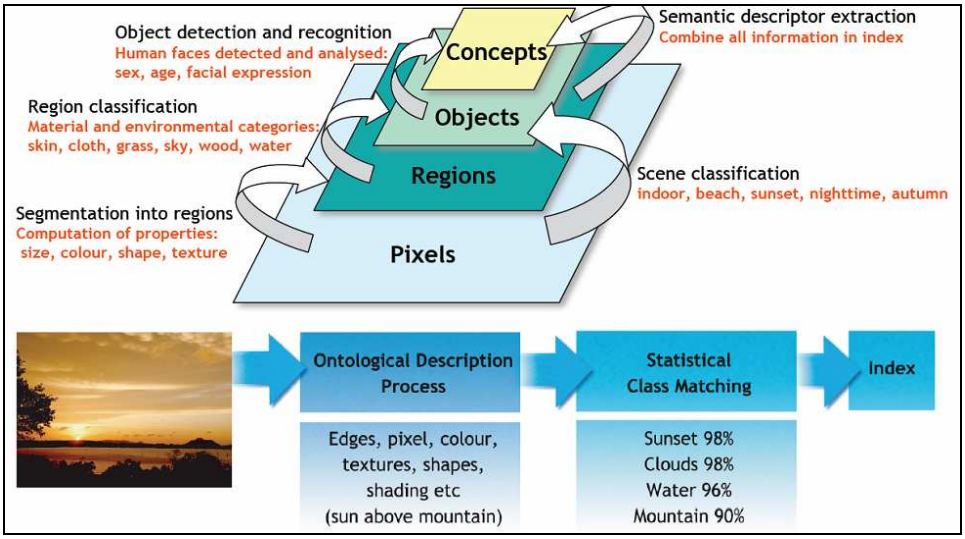


Figure 4-3: Vue en couche de l'analyse des images et le processus de reconnaissance par Imense Ltd[49].

Les régions segmentées sont automatiquement classées dans un ensemble de catégories matérielles et d'environnement tel que le gazon, ciel, eau, etc. Des méthodes statistiques sophistiquées de *machine learning* sont utilisées pour fournir une classification probabiliste optimale. Une seconde étape pour la classification est de catégoriser les régions selon leurs contenus en identifiant la nuit, le couché de soleil, l'automne, etc.

La Détection et la reconnaissance des objets est présentés par l'analyse de l'image et la détection des objets communs tel que la face humaine, qui est automatiquement détectée et classée selon quelques attributs comme l'age, le sexe et les expression de visage comme montré par la figure 4-4.

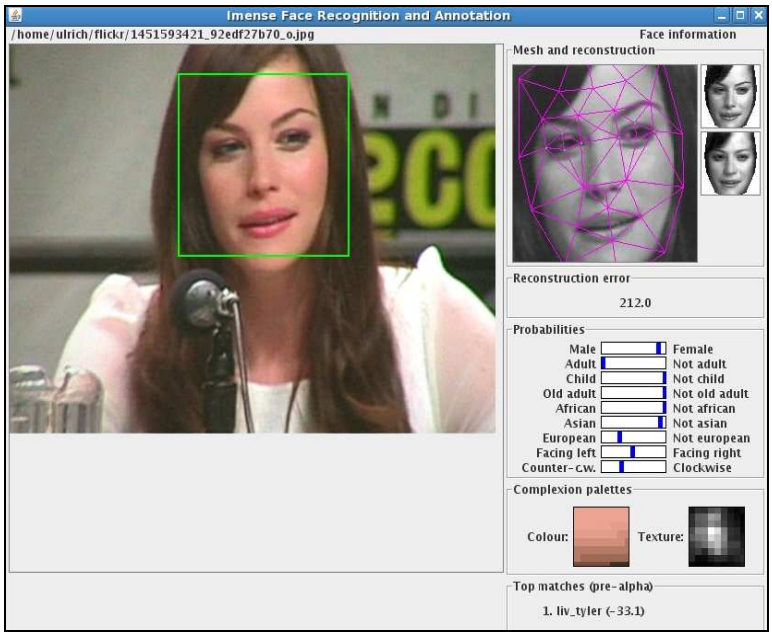


Figure 4-4: analyse du visage humain.

L'Indexation est faite après l'étape d'analyse des images, les informations collectées des différents classifieurs sont combinées dans un format d'indexation spécial, et compilées au niveau des serveurs distribués sur plusieurs sites de la grille, cela a pour raison de supporter l'indexation d'un grand nombre d'images qui atteint plusieurs millions.

Un autre projet qui a porté sur les applications médicales complexes [50], elles peuvent bénéficier de l'utilisation de la technologie des workflow, de leurs performances, leurs designs et leurs implémentations rapides. De même, une combinaison avec la technologie des grilles de calcul est apparue très utile dans l'exploitation des ressources. Dans le travail à décrir, les auteurs ont réalisé une connexion entre le système de gestion de workflow Taverna [51] et l'infrastructure de grille EGEE (Enabling Grids for E-sciencE) à travers un pluging gLite, qui s'interface entre les workflow et l'infrastructure de la grille. Le pluging réalisé incrémente les performances des applications médicales.

Le Workbench Taverna est un gestionnaire de données dataflow open source, développé à l'UK par l'association myGrid [52]. Dès la première version de Taverna réalisée en 2004, elle a été largement adoptée par la communauté e-science, elle inclut une interface graphique riche pour la conception des workflow. Un workflow Teverna consiste en une collection de processeurs connectés par des liens de données, ils établissent une dépendance entre les sorties des processeurs et les entrées des autres. Les processeurs sont de différents types selon le code de l'application invoqué, généralement, les processeurs sont des Web Services avec des ports d'entré et de sortie qui correspondent aux opérations définies dans le fichier WSDL descriptif du Web Service. Dans le contexte du même travail, un nouveau type de processeur gLite été défini, le système orchestre l'exécution des processeurs et gère les données du workflow à travers les ports des processeurs.

Le Framework EGEE est Le projet qui réunit des experts de plus de 27 pays autour d'un objectif commun qui est de développer une infrastructure de service de grille accessible aux scientifiques 24 h/24. Actuellement, il supporte plus de 125 organisations virtuelles avec 9000 utilisateurs dans 50 pays, avec approximativement 20 petabytes mémoire de stockage et 80000 cores disponibles. L'infrastructure EGEE est constituée de différents composants, ils fonctionnent en coordination au niveau logique qu'au niveau physique. Le stockage de données d'EGEE est réalisé en forme d'Éléments de Stockage (SE) qui facilitent le stockage physique. Le Système de Gestion de Fichiers Logiques, avec un service de catalogage, maintient les données sur ces SEs. Les capacités de calcul sont fournies par les Eléments de calcul (CE), qui sont généralement représentés par un groupe de nœuds travailleurs (WN). Un

courtier de ressource (Resource Broker RB) Schedule les Jobs dans les filles d'attentes appropriées. Le composant d'enregistrement et de compatibilité (Logging and Bookkeeping LB) gère l'état des Jobs dans les files d'attente.

L'Intergitiel gLite est distribué sous une licence de type logiciel libre, «business friendly», gLite intègre des composants provenant des meilleurs projets middleware en cours, tels que Condor et le Toolkit Globus, ainsi que les composants développés pour le projet LCG. Le produit est d'une meilleure qualité, il utilise la solution middleware de bas niveau, compatible avec des gestionnaires de files d'attente tels que PBS, Condor et LSF. De même, le produit est inter-opérable et fournit les services de base qui facilitent la construction des applications basées grille dans plusieurs domaines. GLite supporte l'exécution du lot de tâches sous la forme d'un système de soumission de Jobs, le JDL simple (Job Description Language) est utilisé pour construire les spécifications des Jobs à soumettre. Le cycle de vie des Jobs est manipulé sans notification de l'utilisateur, ce dernier doit périodiquement interroger le système pour découvrir la progression des tâches. Le transfert de données entre localhost de l'utilisateur et les noeuds de réseau est exécuté avec des protocoles sécurisés comme sFTP et gridFTP. Les langages de programmations supportés sont C++ et des API de java pour la gestion des Jobs.

Le pluging T2 gLite est développé dans le but d'interfacer T2 workbench avec l'infrastructure de grille EGEE. Il permet à l'utilisateur de T2 de soumettre des Jobs à EGEE via le middleware gLite sous forme de tâches de workflow. L'état d'exécution des Jobs est consulté régulièrement et les résultats sont rassemblés en local à l'achèvement avec succès de chaque tâche. Les tâches sont exécutées d'une manière asynchrone sur l'infrastructure EGEE. La conception et la mise en œuvre du pluging sont motivées par les objectifs suivants : 1) Démultiplient la puissance du mode de données parallèle de T2 workflow, 2) l'Exécution simultanée de tâches sur l'infrastructure EGEE et 3) l'interface Robuste et insensible aux défaillances qui peut traiter les diverses erreur généralement rencontrées dans un environnement de grille complexe.

Le plugin gLite lance des appels asynchrones de T2 au système de gestion des Jobs de EGEE, et notifie le core de T2 dès l'achèvement de l'exécution du batch. La figure 4-5 montre l'interaction entre T2 et EGEE via le plugin gLite. T2 exécute le processeur gLite qui contacte à son tour le middleware. Le plugin prépare le Job et le soumettre à EGEE, il obtient un JobID, et consulte le statut de la tâche avec un thread indépendant jusqu'à son achèvement. Les tâches de T2 sont naturellement parallélisées, les tâches qui n'ont pas une dépendance de

données avec les Jobs EGEE peuvent être exécutées d'une manière asynchrone. Jusqu'à l'achèvement du Job, les résultats sont envoyés au gestionnaire de données de T2.

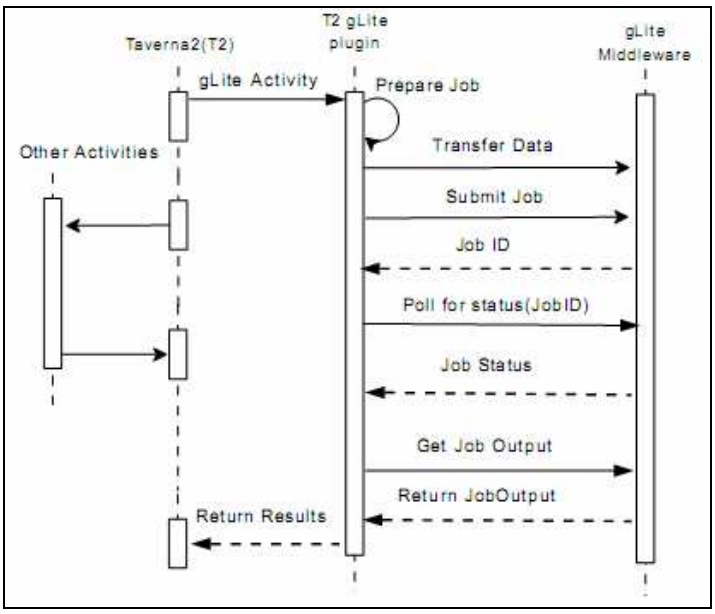


Figure 4-5: interaction entre le T2 et EGEE via le plugin T2 gLite durant une soumission de Job

Le concepteur de workflow configure les testes de T2 comme un Job de gLite. La description des Jobs inclut des informations sur le fichier exécutable et les arguments. Ces propriétés sont passées à l'API gLite. Les données sont transférées aux Éléments de Stockage de EGEE et identifiées avec un ID unique géré par le Système de Gestion de Données. Le protocole de transmission utilisé est le standard gridFTP.

Pour la soumission de Job, ce dernier est transmis vers un utilisateur de la VO, gLite renvoi un JobID unique retenu par le plugin comme une future référence, et aussi pour consulter l'état de l'exécution de Job au niveau de EGEE.

Le plugin gLite contient des mécanismes de traitement d'erreurs pour réduire la probabilité d'échec du workflow qui est lancé dans un environnement distribué. Il inclut (i) Une politique de resoumission de Job (après un certain temps d'attente, les Jobs sont resoumis [53]), (ii) Une politique Round-Robin pour le chargeur de Job de EGEE et (iii) Resoumission des demandes de transfert des données en cas d'erreurs pendant l'opération.

Les étapes suivantes récapitulent la procédure point par point de création, d'exécution et de réutilisation du workflow T2 avec des processeurs gLite :

- Créer un workflow en utilisant Taverna.
- Ajouter le processeur gLite au workflow.

- Configurer les propriétés du processeur gLite.
- Créer un certificat du mandataire de grille pour l'authentification.
- Déléguer le mandataire au middleware gLite.
- Ajouter des entrées/sorties et d'autres processeurs local/gLite au workflow.

Dans un autre travail [74], les auteurs ont fait la Griddification d'un système d'extraction d'images à base de contenu, appelé GNU Image Finding Tool (GIFT) [76]. L'objectif de cette étude été de montrer l'avantage des grilles de calcul pour les applications médicales. Le middleware utilisé dans ce projet est l'ARC [57] (Advance Resource Connector) par l'intermédiaire du projet de recherche KnowARC financé par l'Union européenne. La partie extraction de caractéristiques de GIFT été Griddifié. La performance de la grille est mesurée avec la vitesse du système Griddifié par la présentation de plusieurs scénarios. L'utilisation de la grille montre qu'en particulier, des tâches de calcul intensif, telles que l'extraction de caractéristiques visuelles de grandes bases d'images peuvent être réalisés beaucoup plus rapidement. Comme le middleware ARC doit être installé sur un système d'exploitation Linux, les auteurs ont utilisé pour leurs tests, des machines virtuelles VMware [75] (Virtual Machine Ware) installées sur des Système d'exploitation Windows.

Pour chercher les images similaires à une autre, présentée comme une requête, on a besoin d'indexer toute la collection, ce qui signifie que les caractéristiques visuelles extraites, décrivent chacune des images de la collection. La base des images médicale est fournie par ImageCLEF *medical image retrieval task* [13]. ImageCLEF est une partie de Cross Language Evaluation Forum (CLEF), qui est un forum pour la standardisation de la recherche des informations. La base de données utilisée contient 50.000 images en 2005, et plus que 70 .000 images en 2007. Les caractéristiques visuelles d'une image peuvent êtres calculées en parallèle, indépendamment des autres images, dans les scénarios de test effectués dans ce travail, une collection de 500 à 1000 images est à exécuter sur le même nœud.

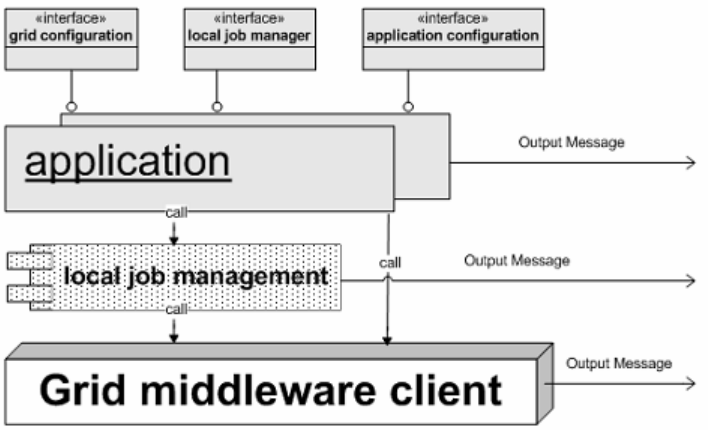


Figure 4-6: l'architecture de base de l'application gridifiée.

Dans la figure 4-6, les auteurs présentent l'architecture de base de l'application griddifiée. Un autre obstacle majeur est l'interface pour gérer les tâches en cours d'exécution sur la grille, L'utilisateur doit pouvoir superviser l'état des Jobs lancés sur la grille. ARC fournit un interface pour la gestion des Jobs appelé GridJM [78].

Un autre aspect important est que les informations présentées aux utilisateurs doivent être intuitives. La sortie (en particulier les codes d'erreur) à partir d'un Intergitiel de grille est souvent évident pour les experts mais difficile à comprendre pour les utilisateurs inexpérimentés, d'où l'importance de masquer tous les détails d'un middleware de grille et transmettre des messages appropriés basés sur les connaissances de l'utilisateur.

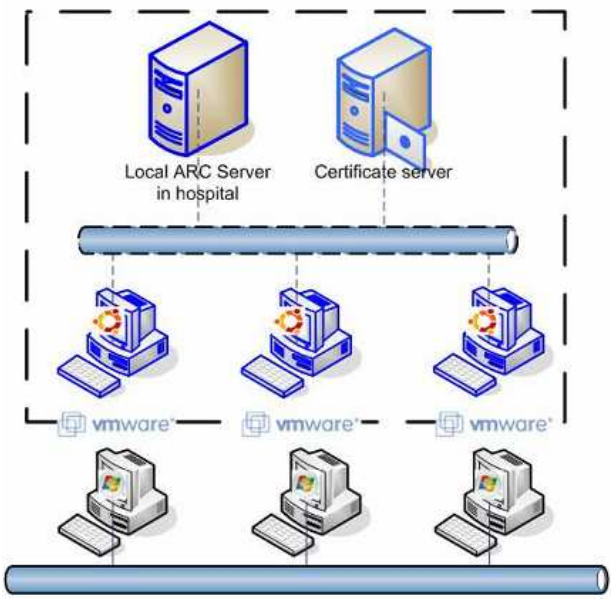


Figure 4-7: l'infrastructure utilisée pour le déploiement de l'application

Pour exploiter les ressources du bureau dans les hôpitaux. La figure 4-7 montre une configuration de réseau virtuel pour activer la connectivité réseau. Toutes les machines en bleu sont virtuelles, elles offrent des CPUs, mémoire et de l'espace disque. La version gratuite de VMware Server est utilisée dans les tests.

Dans un autre travail [79], les auteurs ont traité la manipulation des images médicales avec la technologie des grilles, y compris la production des images, le stockage sécurisé, et le traitement. Les résultats obtenus, montre que les grilles sont encore dans leur vigueur à résoudre tous les problèmes complexes liés aux applications d'imageries médicales.

Le gestionnaire des données DataGrid marque les fichiers par un identifiant unique de grille (GUID). Pour chaque GUID est associé un ou plusieurs fichiers nommés replicas. Le gestionnaire des données manipule des fichiers qui sont stockés dans différents Systèmes de Stockage de Mass (MSS) à travers une interface de stockage unifiée.

Afin d'assurer la tolérance aux pannes et assurer un accès efficace aux données, les fichiers sont enregistrés dans le gestionnaire de données et reproduites de façon transparente par le middleware. Quand un fichier est demandé, l'Intergitiel de grille choisira automatiquement la copie la plus facile à recueillir. Pour manipuler facilement les images de médecine de l'EDG, les auteurs ont conçu une interface de stockage aux serveurs médicale DICOM. Cela s'est avéré être difficile, car les données DICOM ne sont pas structurées comme des fichiers, mais que la collecte de tranches d'image (série DICOM) et les tranches de DICOM sont contenant à la fois les données d'images brutes et les métadonnées. Le gestionnaire des données médicales distribuées (DM2) qui a été développé, définit une abstraction d'images médicales, il sépare entre les données des images brutes et les métadonnées. Un DM2 connecté au gestionnaire des données DataGrid est représenté dans la figure 4-8.

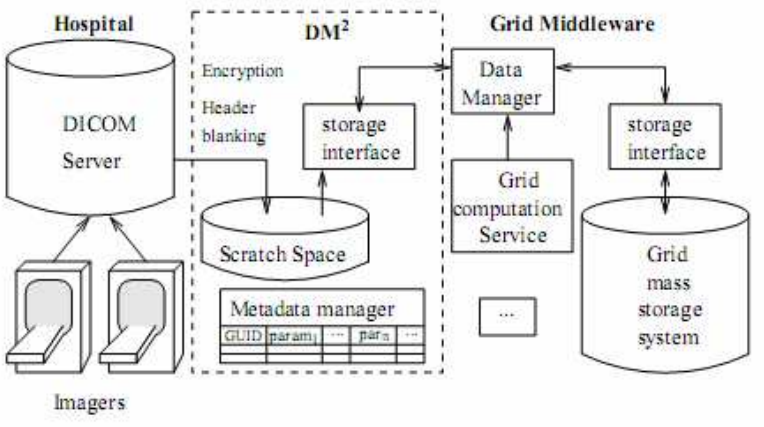


Figure 4-8: DM2 interface entre les données médicales et la grille.

L'application déployée sur la grille implémente des notions de base :

Pipelines : Les applications médicales nécessitent généralement plus qu'un middleware offrant des services de soumission de tâches et d'accès aux données. Une expérience médicale implique souvent plusieurs algorithmes, et un ensemble de traitements qui peuvent parfois s'exécuter simultanément. Les traitements pipelines sont des tâches complexes qui passent par des étapes élémentaires enchaînées. Le projet EDG a développé un Directed Acyclic Graph (DAG) Service de soumission de tâches permettant de décrire celles composées tel que le graphe des processus élémentaires. Le gestionnaire de tâche DAG est un régulateur de vitesse de calcul. Les pipelines sont d'un intérêt réel lors du traitement de grand nombre de données d'entrée.

Calcul parallèle : Certaines tâches comme le traitement des images, la simulation et la modélisation des algorithmes de calcul intenses, ont besoin d'une mise en œuvre d'une exécution parallèle afin d'achever les opérations demandées dans des délais raisonnables. Le parallélisme au réseau local est aujourd'hui largement disponible via le message passing interface (MPI), Le projet EDG a développé une interface de travail parallèle basé du MPICH-G2 (MPICH de Globus Toolkit2).

Applications interactives : L'interaction avec l'utilisateur est nécessaire pour contrôler un algorithme, pour résoudre des problèmes juridiques lorsqu'il s'agit de données médicales, ou pour l'application elle-même (le simulateur de thérapie par exemple). La compression de données et la gestion de réseaux à haut débit devraient être effectuées en un temps limité pour un usage interactif. Le Feedback interactif implique souvent la visualisation 3D de scènes médicales. Cela représente un défi en raison de la grande taille des images médicales 3D et la complexité des anneaux utilisées pour la modélisation.

Dans un autre projet, les auteurs ont prouvé l'utilité des grilles dans les Systèmes CBIR dans le domaine médicale [55]. La gestion classique de grande quantité d'informations est une tâche délicate, qui motive la création de simples applications Web afin de gérer et utiliser le mieux les données collectées à partir de différentes bases d'images au niveau des cliniques. L'auteur montre comment MedGIFT [56] (Medical GNU Image Finding Tool) peut être utilisé pour exécuter des requêtes de recherche sur plusieurs bases d'images médicales. En utilisant MedGIFT, les médecins peuvent utiliser le matériel visuel lié au patient comme les Radiographies, et cherchent les similitudes dans les images des patients précédents.

Le projet MedGIFT des hôpitaux universitaires de Geneva a adopté la recherche des images à base de contenu. Le département de Radiologie produit en 2007 plus de 70.000 images par jour. Un tel nombre, rend la recherche des images à base de contenu une tâche gourmande, leurs indexation nécessite plus de 20 heures de calcul avec des CPUs performants.

Les outils du calcul distribué peuvent fournir une solution possible pour couvrir les demandes excessives des ressources de calcul dans les systèmes CBIR. L'environnement de calcul distribué utilisé est ARC (advanced resource connector). Le projet MedGIFT essai d'utiliser les nouvelles méthodes et technologies d'informations dans le domaine médical. Pour mettre ces systèmes en usage réel dans le domaine médical, une infrastructure de recherche a été mise en place.

La recherche se fait en élaborant d'avantage de grains fins (et donc souvent de calculs plus coûteux) des descripteurs visuels destinés à une future utilisation. En outre, le nombre d'images à traiter est en constante augmentation, et le projet vise à répondre aux besoins de ressources de calcul et de stockage.

La figure 4-9 illustre l'interface Web de l'utilisateur de l'application MedGIFT. Le médecin lance une recherche en fournissant une requête image, à base de laquelle le système va chercher les cas similaires. L'utilisateur assigne une évaluation des images trouvées pour être prise en considération dans une future recherche.

Figure 4-9: Interface Web de système MedGIFT.

Une solution de calcul distribué est proposée, elle est utilisée pour extraire les caractéristiques visuelles d'une collection d'images. Les caractéristiques visuelles représentent l'image dans les bases de données, et elles sont utilisées pour calculer les distances entre les images. Pour

optimiser les algorithmes et les paramètres utilisés dans la tâche d'extraction, le spécialiste de l'informatique médicale a besoin d'exécuter la tâche d'extraction plusieurs fois pour optimiser l'ensemble de fonctionnalités. La solution rend cette tâche beaucoup plus efficaces, et permet de procéder à la transformation de toute la collection d'images plusieurs fois durant une seule journée de travail.

Pour l'implémentation, les auteurs ont utilisé le middleware ARC pour obtenir un accès aux ressources informatiques disponibles grâce à la collaboration NorduGrid [58] au sein du projet KnowARC. Le travail s'est fondé sur l'utilisation de la méthode standard pour définir les Jobs ARC à l'aide de descriptions de Jobs. En outre, ils ont utilisé un gestionnaire de Job dans la grille pour coordonner leurs exécutions distribuées.

Le Job est exécuté de façon séquentielle sur une ressource distante de la grille. Un Job typique a un ou plusieurs fichiers d'entrées et un exécutable, il est utilisé pour produire le(s) fichier(s) de sortie(s). Un Job ARC basé sur la grille est défini par une description avec XRSL (extended resource specification language). Chaque Job a sa propre spécification et peut être exécuté indépendamment des autres Jobs.

```
1: (* executable for the feature extraction *)
2: ( executable=’’python-local.py’’ )
3: (* 1st argument, images for feature extraction*)
4: ( arguments="imgs-0.tar")
5: (* inputs, images, extractor source, local coordinator *)
6: ( inputFiles=(imgs-0.tar "") (src.tar "") (local.py ""))
7: (* stdout file for storing the results *)
8: ( stdout=’’stdout-gift-0.tar’’ )
9: ( outputFiles=("fts-0.tar" "gsiftp://dn.ch/fts-0.tar"))
10: ( stderr=’’geneva-gift0.err’’ )
11: (* jobname for easy monitoring *)
12: ( jobname="geneva-gift0")
```

Figure 4-10: Une description XRSL de Job pour l'exécution distribuée avec l'utilisation du middleware ARC.

La figure 4-10 illustre un exemple de description de Job ARC. Chaque Job envoyé pour exécution via le middleware ARC est défini par une description. La planification est réalisée au niveau client ARC. Après que le client choisi les ressources éloignées où le Job sera exécuté, les fichiers d'entrée comme défini dans la ligne 6 de la figure 4-10 sont téléchargés par la ressource où le Job sera exécuté. La ligne 2 de la description définit le fichier qui sera exécuté sur la ressource distante. Le fichier indiqué par l'attribut exécutable est également

transféré. Après la réussite de l'exécution du Job, l'Intergitiel de la grille écrit les résultats dans un fichier de sortie défini à la ligne 9.

La description du Job définit les opérations et les paramètres de grille, elle est utilisée pour choisir la ressource distante. Une fois le fichier exécutable et les fichiers d'entrée sont téléchargés sur la ressource distante, l'exécutable est lancé comme un processus indépendant. Les auteurs ont créé un exécutable qui se charge de coordonner l'exécution du Job d'une manière séquentielle au démarrage. L'exécutable est un script Python lancé sur l'ensemble de données en entrée, et produit en conséquence une sortie. Ensuite, selon la description du Job, les résultats sont rédigés localement pour être récupérés par le client, ou par l'Intergitiel de la grille qui s'occupe de leurs transferts vers un répertoire distant. Dans le premier cas, le client doit récupérer les résultats des Job après leurs achèvements, alors que dans le dernier cas, les résultats sont immédiatement transmis vers une ressource désignée par l'utilisateur.

Le gestionnaire du Job de la grille [59] (GridJM), est un système de gestion des calculs en parallèle dans une grille. Le système vise à adapter dynamiquement les ressources disponibles. L'utilisateur est doté d'une simple interface pour soumettre des tâches et de recevoir les résultats des Jobs, tandis que les détails de l'infrastructure de distribution sont masqués. Ces détails comprennent des mécanismes pour gérer les opérations spécifiques de la grille comme l'ordonnancement, le traitement fiable, la soumission des Jobs, le suivi et le transfert des résultats d'un Job.

La mise en œuvre de GridJM est faite sur la base du middleware ARC. La solution est également applicable à d'autres Intergitiels de grille. GridJM offre également une simple abstraction de grille pour l'utilisateur, le noyau de système est la vue dynamique des ressources distantes par l'utilisateur. Tandis que dans un environnement stable distribuées, seules les fonctions de présentation, de suivi et d'extraction de résultats sont nécessaires pour un fonctionnement efficace.

Pour diminuer d'avantage le risque des pannes, GridJM fournit également une couche d'évasion de faute (fault avoidance), fondée sur des observations d'historique à court terme des Jobs antérieurs. Le système suppose que si un Job qui a été récemment exécuté d'une manière non optimale sur une ressource précise, il est susceptible de mal exécuter d'autres Jobs dans le futur proche sur la même ressource. En conservant les informations relatives aux Jobs échoués, GridJM évite de soumettre des tâches à ces ressources.

Dans un autre travail : AGIR (Grid Analysis of Radiological Data) [82,83], les auteurs ont élaboré une structure générale pour les systèmes de traitement d'images à base de contenu déployés sur une plateforme de grille. La figure 4-11 montre la structure de AGIR.

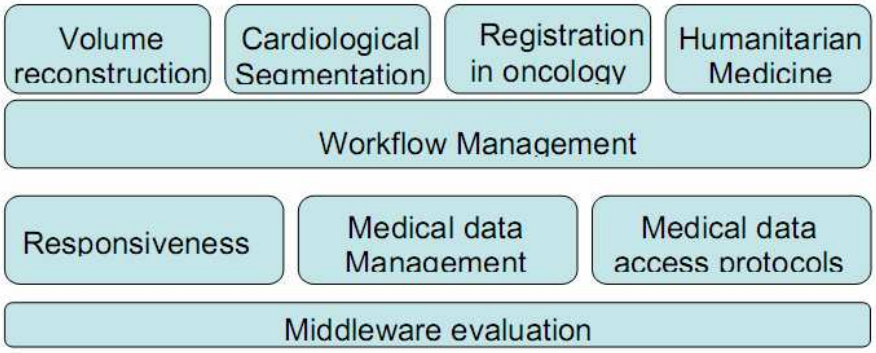


Figure 4-11: structure en couche de AGIR.

Dans la première couche, on trouve les services de base fournis par l'Intergitiel de la grille. La seconde contient les services de base dédiés aux applications médicales qui ne sont pas disponibles dans le middleware à usage général. Dans la troisième couche, on trouve les services implémentant les algorithmes de traitement d'images médicales, ils se basent dans leurs fonctionnements sur les services sous-jacents de la grille pour traiter de grandes quantités de données ou pour exécuter des calculs coûteux. Dans la dernière couche, on trouve les applications cliniques développées pour répondre aux besoins médicaux.

Dans le travail [81], l'objectif de la plateforme réalisée est principalement de permettre l'échange et l'utilisation de l'information médicale sur la grille de manière sécurisée

La plate-forme a été conçue pour satisfaire aux exigences complémentaires:

– Les applications fonctionnant sur la plateforme doivent êtres faciles à utiliser. En particulier, la complexité de la grille doit être complètement cachée.

– La plateforme devrait être adaptable à n'importe quel domaine médical.

– La plateforme doit être en mesure d'interagir avec les données des hôpitaux existantes et des systèmes d'archivage d'images (PACS).

– La plateforme doit être tolérante aux pannes.

– La plateforme ne devrait pas exiger une connectivité à grande échelle. Toute défaillance du réseau extérieur de l'hôpital ne devrait pas affecter la gestion des données à l'hôpital, ni à l'échange et le traitement de l'information médicale en dehors de la zone affectée par la défaillance.

– La plateforme doit remplir toutes les exigences légales en matière de sécurité.

L'architecture de la plate-forme est construite sur trois piliers:

a) Les données médicales sont stockées dans un serveur AMGA à l'intérieur de l'hôpital. AMGA est le catalogue de métadonnées développé par le projet EGEE abordé plus haut.

b) Les professionnels de la santé accèdent à tous les services de la plateforme à travers un portail Web développé avec le conteneur GridSphere.

c) L'infrastructure de grille fournit les moyens de stockage et les ressources de calcul nécessaires pour la sauvegarde des images médicales et le traitement des données en fonction des besoins des utilisateurs.

La figure 4-12 illustre l'architecture de la plate-forme (la partie gauche de la figure) et montre comment l'information est partagée entre les hôpitaux dans des endroits différents (partie droite de la figure).

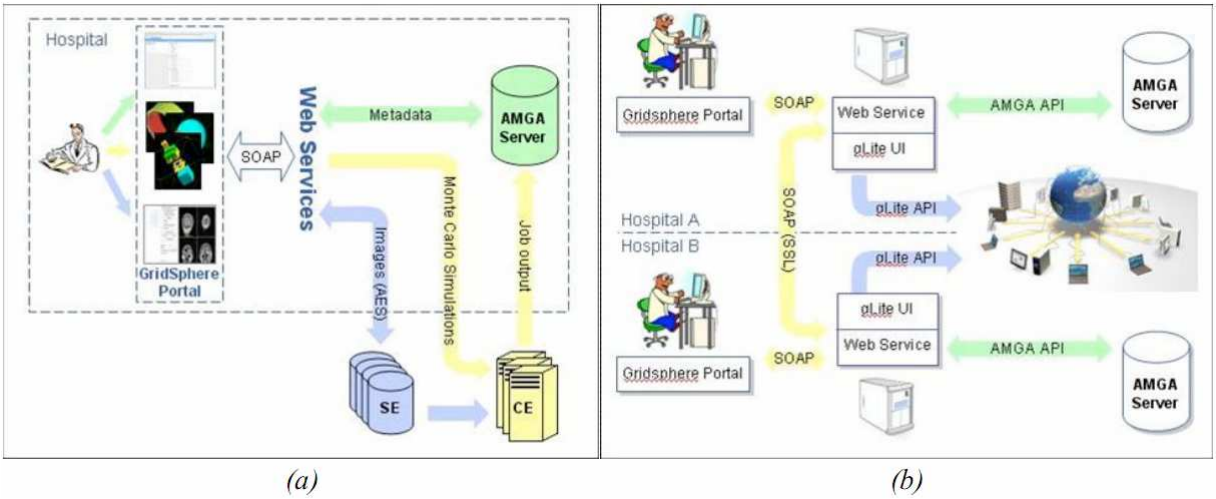


Figure 4-12 : (a) gestion des informations et des images à travers les hôpitaux; (b) partage des informations et des images entre les hôpitaux et les différentes stations.

Dans la conception de système, une séparation a été faite entre la couche de présentation, développée avec le conteneur Gridsphere, et la partie logique d'entreprise qui a été développé en utilisant les services Web pour gérer toutes les interactions de bas niveau avec les services de la grille. Cela fournit une interface personnalisée pour chaque client différent sans changer quoi que ce soit sur la gestion des données ou sur le côté interaction grille.

La conception de la plateforme facilite l'interface avec les services de gestion de données médicales (MDM). Le choix de la technologie de service Web est faite du besoin d'utiliser des protocoles standards de communication, tel que SOAP pour garantir l'interopérabilité et la sécurité entre les différentes instances de la plateforme installées dans les hôpitaux éloignés.

La couche de service Web gère l'ensemble des mécanismes de routage utilisé pour échanger des messages SOAP entre les instances distantes, ce qui permet aux médecins de différents hôpitaux de partager les fichiers des patients, ainsi que de partager les examens cliniques et les images médicales.

Les informations et les métadonnées extraites des images médicales sont gérées à l'aide de catalogue AMGA. Ce service de grille permet d'accéder à différentes bases de données back-end d'une manière uniforme et indépendante du système de base de données utilisé. Cela rend AMGA un service intéressant pour développer des solutions logicielles dans un environnement distribué et hétérogène.

Pour s'adapter aux systèmes d'information hospitaliers existants, l'architecture permet aux différents hôpitaux de spécifier un fichier qui représente la structure de base de données. Pour ce faire, les auteurs utilisent le langage XML pour représenter toutes les informations gérées et échangées dans la plateforme. Avec le protocole SOAP et la technologie Web Service, l'utilisation des documents XML pour stocker et échanger des informations rend la plateforme complètement indépendante et découplée de toute structure de base de données particulière déployée à l'intérieur de chaque hôpital.

4.3 Conception de plateforme CBIR basée sur Globus

4.3.1 Introduction

La plateforme est destinée aux traitements de masse des images, cela concerne la recherche à base de contenu et l'indexation pour de grandes bases d'images. La nature séquentielle des systèmes CBIR rend l'opération de la recherche très lourde et moins utile comme dans le cas des systèmes de traitement des images médicales, où la qualité des résultats et le temps de repense présentent un facteur important, à voire critique. La plateforme à présenter rentre dans le cadre des systèmes CBIR gourmands, ils demandent une grande puissance de calcul afin de clôturer des traitements coûteux sur des images de différents types et de différentes tailles. Partant de l'étape de prétraitement, à l'analyse jusqu'à la reconnaissance et la classification des images, le système a besoin de gérer plusieurs ressources de nature hétérogènes.

La technologie des grilles est largement utilisée dans les Systèmes CBIR pour partager, synchroniser et exécuter les tâches demandées, cela joue un rôle important dans l'augmentation des performances de la plateforme en question.

La plateforme CBIR réalisée dans cette étude porte sur l'utilisation des grilles de calcul avec un middleware largement utilisé Globus, cela est fait dans le but d'améliorer les performances d'un moteur de recherche des images à base de leurs contenues.

L'utilisateur saisit sa requête (image, mots clés), le système analyse cette dernière, il partage et lance les tâches nécessaires, et pour terminer, les résultats sont regroupés, structurés et fournis à l'utilisateur. Afin d'améliorer la qualité des résultats, d'autres fonctions interactives avec l'utilisateur sont disponibles. La plateforme est développée avec des technologies open source, ce qui offre la possiblilier d'ajout de nouveaux modules pour d'autres fonctionnalités.

Interface de passage de message (MPI)

Utilisé pour l'échange de message entre les processus lancés en parallèle, il peut être utilisé dans le développement des applications basées sur les grilles. MPI fournit une librairie pour le transfert de données et la communication entre les programmes en cours d'exécution sur plusieurs processeurs dans les systèmes distribués. Le standard MPI est conçu pour améliorer la portabilité des applications parallèles [86].

La boite à outil Globus, utilise la bibliothèque de communication MPICH-G2, ce dernier implémente MPI pour standardiser la communication via l'échange des messages entres les services des plateformes.

Architecture parallèle

On distingue deux architectures principales, une à mémoire distribuée et l'autre à mémoire partagée. Dans le cas de l'architecture à mémoire distribuée, les ordinateurs aient un accès rapide à leurs mémoires locales, chacun travail à son côté pour résoudre une partie du même problème. Les données sont échangées entre les nœuds (généralement via un réseau haut débit) sous forme de messages. Dans l'architecture à mémoire partagée, les processeurs utilisent un espace de mémoire commun via un bus mémoire à haute vitesse. Cette mémoire partagée permet aux processeurs d'échanger et partager l'accès aux données.

Typiquement le nombre de processeurs utilisés dans cette architectures est limité à seulement (2 - 16) processeurs [87]. C'est parce que la quantité des données qui peuvent être traitées est limitée par la largeur de bande de bus de mémoire.

Types de tâches dans une application parallèle

a) Les tâches séquentielles

Se sont des tâches qui s'exécutent sur un seul processeur. Simple à implémenter, beaucoup d'algorithmes parallèles se basent sur l'exécution séquentielle des tâches, ce qui n'est pas toujours performant [88].

b) Les tâches parallèles

Les tâches parallèles sont des tâches pouvant s'exécuter sur un ou plusieurs processeurs. On peut distinguer trois classes de tâches parallèles :

- Les tâches rigides où le nombre de processeurs qui doivent exécuter la tâche parallèle est fixé à priori.

- Les tâches modelables pour lesquelles le nombre de processeurs n'est pas fixé mais est déterminé avant l'exécution. Cependant, comme dans le cas précédent, ce nombre de processeurs ne change pas jusqu'à la fin de l'exécution.

- Les tâches malléables peuvent voir le nombre de processeurs qui leur est alloué changer au cours de l'exécution (par préemption des tâches ou par redistribution de données).

Décomposition de problème

La conception des algorithmes parallèles repose sur la décomposition du problème en sous problèmes. Ces derniers sont assignés aux CPUs pour les résoudre simultanément. Ils existent deux sortes de décomposition.

- Décomposition de domaine : (données parallèles), les données sont divisées approximativement de même taille, et fournies ensuite aux différents processeurs, chaque processeur travail sur la partie des données qui y est assignée. Les processus peuvent communiquer périodiquement pour échanger les données. Le parallélisme des données a l'avantage de maintenir un seul contrôle de flux. L'algorithme de données parallèle consiste en une séquence d'instructions élémentaires, une instruction ne peut être exécutée qu'après l'exécution de celle qui la précède. Single-Program-Multiple-Data (SPMD) suit ce modèle lorsque le code est identique sur tous les processeurs.

- Décomposition fonctionnelle : Dans la plupart des temps, la décomposition de domaine n'est pas efficace ou même n'est pas applicable à certains problèmes. Dans les algorithmes de parallélisme fonctionnel, la tâche est divisée en plusieurs sous tâches, ces derniers sont assignées aux processeurs lors leurs disponibilités, et le processeur qui termine le traitement plus vite, aura le maximum de tâches à exécuter.

4.3.2 Fonctionnalités de la plateforme

La plateforme est conçu pour la réalisation des tâches gourmandes en imagerie sur des bases d'images locales ou distantes, elle doit rependre aux attentes des utilisateurs, elle fournit les fonctionnalités suivantes :

A. Traitement de masse

Avec cette fonctionnalité, l'utilisateur peut réaliser les opérations de base tels que celles de la phase de prétraitement, cette dernière inclut la tâche d'application des filtres en choisissant un algorithme précis et implémenté par la plateforme. Ainsi, l'utilisateur peut effectuer d'autres manipulations comme la conversion du format des images, faire un zoom, ajouter un bruit, etc.

De même, l'utilisateur peut lancer la segmentation sur un ensemble d'images avec la possibilité de varier les valeurs des paramètres de la méthode de segmentation utilisée. Cette fonctionnalité permet la personnalisation de la segmentation selon les valeurs des paramètres fournies par l'utilisateur, le résultat de cette opération est représenté par un fichier (.seg) pour chaque image segmentée, où le fichier résultant contient des informations sur les régions issues de la segmentation telles que la taille, la position, les voisins de chaque région.

B. Reconnaissance, classification et indexation

Ces fonctionnalités sont nécessaires pour la gestion des classes des images. Toutes les informations sur ces classes sont stockées dans une base de données locale ou distante. Les étapes de reconnaissance, de classification et d'indexation sont :

1- La définition ou l'ajout des classes d'image : cette étape est effectuée en fournissant des descriptions telles que le nom, la catégorie, les caractéristiques descripteurs de la classe qui sont nécessaires dans la comparaison des caractéristiques visuelles entres images.

2- La suppression et la modification des informations sur les classes déjà définies.

3- Lancement de l'extraction des caractéristiques visuelles des images qui appartiennent à une même catégorie, cela résulte un fichier descripteur des caractéristiques d'une classe, ce fichier sera utilisé par la suite dans l'étape du calcul de distance entre les nouvelles images à classifier et les images de la classe en question.

4- Lancement de la classification sur une base d'images seule ou avec des fichiers (.seg) qui représentent le résultat de segmentation personnalisée d'images. Après analyse et extraction des caractéristiques descripteurs du contenu de chaque image, le système fait la classification en calculant la distance entre chaque nouvelle image et chaque classe. La classification est représentée par des pourcentages qui traduisent le taux d'appartenance d'une image à une classe. Le résultat de la classification est l'insertion des entrées dans la base de données, où chacune contient des informations sur l'image et sur les classes dont elle appartient. La structure de la Table Indexe est présentée dans la partie Annexe B sur une base de données Postgresql avec l'outil pgAdmin3.

C. Recherche

La fonctionnalité de recherche dépend principalement des caractéristiques descripteurs extraites dans l'étape d'indexation, on peut chercher une image qui possède un taux d'une couleur précise, et ou contient un ou plusieurs objets spécifiés. L'utilisateur lance une requête représentée par une image ou par un ensemble de mot clé comme par exemple « *voiture verte* », le système fait une analyse de la requête, il fait recours à une ontologie pour extraire les mots clés qui ont une relation avec ceux en entrés, il construit une nouvelle requête sous format SQl, et à la fin, il interroge la base de données qui contient les descriptions suffisantes pour rependre à la requête. Le résultat de la recherche est une liste d'images triée et affichée selon un certain ordre.

4.3.3 Structure de la plateforme

La plateforme est basée sur la technologie des grilles, elle se fond sur les web services, elle utilise le principal de l'architecture OGSA et implémente les spécifications WSRF comme le montre la figure 4-13.

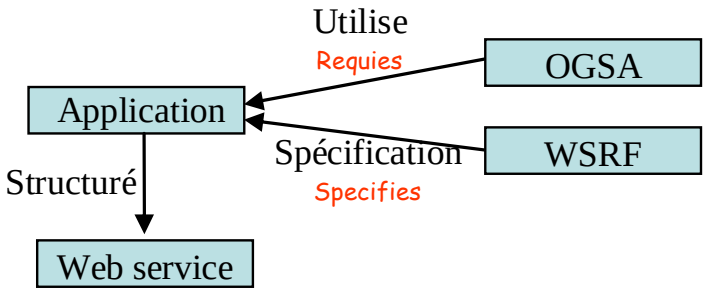


Figure 4-13: plateforme et relation avec les autre technologies de grille.

4.3.4 Architecture de la plateforme :

La plateforme est composée de modules, où chacun regroupe des outils, des services et des APIs nécessaires. L'architecture de la plateforme peut être vue sous forme de couches comme montrée par la figure 4-14.

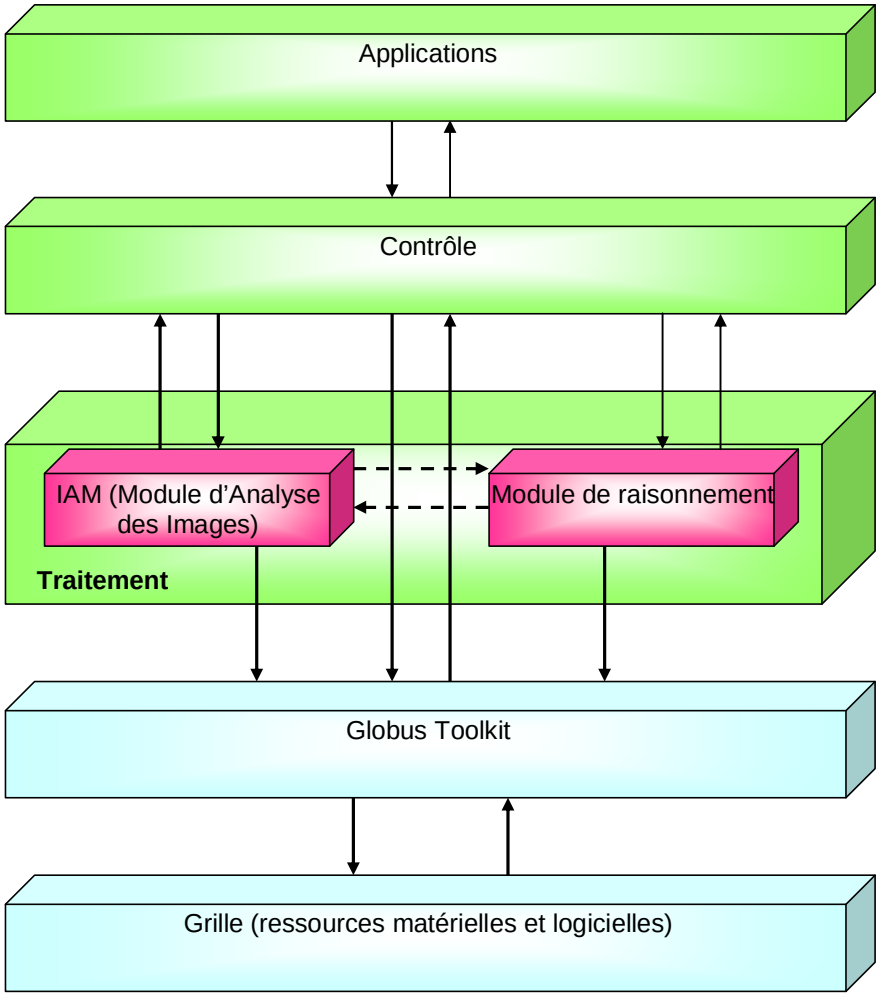


Figure 4-14: Architecture en couche de la plateforme.

Les couches de la plateforme sont:

A. Ressources matérielles et logicielles (La grille)

Elle englobe le matériel informatique (CPUs, unités de stockages, réseau..) et logiciel (Système d'exploitation, utilitaires, applications..). La plateforme est déployée sur une grille de test, elle est composée de quatre nœuds dotés d'un système d'exploitation Linux et d'outils de développement.

B. L'Intergitiel Globus

Il fournit des services de grille qui permettent l'exploitation des ressources. Avec les différents modules de Globus, les services de la plateforme peuvent transférer des fichiers, gérer l'exécution des tâches, sécuriser les transactions et récupérer les informations. L'installation de l'Intergitiel Globus est détaillée dans l'Annexe A.

C. La couche traitement (Processing)

Elle contient des services de grille destinés à réaliser les traitements basiques en imagerie. On trouve à ce niveau une implémentation de différentes méthodes de raisonnement et de prise de décision, elles sont utilisées pour la reconnaissance et la classification des images. Cette couche reçoit des requêtes de la couche supérieure, elle fait appel au service demandé qui procède via la couche inférieure, cette dernière lui fournit les ressources de calcul et du stockage nécessaires. Les résultats sont structurés et adaptés pour êtres bien exploités, organisés, combinés si nécessaire avec d'autres résultats et présentés à l'utilisateur final.

La couche Processing comporte deux principaux modules:

3.1- IAM (Module d'analyse des images) : il contient des services de grille, où chacun est responsable de réaliser une opération en imagerie, ce module peut être vu comme une librairie utilisée pour le traitement des images. Les fonctionnalités fournies par cette librairie sont : le filtrage, la segmentation, extraction de régions pertinentes, extraction des vecteurs de caractéristiques visuelles et d'autres opérations.

L'ajout d'une nouvelle fonctionnalité se traduit par : 1) la création et le déploiement d'un service de grille qui implémente la solution demandée, 2) adaptation de la partie cliente de l'application pour les nouvelles fonctionnalités ajoutées, 3) injecter la méthode d'appel du service au niveau client de l'application comme le montre la figure 4-15. On note que l'ajout d'une fonctionnalité dans le module IAM ne nécessite pas le redéploiement des anciens services qui se fait généralement avec des fichiers GAR et qui sont équivalents aux fichiers War pour les applications web en Java, mais l'ajout nécessite le redémarrage du conteneur.

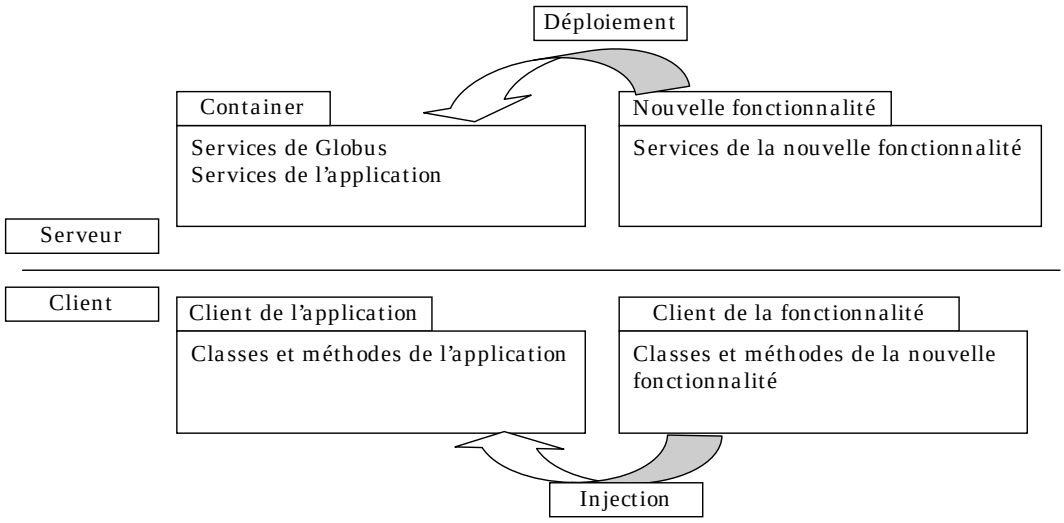


Figure 4-15 Ajout d'une nouvelle fonctionnalité au module IAM

Les services de la couche supérieure invoquent ceux du module IAM, ces derniers reçoivent les paramètres nécessaires pour leurs exécutions sous forme d'un fichier XML. Avec ces

fichiers, les services du module IAM localisent les données à utiliser et l'emplacement où ils doivent sauvegarder les résultats.

La structure générale du fichier XML utilisé est comme suit :

```
<request>
  <description>
        <serviceID>
        </serviceID>
        <host>
        </host>
  </description>
  <in>
        <dataSource>
        </dataSource>
  </in>
  <out>
        <host>
        </host>
        <path>
        </path>
  </out>
</request>
```

– Balise *request* : c'est la balise globale de la requête récupérée par les services de IAM.

– Balise *description* : contient l'URI du service demandeur et le host correspondant.

– Balise *in* : contient des références sur les sources de données à exploiter. Ces derniers contiennent généralement des images, ou dans d'autres cas, des fichiers représentants des résultats des opérations antérieures tels que les fichiers .seg de la segmentation.

– Balise *out* : elle décrit le chemin de la sortie des opérations. On ne trouve pas cette balise dans le cas des traitements qui modifient directement les images.

A la fin d'exécution du service interrogé, le module IAM affecte un état à chaque processus lancé sur une source de données. Cet état peut être soit une réussite, soit un échec, dans ce dernier cas, une description de l'exception est ajoutée au résultat de l'exécution. A la fin, la requête d'état est retournée au service appelant. La figure 4-16 montre l'interaction entre la couche contrôle et le module IAM.

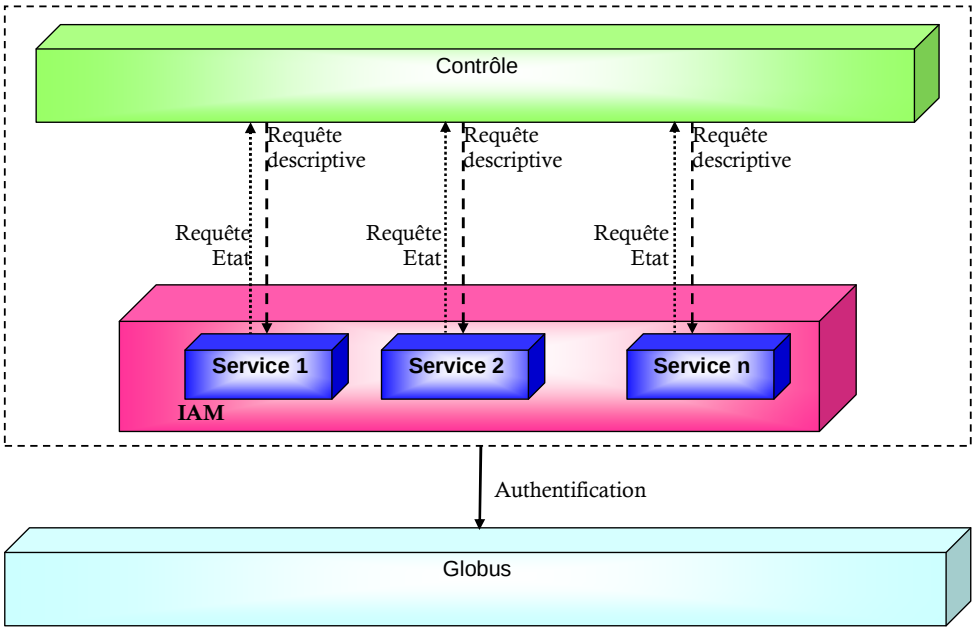


Figure 4-16 : interaction entre le module IAM et la couche supérieure

Les principaux services du module IAM sont :

Service du filtrage : instancié lors des opérations de prétraitement sur une collection d'images, il reçoit les paramètres nécessaires dans la requête descriptive. Parmi ces paramètres, on peut trouver le type de filtre à appliquer. Le résultat de succès de cette opération est envoyé via la requête d'état. Dans le cas où l'utilisateur veut que les résultats de filtrage n'écrasent pas les images de la source, il spécifie l'adresse de stockage des résultats pour chaque source de données.

Service de segmentation : ce service fournit une méthode de segmentation personnalisable, il implémente l'algorithme de type croissance de régions. Comme le cas du service du filtrage, il reçoit les paramètres à utiliser via la requête descriptive. Les paramètres qu'on peut avoir pour ce service sont la variance, le nombre de voisins et la taille maximal des régions, ces paramètres sont utilisés par l'algorithme de segmentation pour l'étape de fusion des régions. Le résultat de cette opération est représenté par des fichiers .seg contenant une description de chaque segment pour chacune des images traitées, ainsi les résultats sont sauvegardés dans le média précisé par la requête utilisateur.

Service de détection des régions optimales : cette opération est traduite par la combinaison conditionnée des régions après segmentation, cette combinaison est faite dans le but de construire une carte des régions d'intérêts. Elle est réalisée avec la coopération entre les services de segmentation et le module de raisonnement au niveau de la couche traitement (processing). La carte des régions optimales est sauvegardée dans des fichiers .rop (Régions optimales), ils décrivent les régions jugées optimales, leurs positions, leurs voisinages et leurs tailles…

Service d'extraction des caractéristiques : à partir des régions optimales extraites par le service cité précédemment, ce service extrait un ensemble de caractéristiques en utilisant les matrices de co-occurrence, il crée des vecteurs nommés les vecteurs de caractéristiques visuelles sur lesquels se base la classification des images.

Services de traitement de base : il fournit des fonctionnalités basiques pour le traitement des images tel que le Zoom, l'ajout de bruit…, par le même principe, il reçoit ses paramètres dans la requête descriptive. Ce service applique des fonctions mathématiques sur chaque pixel de l'image, il demande plus de ressources de calcul.

3.2- Module du résonnement (Reasoning module) : contient un ensemble de services de grille, utilisé principalement pour l'aide à la décision et le classement des images.

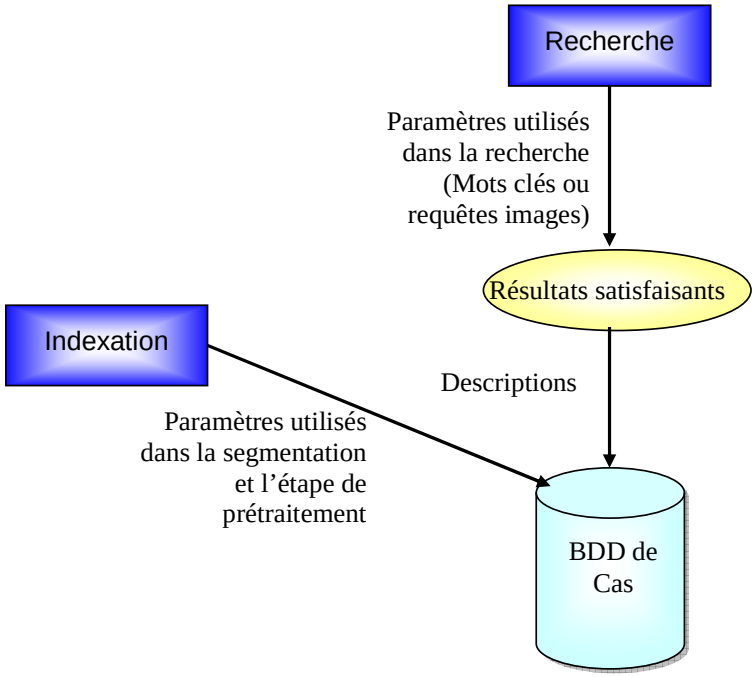


Figure 4-17 Informations sauvegardées dans la base de cas

Le module du raisonnement utilise un CBR pour la recherche des images avec des mots clés déjà utilisés, ou avec des requêtes images dont les caractéristiques visuelles ressemblent à

d'autres déjà traitées. Le CBR sauvegarde les cas des requêtes qui ont eu des résultats jugés satisfaisants.

La figure 4-17 montre les sources d'informations sauvegardées dans la base des cas. Le CBR est utilisé pour améliorer la recherche des images en termes de temps et de qualité.

Le module de résonnement est constitué d'un service qui implémente le CBR, et un autre de calcul de distances entres vecteurs de caractéristiques extraites afin de classifier les images.

Service avec raisonnement à base des cas : il utilise une base de données pour sauvegarder les traces des opérations de recherche effectuées et jugées satisfaisantes par les utilisateurs (retour de pertinence). La base de données contient une entrée par opération de recherche, on sauvegardes les mots clés pour les requêtes textes, et le vecteur des caractéristiques d'une image dans le cas d'une requête image, on sauvegarde aussi le fichier descriptif du résultat et la trace des opérations de traitement effectuées durant l'indexation pour les reproduire sur un traitement des requêtes similaires.

Service de calcul de distances entre vecteurs de caractéristiques : utilisé pour la classification des images. Pour chaque vecteur de caractéristiques décrivant les régions d'une image, on calcul leurs distances avec les vecteurs représentants une autre image déjà classifiée. Après toute une série de calcul, on affecte la nouvelle image à la classe avec laquelle la distance été minimale. Un classement supervisé peut être effectué afin d'améliorer la recherche et l'indexation.

La couche contrôle

Elle contrôle tous les services de la couche traitement et s'interface avec la couche application. Elle contient des services de grille pour la recherche, l'indexation, la gestion des classes des images et un dernier pour la collection des informations et la synchronisation entres les services.

La couche contrôle fait référence à celle inférieure (Processing) pour lancer les traitements demandés, et à l'Intergitiel Globus pour effectuer des opérations sur la grille tel que le transfert des fichiers, la collecte des informations, le lancement des Jobs et la sécurisation des transactions entres les services.

Durant l'indexation et la gestion des classes d'images, les services responsables mettent à jours la base de données des classes existantes et des images déjà indexées. La figure 4-18 montre l'interaction de la couche contrôle avec les autres couches de la plateforme.

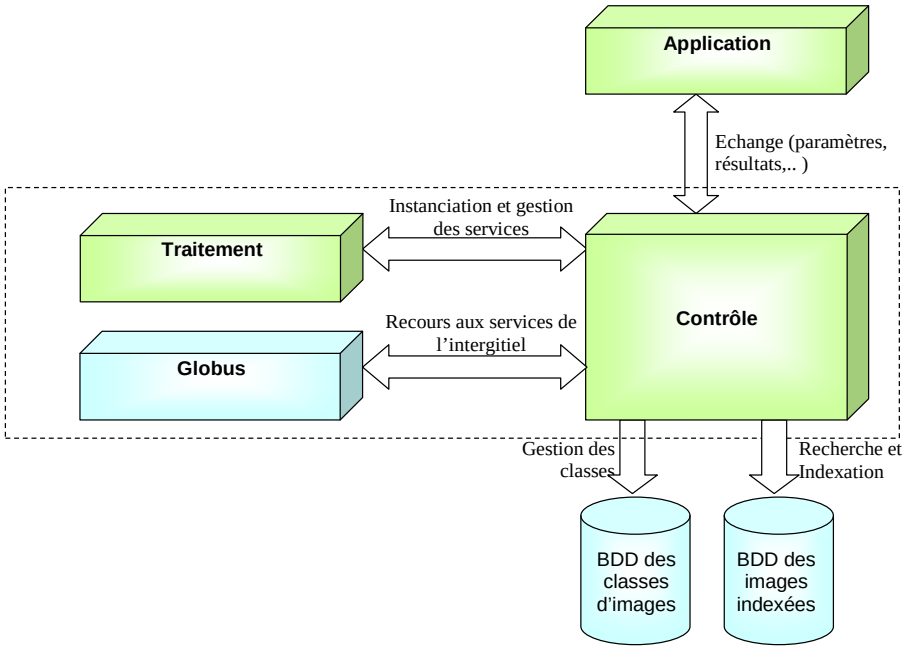


Figure 4-18 : Interaction du module contrôle avec les autres modules de la plateforme

Service gestionnaire des classes : utilisé pour la création, la modification et la suppression des classes d'images. Les informations des classes sont sauvegardées sur une base de données dans la grille. La couche application fournit à ce service une interface graphique facilitant à l'utilisateur la gestion des différentes classes. Dans le cas d'une suppression ou d'une modification, le service reçoit les paramètres nécessaires et crée une connexion avec la base de données, il modifie ou supprime les entrées en question, cette action fait recours aux services de l'Intergitiel Globus pour lancer les tâches à distance afin de mettre à jours la base

de données comme illustré dans la figure 4-19. Dans le cas de création de nouvelles classes d'images, le système récupère les informations nécessaires telles que le nom et la catégorie.

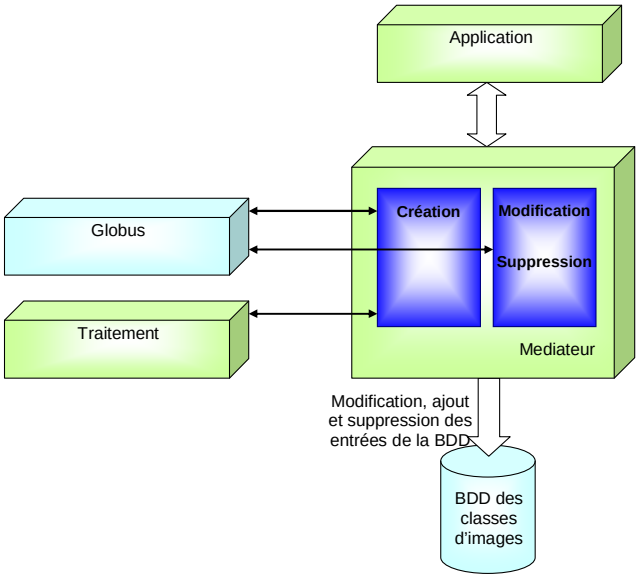


Figure 19: gestion des classes d'images

La création d'une nouvelle classe est accompagnée par la génération d'un ensemble de fichiers descripteurs pour chaque image de la classe, cela et réalisé par l'extraction des caractéristiques sur une base d'images qui représente un objet ou une notion. Cette action est faite au niveau du service d'extraction des caractéristiques du module precessing. Les chemins (Path) des fichiers descripteurs sont sauvegardés avec d'autres informations concernant la nouvelle classe, tel que le nom, la couleur et la nature de l'objet à représenter. La sauvegarde est effectuée au niveau de la base de données des classes. La création d'une classe nécessite une capacité de calcul qui dépend du nombre, de la qualité et de la taille des images utilisées dans la représentation de l'objet, cette capacité de calcul est assurée par la grille en général.

Le service d'indexation : ce service utilise une ontologie et une base de données pour sauvegarder les informations concernant les images indexées. L'indexation se fait sur une ou plusieurs bases d'images seules ou avec des fichiers (.seg) issus d'une segmentation déjà effectuée.

Le service d'indexation procède comme suit :

- Si les images ne sont pas segmentées, il fait appel à la couche processing pour lancer les opérations avec les paramètres de prétraitement et de segmentation par défaut.

- Pour chaque image à indexer, il appel le Service de détection des régions optimales et le service d'extraction de caractéristiques de la couche Processing. Cette étape résulte un fichier descripteurs par région optimale, et un autre pour toute l'image.

- Il appel le service de calcul de distance et du classification des images, une image peut être classifiée sous plusieurs classes en affectant un pourcentages. De même, une image peut être affectée à une ou plusieurs autres classes après consultation de l'ontologie. L'objectif de l'ontologie est de modéliser un ensemble de connaissances dans un domaine donné. Pour un objet, on définit les mots clés similaires, ses composants, et son contexte pour rendre le processus de recherche et d'indexation plus efficaces.

- Le service d'indexation crée des nouvelles entrées dans la base des images indexées, il sauvegarde des informations utiles comme le pourcentage d'existence d'une couleur dans une image, les étapes et les valeurs des paramètres avec lesquels le processus d'indexation a passé, ..

Service de recherche : selon le type de requête utilisateur (textuelle ou image), on distingue deux mode de recherche. La première porte sur une analyse textuelle, et la seconde sur l'extraction des caractéristiques visuelles des régions optimales d'une image requête.

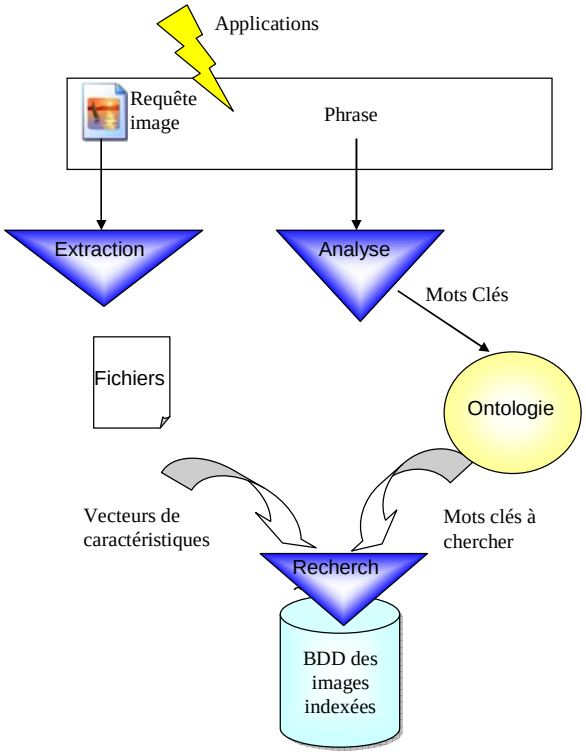


Figure 4-20 :L'opération de recherche à base de contenu.

- pour une requête image, le service fait appel au module IAM pour analyser l'image. Il extrait leurs vecteurs descriptifs et les compare avec ceux stockés dans la base de données des images indexées. La comparaison entres les vecteurs des images résulte une distance. Le module IAM retourne le résultat de ses calculs sous forme d'un taux (distance) avec un ensemble de liens des images dont la distance été minimale. A la fin de l'opération, le service de recherche affiche les images selon un ordre décroissant de la distance calculée. La figure 4-20 montre le processus utilisé pour l'indexation et la recherche implémentée.

- Pour une requête textuelle, le service de recherche fait recours à l'ontologie pour trouver toutes les dépendances des mots clés en entré. le service cherche les références des mots clés dans la base de données des images indexées.

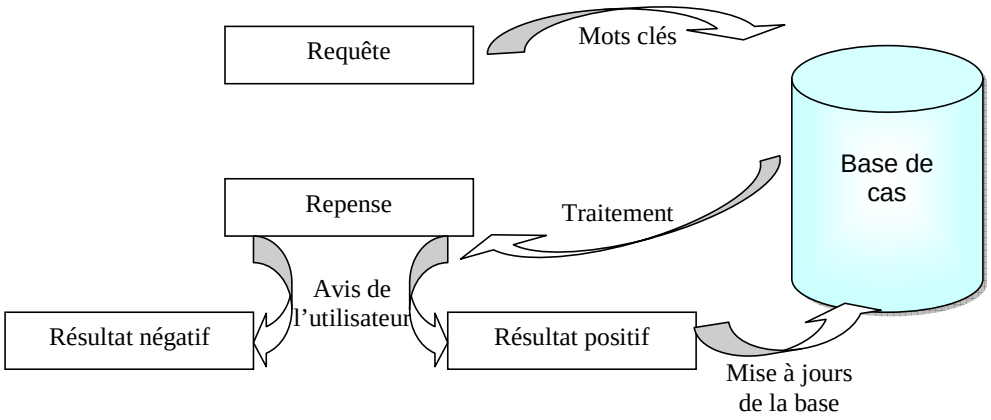


Figure 4-21 : introduction du CBR dans la recherche.

La recherche avec retour de pertinence est une fonctionnalité dans laquelle l'avis de l'utilisateur est pris en considération, il peut juger le résultat d'une recherche s'il est positif ou non.

Dans le cas où le résultat de recherche satisfait les besoins de l'utilisateur, les paramètres utilisés dans la recherche sont sauvegardés dans la base des cas via le service de raisonnement à base de cas afin d'améliorer les futurs recherches qui portent sur le même contexte. Dans une seconde opération de recherche, le système fait recours à la base des cas afin de vérifier l'existence des résultats satisfaisants, et qui corresponds aux mots clés de la nouvelle requête en entrée, la figure 4-21 montre l'introduction de CBR dans le processus de la recherche.

Service d'information et de synchronisation (SIS): ce service est le noyau de la plateforme, il est responsable de la gestion des tâches et la synchronisation des services. Ainsi, il est déployé au niveau de chaque nœud de la grille, il collecte les informations dynamiques via les

commandes fournies par Globus tel que *globus-wsrf-get-property*, et *globus-wsrf-get-properties.*

Les principales fonctionnalités offertes par ce service sont :

- la collecte des informations sur les Jobs: SIS fait recours au module MDS de l'Intergitiel Globus pour collecter les informations sur : (1) l'ensemble des tâches en cours d'exécution, (2) les files d'attente, (3) nombre de Jobs dans chaque nœud, (4) services lancés dans le conteneur d'un hôte distant…

- La collecte des informations concernant les ressources matérielles : afin de gérer l'exécution des Jobs en tenant compte de la charge globale de la grille, la plateforme a besoin des informations sur chaque nœud telles que la mémoire physique totale, la mémoire disponible, la vitesse du CPU et la mémoire de stockage disponible.

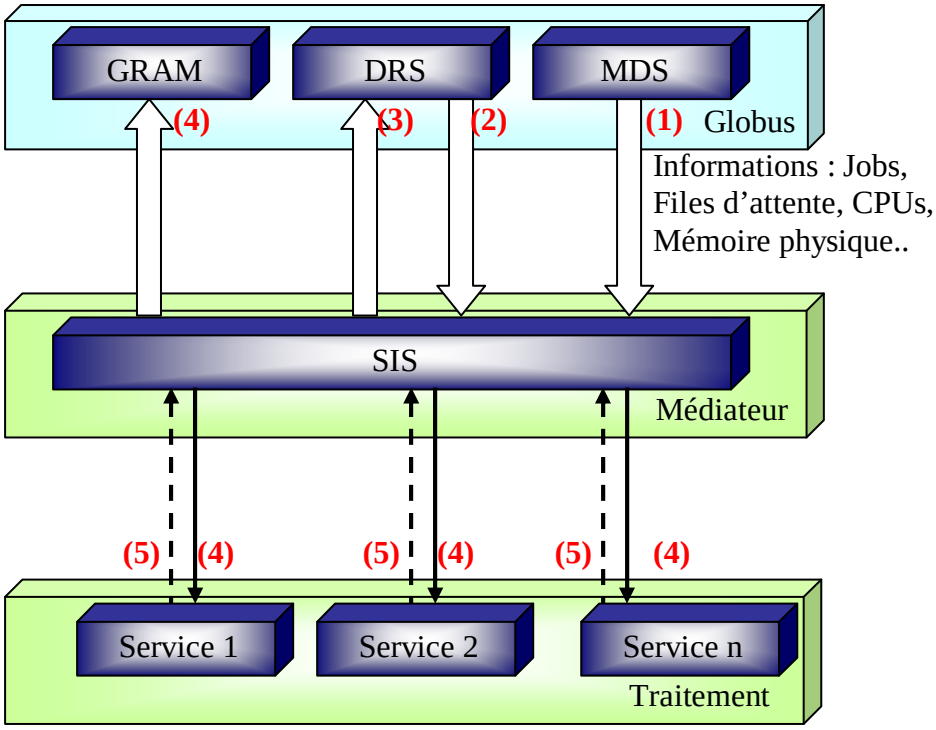
- Transfert de données et Lancement des Jobs : Avec les informations collectées à partir du MDS, le service localise les nœuds les moins chargés pour l'exécution de quelques Jobs en parallèle. Avec le module de gestion de données de l'Intergitiel Globus (Service DRS), on peut savoir si les données nécessaires pour l'exécution des Jobs sont disponibles sur le nœud ou pas, sinon dans ce dernier cas, le service utilise les ressources distantes, lance le job et ensuite, la gestion des données et la réplication en question.

- Le regroupement et l'analyse des résultats : à la fin de chaque Job, le service procède selon la natures des résultats, dans le cas où ces derniers sont des fichiers à recopier vers une média cible, il lance le service de transferts de fichier gridFTP, et un état de bon achèvement de tâche est récupéré afin de mettre à jour l'état des Jobs Lancés. Après, une autre collection de Job est lancée sur les hôtes libérés et une mise à jour de l'état globale de la fonction à exécuter est effectuée. Autrement, si les résultats sont des insertions dans la base de données des images indexées, ou des modifications directes sur les bases des images, une remise est récupérée. De même, la gestion des Jobs est révisée.

- La synchronisation entre les services : pour gérer les tâches parallèles et les tâches séquentielles, le service (SIS) a besoin de savoir le comportement à prendre envers un service en cours d'exécution, ce type d'information peut être trouvé dans les fichiers de configuration de la plateforme. La synchronisation est gérée entre deux services dont le second utilise les résultats du premier. La réalisation d'une fonctionnalité par la plateforme peut provoquer l'appel d'une dizaine de services, où pour chacun, on peut créer une autre dizaine d'instances,

les services communiquent entre eux avec des messages et un service ne peut être lancé qu'après sa notification par le service d'information et de synchronisation.

La figure 4-22 montre la relation entre le service d'information et de synchronisation avec les services de l'Intergitiel Globus, les services du module contrôle et les services de la couche Processing.



(1) : Collecte des informations et désignation des hôtes les moins chargés.
(2) : Vérification de l'existence de données dur les hôtes cibles.
(3) : Lancement de la réplication de données.
(4) : Lancement des Jobs et des instances des services sur les hôtes cibles.
(5) : Transmission des descripteurs de résultats.

Figure 4-22: collection des informations auprès du service MDS, vérification de l'existence des données et lancement des tâches sur les hôtes moins chargés.

D. Application

Cette couche fournit le côté client des services tels que celui responsable de l'indexation, la recherche, la gestion des classes. Avec ces interfaces GUI, cette couche permet de spécifier les paramètres utiles pour l'exécution des services de la couche contrôle. Après saisi des paramètres, elle appel le service SIS en utilisant son URI.

4.4 Résultats

Dans cette section, on va présenter les résultats obtenus pour plusieurs scénarios d'exécution. La grille de test installée est composée de quatre machines dont les caractéristiques sont illustrées dans la table 1.

Nom de la machine	Adresse IP	Description	Caractéristiques
Poste1.lri-annaba.net	192.168.0.5		2 CPU Intel(R) Core 2 Duo T7200 (2GHz/667MHz/4MB) + RAM 2 Gio + 150 GB HDD espace libre
Poste2.lri-annaba.net	192.168.0.103		Idem Poste1
Poste3.lri-annaba.net	192.168.0.104		Idem Poste1
Poste4.lri-annaba.net	192.168.0.101	propriétaire de certificat	2 CPU Intel(R) Pentium (R) Dual CPU T2330 @ 1.60 GHZ + RAM 2 Gio + 100 GB HDD espace libre

Tableau 1: machine composantes de la grille de test.

Le tableau 2 fait l'objet d'une comparaison entre le temps d'exécution des services de segmentation lancés sur un nombre variant de postes. La base des images utilisée est Wavelet-based Image Indexing and Searching (WBIIS) de Wang [89] (Stanford University), elle contient 10.000 images de résolution 128x128. Dans le cas de l'exécution parallèle, on a fait une division de domaine, où chacun contient 1000 images. Les temps d'exécution illustrés dans le tableau 2 n'inclut pas le temps de transfert de résultats vers les médias cibles, ni le transfert de données de domaine vers la machine d'exécution. L'exécution séquentielle est lancée sur une seule machine avec le même algorithme de segmentation utilisé par les services lancés précédemment sur la grille.

Nbr images /Nbr Machines	Une seule machine (Séquentiel)	Deux machines (Parallèle)	Quatre machines (Parallèle)
1.000	12.67 min	13.82 min	13.87 min
5.000	53.26 min	38.71 min	27.12 min
8.000	102.78 min	54.95 min	29.05 min

10.000	117.13 min	64.33 min	39.98 min

Tableau 2: comparaison des temps CPU de segmentation d'images sur un nombre variant de machine.

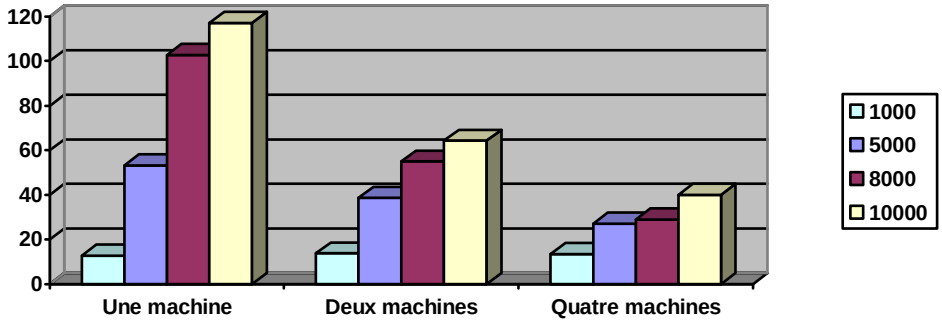


Figure 4-23:Diagramme de comparaison.

Lorsque le nombre des images à segmenter est inférieur ou égale au nombre des images du domaine, une seule instance du service de segmentation est lancée et le temps CPU de l'exécution séquentielle est plus performant que celui de l'exécution parallèle. Et dans le cas où le nombre des images à segmenter est plus élevé, le nombre d'instances du service de segmentation égale au nombre des images à traiter divisé par le nombre des images par domaine, et le temps CPU de segmentation et en relation inversement proportionnel avec le nombre des machines incluses dans l'opération. Le temps d'exécution devient plus stable lorsque le nombre de machine est supérieur ou égale au nombre d'instances de service de segmentation lancées. Le nombre d'instance du service de segmentation est variable d'une machine à l'autre, tout en dépendant de l'emplacement des données sur la grille et la puissance de calcul du CPU. La machine qui possède le CPU le plus performant et la mémoire physique la plus élevée, achève le traitement plus vite et exécute la grande partie de l'opération.

Le deuxième test consiste à lancer la segmentation en incluant le temps de transfert de données et des résultats. Le test porte sur 10.000 images avec une taille de 34.8 MO, elles sont situées sur un seul poste, là où la demande de segmentation été marquée. Pour lancer la segmentation sur cet ensemble d'images, le service d'information et de synchronisation (SIS) localise l'emplacement des données sur la grille (la source de donnée est spécifiée dans la requête de l'utilisateur), par la suite, SIS localise les réplicats, partage le domaine des données (100 images par domaine), lance les instances du service de segmentation sur les nœuds qui contiennent les données, et enfin, il lance la réplication des données d'un domaine précis ainsi qu'une instance de service par domaine répliqué.

Les résultats de cette opération sont récupérés auprès du nœud qui a demandé l'exécution de la tâche de segmentation.

Le tableau 3 illustre les résultats obtenus, il présente le temps CPU de l'exécution sur chaque nœud, le temps de transfert et de récupération de données, le nombre d'instances de service de segmentation lancées sur chaque nœud, et à la fin, un récapitule du temps global pour la segmentation de 10.000 images sur la grille.

Le tableau 3 n'illustre pas la chronologie des étapes avec lesquelles passe la segmentation, ni le temps totale de son achèvement, la figure 4-24 illustre la chronologie des actions et le temps de chaque événement avec un délai ΔT qui représente l'intervalle entre la fin et le début d'une autre tâche.

	N° domaine de données	Temps de réplication	Nombre d'instance de service de segmentation	Temps CPU	Temps de récupération de résultats
Machine 1 (lance la demande de segmentation)	2, 6	13.70 min	2	24.16 min	0 min
Machine 2 (contient la source de données (1000 images))	1, 5, 9, 10	0,02 min	4	41.97 min	1.93 min
Machine 3	3, 7	14.97 min	2	23.30 min	1.21 min
Machine 4	4, 8	12.33 min	2	26.67 min	1.06 min
Total	10 domaines	41,02 min	10	116.1 min	4.2 min

Tableau 3: résultats détaillés des principales opérations de segmentation par machine.

Si on compare le temps global nécessaire pour l'achèvement de la segmentation (50 min en incluant le temps de réplication des données et le temps de transfert des résultats), avec le

temps de segmentation sans inclure le temps de transfert des fichiers (39.98 min à partir de tableau 1), on conclut que la gestion des données sur la grille joue un rôle important sur les performances de la grille, ce qui nécessite une implémentation d'une bonne politique pour la réplication des données tout en minimisant leurs duplications inutiles. Avec ou sans réplication de données, on a obtenu un temps inférieur à celui de l'exécution séquentielle, ce qui prouve l'utilité des grilles dans les CBIR.

A partir de la table 3, on conclut que le nœud qui possède les données est celui le plus exploité. La machine 2 a traité quatre domaines avec l'utilisation de quatre instances de service de segmentation, contrairement aux autres machines qui n'ont traité que deux domaines pour chacune.

La déconnexion de l'une des machines durant l'exécution d'une tâche, influence sur le temps global de l'achèvement de l'opération, mais pas pour le résultat. Les domaines qui non pas été traités seront redistribués et occuperont les ressources libérées.

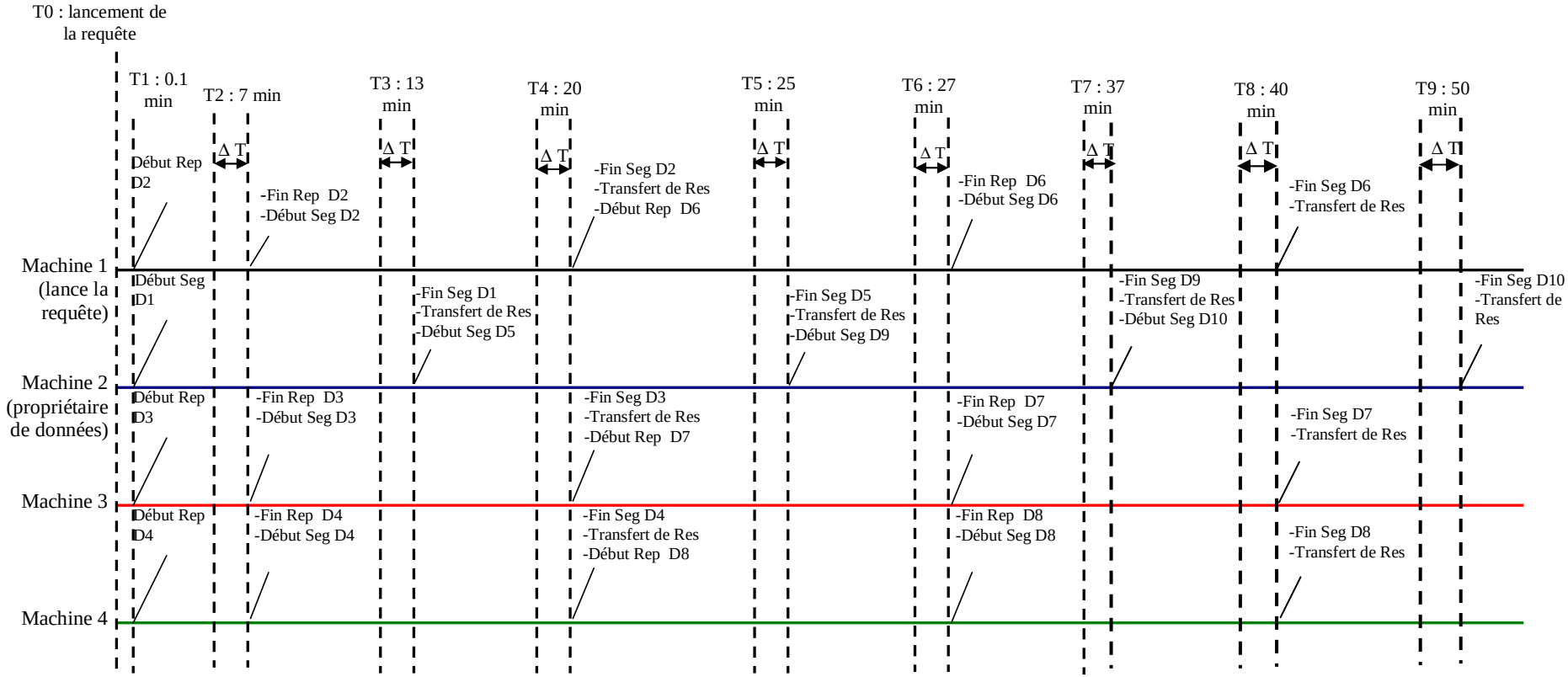


Figure 4-24: Ordre chronologique des étapes de segmentation de 1000 images.

Un troisième scénario d'exécution est réalisé par le relancement plusieurs fois de la segmentation sur les 10.000 images de la même base de Wang, et à chaque fois on change le nœud qui déclanche l'opération. Dans ce scénario, on tient les données répliquées et nous vérifions le temps d'achèvement de l'opération. Pour 6 répétitions, on obtient les résultats illustrés par la figure 4-25.

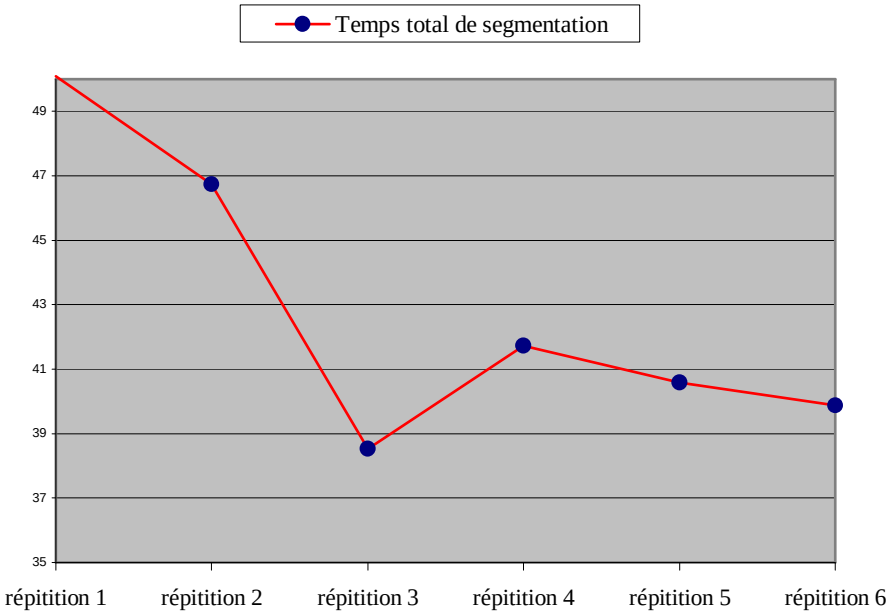


Figure 4-25: stabilisation du temps de segmentation après la troisième répitition.

En comparant le temps global de segmentation illustré par la figure 4-25, on trouve que ce temps se stabilise après la troisième répétition de l'opération. Au début, quand les données n'existent que sur une seule machine, la charge été male équilibrée ce qui influence sur le temps total de l'exécution.

Le temps de segmentation se stabilise après un nombre de répétitions, celui-là est dû à la disponibilité des données nécessaires pour le traitement. Le changement du nœud qui lance la demande de segmentation n'influence pas sur le temps total d'exécution car la charge est équilibrée.

Dans le quatrième scénario, on test le temps nécessaire pour indexer une autre base d'images de Wang créée par le groupe du professeur Wang [89]. Elle représente un sous-ensemble de la base d'images Corel. Elle contient 1000 images naturelles en couleurs. Ces images ont été divisées en 10 classes, chaque classe contient 100 images. L'avantage de cette base est de pouvoir évaluer les résultats. Cette base d'images a été utilisée pour faire des expériences de classification.

Un exemple de chaque classe est présenté par la figure 4-26 (Deselaers, 2003).

Figure 4-26: les dix classe de Wang (Deselaers, 2003).

Dans le but de construire la table des classes, on ne lance le traitement que sur cinquante images par classe, les cinquante autres sont utilisées pour le test durant la classification et l'indexation.

On prend cinquante images aléatoires pour chaque classe, on lance la segmentation et l'extraction des caractéristiques. Les références des fichiers obtenus de l'extraction des caractéristiques visuelles sont sauvegardées dans la table des classes.

Les résultats obtenus de l'exécution séquentielle d'indexation sur les 500 images de test sont illustrés dans la table qui suit.

	Temps CPU
Segmentation	3.67 min
Extraction de caractéristiques des régions optimales (10 régions + toute l'image)	6.40 min
Calcul de distance minimale et classification	3.94 min
Total	14.01 min

Tableau 4: résultats de l'indexation séquentielle sur 50 images.

L'indexation des 50 images de test sur la grille, donne des résultats plus performants comme illustré dans la table qui suit. Dans le cas des données réel, le nombre des images à indexer sera 1000 fois plus grand, et les résultats d'amélioration des performances apparaissent mieux.

Dans notre test, on a répliqué les données sur les nœuds de la grille. Cela nous permet de ne comparer que les temps CPU nécessaires pour l'indexation. Le nombre des images pour chaque domaine est de 10 et l'étape de filtrage n'est pas incluse.

	Temps CPU pour deux machines	Temps CPU pour quatre machines
Segmentation	2.88 min	1.61 min
Extraction de caractéristiques des régions optimales (10 régions + toute l'image)	4.21 min	2.99 min
Calcul de distance minimale et transfert de données pour la classification	3.15 min	2.60 min
Total	10.24 min	7.2 min

Tableau 5: résultats d'indexation de 50 images par les services de la grille.

Les résultats montrés avec ce tableau représentent le temps CPU total pour l'achèvement des opérations intermédiaires telles que la segmentation et l'extraction des caractéristiques.

Le temps total de l'achèvement de l'indexation obtenu par l'exécution sur quatre machines, est près de 50% que celui de l'indexation séquentielle. Ce temps peut aller jusqu'à 30% dans le cas de données plus volumineuses.

L'indexation est effectuée avec un taux d'erreur moyen qui varie de 12% à 27% pour chaque classe, cela est dû principalement au nombre des images limité utilisé pour la construction de la table des classes. Les résultats de la qualité de classification entre dans un autre contexte de recherche, et on a implémenté l'algorithme le plus reconnu dans la segmentation (augmentation des régions) et qui accepte des possibilités de personnalisation par la variation

des conditions de fusion des pixels. Pour extraire les caractéristiques visuelles des régions de l'image, on a utilisé la matrice de co-occurrence et 11 propriétés des 14 mises par Haralik.

4.5 Conclusion

Les grilles de calcul sont un moyen efficace pour exécuter des tâches gourmandes. C'est un assemblage de ressources, elle est performante, tolérante, nécessite une bonne gestion avec des politiques standards et efficaces. En utilisant les puissances des machines disponibles sur la grille, on peut améliorer les performances des plateformes CBIR ou d'autres plateforme de calculs intensifs.

La gestion des services de grille développés est faite avec le service d'information et de synchronisation détaillé précédement, il utilise les services de base de l'Intergitiel Globus, et s'interface avec d'autres services de grille utiles. La gestion des données sur la grille de test installée manque une politique de réplication plus efficace, cette dernière va augmenter d'une façon distinguable les performances du système, dans le cas des données très volumineuses, on a eu un problème de manque d'espace mémoire disque. Et pour cela il faut développer un service de grille qui s'interface avec les registres de réplications fournis par Globus pour estimer l'utilité de quelques données répliquées, et effectuer des nettoyages selon les besoins.

A partir des résultats obtenus et montrés plus haut, on peut dire qu'on a atteint notre objectif, qui est d'augmenter la vitesse de l'achèvement d'une opération dans une plateforme CBIR. Cela nous permet d'améliorer les performances globales du système et ouvre la voie vers d'autres implémentations d'algorithmes de traitement d'images et qui demandent beaucoup de puissance de calcul.

ANNEXE A.

5. ANNEXE A

5.1 Manuel d'installation de l'Intergitiel Globus (GT 4.0.6) sous Linux Fedora core4

Dans cet Annexe, nous allons exposer les principales étapes d'installation de l'Intergitiel Globus dans sa version 4.0.6, sous le système LINUX Fedora 8 core4.

On mentionne que la courante installation peut être personnalisée selon les besoins des utilisateurs et leurs domaines d'application.

Une mise en œuvre d'un nœud de grille de calcul, consiste à préparer d'abord l'environnement, l'initialiser, installer l'Intergitiel, et enfin, configurer ses services.

Dans notre cadre d'étude, nous allons mettre en place un réseau LAN constitué de quatre machines dotées d'un système d'exploitation Linux Fedora core 4.

Pourquoi Linux: Linux est le système d'exploitation le plus approprié pour une installation complète de Globus, ainsi, il offre un environnement très fiable et sécurisé.

5.1.1 Quelques commandes Linux (Pré-requis):

La plupart des commandes à présenter nécessitent d'être en mode administrateur (root) pour être exécutées. La casse utilisée doit être respectée.

man: pour accéder à la page du manuel. Exemple «man cp»

adduser: ajouter un nouveau utilisateur pour le système. Exemple «adduser nouveau_nom_utilisateur»

su: chargement d'un utilisateur. Exemple «su nom_utilisateur»

chown: Désigner l'utilisateur et le groupe propriétaire des fichiers. Exemple «chown group:user file»

L'option -R «chown -R group:user file» permet de rendre l'utilisateur 'user' propriétaire de tous les fichiers inclus dans file, R indique la récursivité.

chmod: permet de spécifier les droits des utilisateurs sur un fichier. Exemple «chmod 777 nom_fichier», c'est pour donner tous les droits de lecture et d'écriture sur le fichier pour tous les utilisateurs.

rm: pour la suppression des fichiers. Exemple «rm nom_fichier»

mv: pour le déplacement des fichiers. Dans la même ligne, on peut donner un nouveau nom au fichier déplacé. Exemple «mv nom_fichier nouveau_emplacement»

cp: pour copier des fichiers. Exemple «cp nom_fichier nouveau_emplacement»

mkdir: créer un répertoire. Exemple «mkdir nouveau_rep»

cd: pour se positionner à un endroit précis de l'arborescence d'un media. Exemple «cd /home/» «cd ../» «cd /».

tar: pour décompresser les fichier de type .tar.gz. Exemple «tar -xz nom_fichier.tar.gz» extraire le fichier en utilisant le filtre gzip.

sh: interpréteur de langage de commande qui lance des commandes à partir d'un fichier .sh ou à partir d'une entrée standard. Exemple «sh nom_fichier.sh»

rpm: une commande pour l'installation de nouveaux paquetages rmp. Exemple «rpm -i nom_paquetage.rpm» l'option -i est suffisante pour faire l'installation du paquetage.

Installation d'un programme

configure (./configure): Cette commande construit un fichier `Makefile utile pour la compilation d'un programme.`

make: permet la construction des programme. Elle recherche la première cible dans le fichier `Makefile` et obéit aux instructions indiquées. Le résultat final attendu est la construction d'un fichier exécutable.

make install: invoque à nouveau **make**, qui recherche la cible `install` dans le `Makefile` et suit les instructions pour installer le programme.

L'éditeur vi : vi est un éditeur pour Linux, il peut être à l'un des deux modes suivants : commande ou édition. Le passage du premier au second se fait par la touche i (insert), le passage inverse se fait par la touche esc. On peut éditer en mode édition et enregistrer nos modifications en mode commande par (':wq' pour écrire dans le fichier et sortir) ou par (' :q' pour quitter sans enregistrer les modification). Afin de pouvoir modifier un fichier on doit avoir l'autorisation en écriture.

Exemple « vi /fichier ».

Notamment, ils existent d'autres éditeurs comme vim et nano.

5.1.2 Mise en place du réseau :

A. Installation du système Linux

La version Linux utilisée est Fedora 8 Core version 4, en faisant attention aux points suivants:

i. Type d'installation poste de travail (WorkStation).
ii. Désactivation de pare-feu pour ne pas gêner le fonctionnement des services de l'Intergitiel.
iii. La date et l'heure doivent être réglées pour chaque machine, cette condition est indispensable dans la phase de signature du certificat pour les autres machines, cela aide à la vérification de la validité du proxy.

B. Configuration du réseau

Lors de l'installation de Linux, nous avons attribué un nom et une adresse IP à chaque noeud de la grille. Chaque nom de l'hôte doit avoir la forme suivante:'nom_machine.nom_domaine' (une exigence de l'Intergitiel Globus toolkit).

La grille de test est composée de quatre machines:

Nom de la machine	Adresse IP	Description
Poste.lri.net	192.168.0.5	Machine client/serveur
Poste2.lri.net	192.168.0.103	Machine client/serveur
Poste3.lri.net	192.168.0.104	Machine client/serveur
Poste4.lri.net	192.168.0.101	Machine client/serveur propriétaire du certificat

Tableau A-1 : éléments de la grille.

C. Outils nécessaires

i. Apache Ant : c'est un exécuteur de tâches, il permet le déploiement des services.
ii. JDK : nécessaire pour la compilation du code java de l'intergitiel.
iii. PostgresSQL : système de gestion de base de données relationnelle, il fonctionne sur des systèmes d'exploitations UNIX. Il est composé de deux parties :

- Partie serveur : c'est la partie fonctionnant sur la machine hébergeant la base de données, elle répond aux requêtes des clients.
- Partie client : cette partie est installée sur les postes client. Les clients interrogent le serveur de base de données par des requêtes SQL.

5.1.3 Préparation de l'installation de globus toolkit 4.0.6

A. Création des comptes utilisateurs

Afin d'installer Globus et pouvoir tester son bon fonctionnement, nous devons créer trois types d'utilisateurs: l'administrateur du système (root), il permet le lancement et l'arrêt du container. Un second utilisateur 'globus' non privilégié par apport à root, cet utilisateur servira à l'installation de l'Intergitiel Globus. Le dernier utilisateur créé est un simple utilisateur 'user' utilisé uniquement pour des raisons de test.

B. Création des répertoires d'installation

Pour chaque nœud, Nous créons deux répertoires sous le chemin *'/usr/local'*, un pour l'installation du Globus Toolkit, et l'autre pour les outils naicéssaires (jdk, ant...).

Création d'un répertoire d'installation globus-4.0.6 sous *'/usr/local'*

```
[root@poste4]# mkdir /usr/local/globus-4.0.6
[root@poste4]# chown globus:globus /usr/local/globus-4.0.6
```

De la même façon, on crée le répertoire d'installation des outils '*outils*' sous *'/usr/local'*,.

```
[root@poste4]# mkdir /usr/local/outils
```

Dans le répertoire 'outils' crée, on copie les outils : Java, Apache-ant et Postgresql.

C. Installation des outils apache Java, Apache-ant et Postgresql

Installation de Java

Après le téléchargement du jdk-1_5_0_14, nous lançons son installation dans le répertoire '*outils*' par la commande d'installation suivante :

```
[root@poste4]# cd /usr/local/outils
[root@poste4 outils]# ./jdk-1_5_0_14-nb-6_0-Linux.sh
```

Après l'installation, nous devons ajouter à la variable d'environnement 'path' le chemin de jdk. Nous pouvons procéder de deux manières distinctes : dans la première nous modifions le fichier */etc/profile.d* et dans la seconde, nous créons un fichier jdk.sh sous */etc/profile.d* dont le contenu est le suivant:

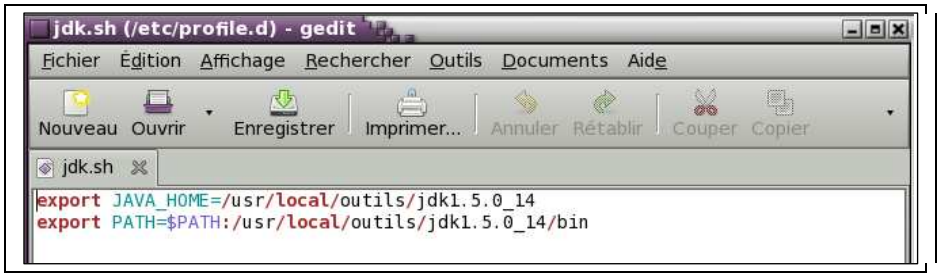


Figure 5-1: Une vue du fichier /etc/profile.d/jdk.sh

Remarque : Il faut supprimer le lien qui pointe vers l'ancienne version de Java représentée par le fichier raccourci 'java', il est localisé sous le chemin */usr/bin.* Nous pouvons tester la réussite de l'installation du JDK en tapant la commande :

```
[root@poste4]# java -version
```

Le résultat est :

```
java version "1.5.0_14"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_14-b03)
Java HotSpot(TM) Client VM (build 1.5.0_14-b03, mixed mode, sharing)
```

Installation de Apache ant

Apache ant est un exécuteur de tâches permettant la compilation et le déploiement des programmes. Après le 'téléchargement de la version de *Apache ant* '*appache-ant-1.7.0-bin.tar.gz'*, on l'installe dans le répertoire *'/usr/local/outils'*.

L'installation de *Apache ant* consiste à le décompresser, et ajouter à la variable d'environnement 'path' le chemin de *ant* en créant un fichier *ant.sh* dans '*/etc/profile.d*'.

La décompression se fait par :

```
[root@poste4 outils]# tar -xzf appache-ant-1.7.0-bin.tar.gz
```

Un aperçu du fichier 'ant.sh' est illustré par la figure suivante:

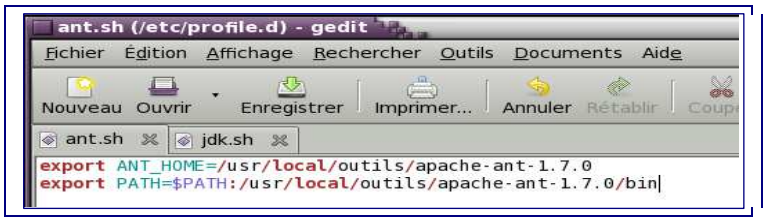


Figure 5-2: Une vue du fichier /etc/profile.d/ant.sh

Pour tester le bon fonctionnement de ant, on lance la commande:

```
[root@poste4 ~]# ant – version
```

Le résultat sera comme suit :

```
Apache Ant version 1.7.0 compiled on July 13 2008
```

Installation de postgresql :

On télécharge le fichier *postgresql-8.2.6.tar.gz*, et on le décompresse dans *'/usr/local/'* :

```
[root@poste4 outils]# tar -xzf postgresql-8.2.6.tar.gz -C /usr/local/
```

On lance la commande suivante:

```
[root@poste4 postgresql-8.2.6]# ./configure -without-readline -without-zlib
```

On lance l'installation:

```
[root@poste4 postgresql-8.2.6] # make
[root@poste4 postgresql-8.2.6] # make install
```

Ensuite, nous créons un utilisateur postgres et nous créons le répertoire *'pgsql/data'*.

```
[root@poste4] # adduser postgres
[root@poste4] # mkdir /usr/local/pgsql/data
[root@poste4] # chown postgres /usr/local/pgsql/data
```

En tant qu'utilisateur 'postgres', on exécute la commande suivante :

```
[postgres@poste4] $/usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data | tee initdb.log
```

On lance le serveur de base de données :

```
[postgres@poste4]$ cd /usr/local
[postgres@poste4 local]$ pgsql/bin/postmaster -D pgsql/data
```

Le résultat sera:

```
LOG:  database system was interrupted at 2008-07-13 13:09:54 CET
LOG:  checkpoint record is at 0/42E904
LOG:  redo record is at 0/42E904; undo record is at 0/0; shutdown FALSE
LOG:  next transaction ID: 0/622; next OID: 24579
LOG:  next MultiXactId: 1; next MultiXactOffset: 0
LOG:  database system was not properly shut down; automatic recovery in progress
LOG:  record with zero length at 0/42E94C
LOG:  redo is not required
LOG:  database system is ready
```

Pour vérifier si le serveur de la base de données est bien installée, nous créons une base de données de *test* comme suit :

```
[postgres@poste4]$ /usr/local/pgsql/bin/createdb test
```

Le résultat de cette commande est :

```
CREATE DATABASE
```

Nous testons la création de la base par cette commande :

```
[postgres@poste4]$  /usr/local/pgsql/bin/psql test
```

Le résultat est:

```
Welcome to psql 8.2.6, the PostgreSQL interactive terminal.
Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit
test=#
```

5.1.4 L'installation de Globus Toolkit

A. Lancement du script d'installation

La version téléchargée de Globus Toolkit est ‘gt4.0.6-x86_fc_4-installer.tar.gz’. Nous commençons par la décompression du fichier dans *‘/usr/local/globus-4.0.6’*. On copie les fichiers décompressés dans le même répertoire. On ajoute la variable $GLOBUS_LOCATION au path, et ensuite, on lance l'opération d'installation :

- Décompresser et copier le zip :

```
[root@poste4 ~]# tar -xzf /usr/local/gt4.0.6-x86_fc_4-installer.tar.gz -C /usr/local/globus-4.0.6/
[root@poste4 ~]# cp -r /usr/local/globus-4.0.6/gt4.0.6-x86_fc_4-installer/* /usr/local/globus-4.0.6/
```

- Désigner l'utilisateur et le groupe propriétaire des fichiers : cette étape permet d’affecter à l’utilisateur globus l’ensemble des fichiers décompressés, ils sont utiles pour l’installation de l’Intergitiel. Avec la commande qui suit, l’installation peut être faite sous l’utilisateur globus:

```
[root@poste4 ~]# chown -R globus:globus /usr/local/globus-4.0.6/
```

L’installation de Globus Toolkit doit être faite sous l'utilisateur globus. Nous créons un fichier `Makefile` avec la commande *./configure :*

```
[globus@poste4 ~]$ cd /usr/local/globus-4.0.6
[globus@poste4 globus-4.0.6]$ ./configure -prefix=$GLOBUS_LOCATION -enable-prewsmd -enable-drs
```

Le résultat sera :

```
checking for javac... /usr/local/outils/jdk1.5.0_14/bin/javac
checking for ant... /usr/local/outils/apache-ant-1.7.0/bin/ant
configure: creating ./config.status
config.status: creating Makefile
```

Les caractéristiques suivantes sont optionnelles et sont toutes désactivées par défaut.

-enable-prewsmd : permet de construire la base de pre-webservices mds.

-enable-wsgram-condor : permet de construire l’interface de GRAM Condor scheduler.

-enable-wsgram-lsf: permet de construire l’interface de GRAM LSF scheduler.

-enable-i18n: permet d’activer l’internationalisation.

-enable-drs: permet d’activer le Service de Réplication de données

- Construire le programme exécutable avec la commande make : nous utilisons *tee* pour garder la trace de l’exécution des commandes d’installation, cela nous aide à localiser les erreurs qui peuvent se produire durant l’installation :

```
[globus@poste4 globus-4.0.6 ]$ make |tee build.log
```

Une partie du résultat de cette commande sera comme suit :

```
tar -C /usr/local/globus-4.0.6 -xzf binary-trees/globus_wsrf_replicator_common_schema-*/*.tar.gz
```

```
tar -C /usr/local/globus-4.0.6 -xzf binary-trees/globus_wsrf_replicator_common_java-*/*.tar.gz
tar -C /usr/local/globus-4.0.6 -xzf binary-trees/globus_wsrf_replicator_service_java-*/*.tar.gz
tar -C /usr/local/globus-4.0.6 -xzf binary-trees/globus_wsrf_replicator_client_java-*/*.tar.gz
echo "Your build completed successfully.  Please run make install."
Your build completed successfully.  Please run make install.
```

Lancer l'installation de Globus avec la commande *make install* :

```
[globus@poste4 globus-4.0.6 ]$ make install|tee install.log
```

Une partie du résultat est la suivante :

```
running /usr/local/globus-4.0.6/setup/globus/setup-globus-job-manager-fork..[ Changing to /usr/local/globus-
4.0.6/setup/globus ]
find-fork-tools: WARNING: "Cannot locate mpiexec"
find-fork-tools: WARNING: "Cannot locate mpirun"
checking for mpiexec... no
checking for mpirun... no
find-fork-tools: creating ./config.status
config.status: creating fork.pm
..Done
```

B. Installation de l'Autorité de Certification (CA):

L'installation de l'autorité de certification est faite sur une seule machine qu'on appel serveur de certificat. Cette étape est très critique pour le reste de l'installation et de la configuration des différents services de l'Intergitiel.

Le script d'installation de l'AC est créé lors de l'étape précédente, l'installation du serveur du certificat doit être faite sous l'utilisateur globus.

Exécution du script d'installation:

Nous rajoutons au fichier «hosts» qui se trouve dans le répertoire ***'/etc/'*** le nom de la machine et l'adresse IP correspondante de la façon suivante:

192.168.0.101 poste4.lri.net poste4

Sous root, nous créons le répertoire *'/etc/grid-security'* qui va contenir le certificat du hôte. Après la création, nous affectons le répertoire à l'utilisateur globus :

```
[root@poste4]# mkdir /etc/grid-security
[root@poste4]# chown globus:globus /etc/grid-security
```

Sous globus, nous lançons l'exécution du script :

```
[globus@poste4]$ $GLOBUS_LOCATION/setup/globus/setup-simple-ca
```

Nous confirmons l'exécution en tapant 'y', puis nous remplissons les champs demandés durant l'installation (Email, durée d'expiration du certificat, mot de passe). Une partie du résultat est la suivante:

```
WARNING: GPT_LOCATION not set, assuming:
      GPT_LOCATION=/usr/local/globus-4.0.6
```

```
   C e r t i f i c a t e   A u t h o r i t y   S e t u p
This script will setup a Certificate Authority for signing Globus
users certificates.  It will also generate a simple CA package
that can be distributed to the users of the CA.
The CA information about the certificates it distributes will
be kept in:
/home/globus/.globus/simpleCA/
The unique subject name for this CA is:
cn=Globus Simple CA, ou=simpleCA-poste4.lri.net, ou=GlobusTest, o=Grid
Do you want to keep this as the CA subject (y/n) [y]:y
Enter the email of the CA (this is the email where certificate
requests will be sent to be signed by the CA):ahmed.dib@lri-annaba.net
The CA certificate has an expiration date. Keep in mind that
once the CA certificate has expired, all the certificates
signed by that CA become invalid.  A CA should regenerate
the CA certificate and start re-issuing ca-setup packages
before the actual CA certificate expires.  This can be done
by re-running this setup script.  Enter the number of DAYS
the CA certificate should last before it expires.
[default: 5 years (1825 days)]:1825
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
creating CA config package...done.
A self-signed certificate has been generated
for the Certificate Authority with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
If this is invalid, rerun this script
/usr/local/globus-4.0.6/setup/globus/setup-simple-ca
and enter the appropriate fields.
-----------------------------------------------------------------
The private key of the CA is stored in /home/globus/.globus/simpleCA//private/cakey.pem
The public CA certificate is stored in /home/globus/.globus/simpleCA//cacert.pem
The distribution package built for this CA is stored in
/home/globus/.globus/simpleCA//globus_simple_ca_637244ab_setup-0.19.tar.gz
This file must be distributed to any host wishing to request
certificates from this CA.
CA setup complete.
The following commands will now be run to setup the security
configuration files for this CA:
$GLOBUS_LOCATION/sbin/gpt-build /home/globus/.globus/simpleCA//globus_simple_ca_637244ab_setup-
0.19.tar.gz
$GLOBUS_LOCATION/sbin/gpt-postinstall
-----------------------------------------------------------------
setup-ssl-utils: Configuring ssl-utils package
Running setup-ssl-utils-sh-scripts...
************************************************************************
Note: To complete setup of the GSI software you need to run the
following script as root to configure your security configuration
directory:
/usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/setup-gsi
For further information on using the setup-gsi script, use the -help
option.  The -default option sets this security configuration to be
the default, and -nonroot can be used on systems where root access is
not available.
************************************************************************
setup-ssl-utils: Complete
```

Installation de GSI:

Pour lancer l'installation de GSI on tape la commande suivante :

```
[root@poste4]# /usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/setup-gsi
```

Le résultat est le suivant :

```
setup-gsi: Configuring GSI security
Making trusted certs directory: /etc/grid-security/certificates/
mkdir /etc/grid-security/certificates/
Installing /etc/grid-security/certificates//grid-security.conf.637244ab...
Running grid-security-config...
Installing Globus CA certificate into trusted CA certificate directory...
Installing Globus CA signing policy into trusted CA certificate directory...
setup-gsi: Complete
```

```
[globus@poste4 globus-4.0.6]$ grid-cert-request -ca
nondefaultca=true
The available CA configurations installed on this host are:
1)  637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
Enter the index number of the CA you want to sign your cert request: 1
Using CA: 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
A certificate request and private key is being created.
You will be asked to enter a PEM pass phrase.
This pass phrase is akin to your account password,
and is used to protect your key file.
If you forget your pass phrase, you will need to
obtain a new certificate.
Generating a 1024 bit RSA private key
......++++++
.....................................................++++++
writing new private key to '/home/globus/.globus/userkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-
poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:
A private key and a certificate request has been generated with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus
If the CN=globus is not appropriate, rerun this
script with the -force -cn "Common Name" options.
Your private key is stored in /home/globus/.globus/userkey.pem
Your request is stored in /home/globus/.globus/usercert_request.pem
Please e-mail the request to the Globus Simple CA ahmed.dib@lri-annaba.net
You may use a command similar to the following:
  cat /home/globus/.globus/usercert_request.pem | mail ahmed.dib@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at ahmed.dib@lri-annaba.net
```

C. Génération du certificat pour le hôte:

- Demande certificat pour un nœud *'host certificates'*

Cette demande se fait sous root par la commande suivante:

```
[root@poste4]# grid-cert-request -host poste4.lri.net
```

L'exécution de cette commande nous donne comme résultat trois fichiers en sortie sous '*/etc/grid-security*', dans les résultats, nous trouvons que le fichier '*hostcert.pem*' qui va porter le certificat valide, est de taille zéro, en faite, ce fichier va contenir le résultat de signature du fichier portant la demande de certificat '*hostcert_request.pem*' :

```
-rw-r--r-- 1 root   root       0    jui 20 22:46 hostcert.pem
-rw-r--r-- 1 root   root   1396    jui 20 22:46 hostcert_request.pem
-r-------- 1   root   root   887     jui 20 22:46 hostkey.pem
```

- Signature du certificat «host certificates»

La signature du certificat se fait par l'utilisateur globus, mais le fichier à signer *hostcert_request.pem* appartient à root, nous devons faire attention aux différents droits des utilisateurs sur les fichiers. Nous procédons à la signature du certificat de l'hôte :

```
[root@poste4~]# cp /etc/grid-security/hostcert_request.pem /tmp/
[globus@poste4~]$ grid-ca-sign -in /tmp/hostcert_request.pem -out /tmp/hostsigend.pem
```

Le résultat sera :

```
To sign the request
please enter the password for the CA key:
The new signed certificate is at: /home/globus/.globus/simpleCA//newcerts/01.pem
```

Nous déplaçons le fichier contenant la clé signée '*hostsigned.pem*' vers le fichier qui va contenir le certificat signé de l'hôte '*hostcert.pem*':

```
[root@poste4 .globus]# cp /tmp/hostsigend.pem /etc/grid-security/
[root@poste4 .globus]# mv /etc/grid-security/hostsigend.pem /etc/grid-security/hostcert.pem
```

Le résultat est:

```
mv: écraser `/etc/grid-security/hostcert.pem'?y
```

Nous vérifions le contenu de '/etc/grid-security'

```
-rw-r--r-- 1 root   root   2682 jui 20 22:55 hostcert.pem
-rw-r--r-- 1 root   root   1407 jui 20 22:46 hostcert_request.pem
-r-------- 1 root   root    891 jui 20 22:46 hostkey.pem
```

D. Génération du certificat pour l'utilisateur globus:

iv. Demande du certificat pour l'utilisateur globus avec la commande suivante:

```
[globus@poste4 ]$ grid-cert-request -ca
```

Le résultat de cette commande est comme suit :

```
nondefaultca=true
The available CA configurations installed on this host are:
1)  637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
Enter the index number of the CA you want to sign your cert request: 1
Using CA: 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
A certificate request and private key is being created.
You will be asked to enter a PEM pass phrase.
This pass phrase is akin to your account password,
```

```
and is used to protect your key file.
If you forget your pass phrase, you will need to
obtain a new certificate.
Generating a 1024 bit RSA private key
......++++++
.....................................................++++++
writing new private key to '/home/globus/.globus/userkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-
poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:
A private key and a certificate request has been generated with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus
If the CN=globus is not appropriate, rerun this
script with the -force -cn "Common Name" options.
Your private key is stored in /home/globus/.globus/userkey.pem
Your request is stored in /home/globus/.globus/usercert_request.pem
Please e-mail the request to the Globus Simple CA ahmed.dib@lri-annaba.net
You may use a command similar to the following:
  cat /home/globus/.globus/usercert_request.pem | mail ahmed.dib@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at ahmed.dib@lri-annaba.net
```

Cette commande nous génère trois fichiers en sortie sous ‘/home/globus/.globus/’:

```
userkey.pem     usercert_request.pem       usercert.pem
```

- Signature du certificat de l'utilisateur globus:

Après génération du fichier ‘usercert_request.pem’, nous devons le signer avec la commande suivante.

```
[globus@poste4 globus-4.0.6]$ grid-ca-sign -in /home/globus/.globus/usercert_request.pem -out
/home/globus/.globus/usercert.pem
```

Le résultat est:

```
To sign the request
please enter the password for the CA key:
The new signed certificate is at: /home/globus/.globus/simpleCA//newcerts/02.pem
```

Nous devons vérifier que les fichiers du certificat de globus ont les droits appropriés :

```
-rw-r--r-- 1 globus globus 2689 jui 20 22:52 usercert.pem
-rw-r--r-- 1 globus globus 1415 jui 20 22:47 usercert_request.pem
-r-------- 1 globus globus  963 jui 20 22:47 userkey.pem
```

E. Génération du certificat pour l'utilisateur user:

v. Demande du certificat pour l'utilisateur ‘user’:

Partant du même principe, nous lançons la commande suivante sous l'utilisateur 'user':

```
[user@poste4 ]$ grid-cert-request -ca -force
```

Le résultat sera :

```
nondefaultca=true
The available CA configurations installed on this host are:
1)  637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
Enter the index number of the CA you want to sign your cert request: 1
Using CA: 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
 /home/user/.globus/usercert_request.pem already exists
 /home/user/.globus/usercert.pem already exists/bin/chmod: modification des permissions de
`/home/user/.globus/usercert.pem': Opération non permise
  /home/user/.globus/userkey.pem already exists
A certificate request and private key is being created.
You will be asked to enter a PEM pass phrase.
This pass phrase is akin to your account password,
and is used to protect your key file.
If you forget your pass phrase, you will need to
obtain a new certificate.
Generating a 1024 bit RSA private key
.....++++++
......++++++
writing new private key to '/home/user/.globus/userkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-
poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:

A private key and a certificate request has been generated with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=user
If the CN=user is not appropriate, rerun this
script with the -force -cn "Common Name" options.
Your private key is stored in /home/user/.globus/userkey.pem
Your request is stored in /home/user/.globus/usercert_request.pem
Please e-mail the request to the Globus Simple CA ahmed.dib@lri-annaba.net
You may use a command similar to the following:
cat /home/user/.globus/usercert_request.pem | mail ahmed.dib@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at ahmed.dib@lri-annaba.net
```

- Signature du certificat de l'utilisateur user:

Nous signons le fichier *'usercert_request.pem'* avec la commande qui suit lancée sous l'utilisateur globus.

```
[user@poste4 ~]$ cp /home/user/.globus/usercert_request.pem /tmp/user_usercertrequest.pem
[globus@poste4 ~]$ grid-ca-sign -in /tmp/user_usercertrequest.pem -out /tmp/usersigned2.pem
[user@poste4 ~]$ cp /tmp/usersigned2.pem /home/user/.globus/
```

```
[user@poste4 ~]$ mv /home/user/.globus/usersigned2.pem /home/user/.globus/usercert.pem
[root@poste4 ~]# chmod 644 /home/user/.globus/usercert.pem
```

Nous vérifions le contenu de *'/home/user/.globus/'* :

```
-rw-r--r-- 1 user user 2683 jui 20 23:04 usercert.pem
-rw-r--r-- 1 user user 1407 jui 20 22:48 usercert_request.pem
-r-------- 1 user user  963 jui 20 22:48 userkey.pem
```

F. Création du certificat du 'container':

Le certificat du container est pratiquement nécessaire pour le lancement des services web. Nous procédons comme suit :

```
[root@poste4]# cp /etc/grid-security/hostcert.pem /etc/grid-security/containercert.pem
[root@poste4]# cp /etc/grid-security/hostkey.pem /etc/grid-security/containerkey.pem
[root@poste4]# chown globus:globus /etc/grid-security/container*.pem
```

G. Ajout des autorisations:

- Ajout des autorisations pour l'utilisateur globus :

L'ajout des autorisations se fait par la création du fichier *'grid-mapfile'* qui garantie une communication sécurisée entre les différents noeuds de la grille, sa création peut être faite en utilisant l'éditeur vi, sous le répertoire grid-security. Pour cela, nous lançons la commande:

```
[root@ poste4]# vi /etc/grid-security/grid-mapfile
```

Nous ajoutons dans le fichier créé une entrée correspondante à chaque utilisateur (globus et user), les entrées à ajouter sont composées du sujet et du propriétaire du certificat. Pour récupérer ces derniers, pour chaque utilisateur, nous procédons comme suit:

```
[globus@poste4]$ grid-cert-info –subject
```

```
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus
```

On lance la commande :

```
[globus@poste4]$ whoami
```

Le résultat est :

```
globus
```

Sous root, nous ajoutons une entrée correspondante au certificat de l'utilisateur globus dans le fichier grid-mapfile :

```
[root@poste4]# /usr/local/globus-4.0.6/sbin/grid-mapfile-add-entry -dn
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus" -ln globus
```

Le résultat sera :

```
Modifying /etc/grid-security/grid-mapfile ...
New entry:
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus" globus
(1) entry added
```

- Ajout des autorisations pour l'utilisateur user :

Nous procédons de la même manière. Pour récupérer le sujet du certificat et son propriétaire, nous exécutons :

```
[user@poste4 ~]$ grid-cert-info -subject
```

Le résultat est :

```
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=user
```

Après, nous lançons cette commande:

```
[user@poste4 ~]$ whoami
```

Résultat:

```
user
```

Sous root, nous ajoutons une entrée correspondante au certificat de l'utilisateur user dans le fichier grid-mapfile :

```
[root@poste4 ~]# /usr/local/globus-4.0.6/sbin/grid-mapfile-add-entry -dn "
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=user" -ln user
```

Résultat:

```
Modifying /etc/grid-security/grid-mapfile ...
New entry:
" /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=user" user
(1) entry added
```

H. Vérification des certificats des utilisateurs

Pour la vérification du certificat de l'utilisateur globus, on lance le proxy avec la commande suivante:

```
[globus@ poste4/]$ grid-proxy-init -debug -verify
```

Le résultat est le suivant:

```
User Cert File: /home/globus/.globus/usercert.pem
User Key File: /home/globus/.globus/userkey.pem
Trusted CA Cert Dir: /etc/grid-security/certificates
Output File: /tmp/x509up_u500
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus
Enter GRID pass phrase for this identity:
Creating proxy .....++++++++++++
.....++++++++++++
 Done
Proxy Verify OK
Your proxy is valid until: Mon Jul 21 11:12:10 2008
```

De même, sous l'utilisateur user :

```
[user@ poste4/]$ grid-proxy-init -debug –verify
```

Le résultat est

```
User Cert File: /home/user/.globus/usercert.pem
User Key File: /home/user/.globus/userkey.pem
```

```
Trusted CA Cert Dir: /etc/grid-security/certificates
Output File: /tmp/x509up_u501
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=user
Enter GRID pass phrase for this identity:
Creating proxy .....++++++++++++
...++++++++++++
 Done
Proxy Verify OK
Your proxy is valid until: Mon Jul 21 11:12:19 2008
```

5.1.5 Installation de certificat pour plusieurs machines

Le certificat sera installé sur les machines autres que celle utilisée pour l'installation de certificat d'autorité.

A partir de la machine 4, nous copions le fichier 'globus_simple_ca_637244ab_setup-0.19.tar.gz.' qui se trouve sous *«/home/globus/.globus/simpleCA»*, on le copie sur les machines sur lesquelles nous allons installer le certificat sous *«home/globus/»*, puis nous lançons la commande suivante sous globus. Nous prenons comme exemple l'installation du certificat sur la machine 'poste' :

```
[root@poste4 sbin]# scp /home/globus/.globus/simpleCA/globus_simple_ca_637244ab_setup-0.19.tar.gz
root@poste.lri.net/home/globus/
[globus@poste ~]$ $GLOBUS_LOCATION/sbin/gpt-build globus_simple_ca_637244ab_setup-0.19.tar.gz
gcc32dbg
```

Nous obtenons:

```
gpt-build ====> Changing to /home/globus/BUILD/globus_core-4.30/
gpt-build ====> BUILDING FLAVOR gcc32dbg
gpt-build ====> Changing to /home/globus/BUILD
gpt-build ====> REMOVING empty package globus_core-gcc32dbg-pgm_static
gpt-build ====> REMOVING empty package globus_core-noflavor-doc
gpt-build ====> CHECKING BUILD DEPENDENCIES FOR globus_simple_ca_637244ab_setup
gpt-build ====> Changing to /home/globus/BUILD/globus_simple_ca_637244ab_setup-0.19/
gpt-build ====> BUILDING globus_simple_ca_637244ab_setup
gpt-build ====> Changing to /home/globus/BUILD
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-data
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-dev
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-doc
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-pgm_static
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-rtl
```

Puis la commande:

```
[globus@poste]$ $GLOBUS_LOCATION/sbin/gpt-postinstall
```

Le résultat sera:

```
running /usr/local/globus-4.0.6//setup/globus/./setup-ssl-utils.637244ab..[ Changing to /usr/local/globus-
4.0.6/setup/globus/. ]
setup-ssl-utils: Configuring ssl-utils package
Running setup-ssl-utils-sh-scripts...
************************************************************************
Note: To complete setup of the GSI software you need to run the
following script as root to configure your security configuration
directory:
/usr/local/globus-4.0.6//setup/globus_simple_ca_637244ab_setup/setup-gsi
```

```
For further information on using the setup-gsi script, use the -help
option.  The -default option sets this security configuration to be
the default, and -nonroot can be used on systems where root access is
not available.
********************************************************************************
setup-ssl-utils: Complete
..Done
WARNING: The following packages were not set up correctly:
       globus_simple_ca_637244ab_setup-noflavor-pgm
Check the package documentation or run postinstall -verbose to see what happened
```

Maintenant nous installons le GSI par la commande suivante:

```
[root@poste]#$GLOBUS_LOCATION/setup/globus_simple_ca_637244ab_setup/setup-gsi -default
```

Le résultat :

```
bash: /usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/etup-gsi: Aucun fichier ou répertoire de
ce type
[root@poste ~]# /usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/setup-gsi -default
setup-gsi: Configuring GSI security
Making /etc/grid-security...
mkdir /etc/grid-security
Making trusted certs directory: /etc/grid-security/certificates/
mkdir /etc/grid-security/certificates/
Installing /etc/grid-security/certificates//grid-security.conf.637244ab...
Running grid-security-config...
Installing Globus CA certificate into trusted CA certificate directory...
Installing Globus CA signing policy into trusted CA certificate directory...
setup-gsi: Complete
```

Nous ajoutons au fichier *'/etc/hosts'* les adresses IP, ainsi que les noms des autres hôtes reliés par la grille. Nous ouvrons le fichier avec l'éditeur vi :

```
[root@poste]# vi /etc/hosts
```

Et nous ajoutons les lignes :

192.168.0.103 poste2.lri.net

192.168.0.104 poste3.lri.net

192.168.0.101 poste4.lri.net

Nous lançons la demande de certificat pour l'hôte 'poste' :

```
[root@poste ~]# grid-cert-request -host poste.lri.net
```

Le résultat :

```
Generating a 1024 bit RSA private key
.........++++++
.......++++++
writing new private key to '/etc/grid-security/hostkey.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
```

```
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-poste4.lri.net]:Name (e.g., John M. Smith) []:
 A private host key and a certificate request has been generated
with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=host/poste.lri.net
---------------------------------------------------------
The private key is stored in /etc/grid-security/hostkey.pem
The request is stored in /etc/grid-security/hostcert_request.pem
Please e-mail the request to the Globus Simple CA ahmed.dib@lri-annaba.net
You may use a command similar to the following:
 cat /etc/grid-security/hostcert_request.pem | mail ahmed.dib@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at ahmed.dib@lri-annaba.net
```

Dans le répertoire /etc/*grid-security*, trois fichiers sont créés :

```
-rw-r--r-- 1 root root    0 jui 21 10:20 hostcert.pem
-rw-r--r-- 1 root root 1406 jui 21 10:20 hostcert_request.pem
-r-------- 1 root root  887 jui 21 10:20 hostkey.pem
```

La signature du certificat du hôte, se fait par l'utilisateur globus de la machine 4, donc, nous devons déplacer les deux fichiers résultant *hostcert_request.pem* et *usercert_request.pem,* de la machine ‘poste’ à la machine ‘poste4’ et les signer par l’utilisateur globus :

```
[globus@poste4 ~]$ grid-ca-sign -in /tmp/hostcert_request.pem -out /tmp/hostsigend.pem
```

Le résultat sera :

```
To sign the request
please enter the password for the CA key:
The new signed certificate is at: /home/globus/.globus/simpleCA//newcerts/05.pem
```

Après la signature, nous remplaçons le fichier signé hostsigned.pem par le fichier hostcert.pem de la machine ‘poste’. Nous plaçons hostsigned.pem sous ‘*/tmp/*’ et nous exécutons la commande:

```
[globus@poste]$ mv /tmp/hostsigned.pem /etc/ grid-security/hostcert.pem
```

Il faut s’assurer que les fichiers *hostcert.pem, hoscert_request.pem* et *hostkey* ont respectivement les priorités suivantes : 644, 644 et 400. Une priorité excessive ou manquante peut générer des erreurs de permission.

De même, nous demandons le certificat pour les autres utilisateurs de la machine ‘poste’. On lance la commande suivante sous l’utilisateur globus de la seconde machine :

```
[globus@poste ~]$ grid-cert-request -ca -force -cn globus_poste
```

Résultat:

```
nondefaultca=true
The available CA configurations installed on this host are:
```

```
1)  637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
 Enter the index number of the CA you want to sign your cert request: 1
Using CA: 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
    /home/globus/.globus/usercert_request.pem already exists
    /home/globus/.globus/usercert.pem already exists
    /home/globus/.globus/userkey.pem already exists
A certificate request and private key is being created.
You will be asked to enter a PEM pass phrase.
This pass phrase is akin to your account password,
and is used to protect your key file.
If you forget your pass phrase, you will need to
obtain a new certificate.
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to '/home/globus/.globus/userkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-
poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:
 A private key and a certificate request has been generated with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus_poste
If the CN=globus_poste is not appropriate, rerun this
script with the -force -cn "Common Name" options.
Your private key is stored in /home/globus/.globus/userkey.pem
Your request is stored in /home/globus/.globus/usercert_request.pem
Please e-mail the request to the Globus Simple CA ahmed.dib@lri-annaba.net
You may use a command similar to the following:
  cat /home/globus/.globus/usercert_request.pem | mail ahmed.dib@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at ahmed.dib@lri-annaba.net
```

Les trois fichiers ‘ .pem’ générés sont les suivants :

```
[globus@poste ~]$ ls -l /home/globus/.globus/
```

Le résultat est

```
total 20
-rw-r--r-- 1 globus globus    0 jui 21 10:22 usercert.pem
-rw-r--r-- 1 globus globus 1415 jui 21 10:22 usercert_request.pem
-r-------- 1 globus globus  963 jui 21 10:22 userkey.pem
```

Après la génération du fichier *usercert_request.pem*, nous devons le signer sur la machine 4, nous plaçons *usercert_request.pem* dans *‘/tmp/’* du poste4 et nous lançons la commande :

```
[globus@poste4 ~]$ grid-ca-sign -in /tmp/usercert_request.pem -out /tmp/usersigend.pem
```

Le résultat sera:

```
To sign the request
```

```
please enter the password for the CA key:
The new signed certificate is at: /home/globus/.globus/simpleCA//newcerts/07.pem
```

Nous plaçons le fichier signé *signed.pem* de la machine 4, dans */tmp/* de la machine 'poste' et nous lançons la commande suivante sous globus:

```
[globus@ poste]# mv /tmp/signed.pem /home/globus/.globus/usercert.pem
```

De même, il faut vérifier que les fichiers *usercert.pem, usercert_request.pem* et *userkey.pem* ont respectivement les priorités suivantes : 644, 644 et 400.

Finalement, nous lançons la création du proxy avec la commande :

```
[globus@ poste/]$ grid-proxy-init –debug -verify
```

Résultat:

```
User Cert File: /home/globus/.globus/usercert.pem
User Key File: /home/globus/.globus/userkey.pem
Trusted CA Cert Dir: /etc/grid-security/certificates
Output File: /tmp/x509up_u501
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus_poste
Enter GRID pass phrase for this identity:
Creating proxy ...++++++++++++
..............................++++++++++++
 Done
Proxy Verify OK
Your proxy is valid until: Mon Jul 21 22:50:58 2008
```

On demande le certificat pour le deuxième utilisateur 'saida' :

```
[saida@poste ~]$ grid-cert-request -ca
```

Le résultat:

```
nondefaultca=true
The available CA configurations installed on this host are:
1)  637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
Enter the index number of the CA you want to sign your cert request: 1
Using CA: 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
A certificate request and private key is being created.
You will be asked to enter a PEM pass phrase.
This pass phrase is akin to your account password,
and is used to protect your key file.
If you forget your pass phrase, you will need to
obtain a new certificate.
Generating a 1024 bit RSA private key
.................++++++
..............................++++++
writing new private key to '/home/saida/.globus/userkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
```

```
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-
poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:
A private key and a certificate request has been generated with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=root
If the CN=root is not appropriate, rerun this
script with the -force -cn "Common Name" options.
Your private key is stored in /home/saida/.globus/userkey.pem
Your request is stored in /home/saida/.globus/usercert_request.pem
Please e-mail the request to the Globus Simple CA ahmed.dib@lri-annaba.net
You may use a command similar to the following:
  cat /home/saida/.globus/usercert_request.pem | mail ahmed.dib@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at ahmed.dib@lri-annaba.net
```

Visualisation des trois fichiers '.pem' :

```
[saida@poste ~]$ ls -l /home/saida/.globus/
```

```
total 20
-rw-r--r-- 1 saida saida    0 jui 21 10:25 usercert.pem
-rw-r--r-- 1 saida saida 1408 jui 21 10:25 usercert_request.pem
-r-------- 1 saida saida  963 jui 21 10:25 userkey.pem
```

Après la génération du fichier *hostcert_request.pem*, nous devons le signer sur la machine 4, nous plaçons usercert_request.pem dans *'/tmp/'*

```
[globus@poste ~]$ cp /home/globus/.globus/usercert_request.pem /tmp
```

Et nous lançons la commande de signature:

```
[globus@poste4 ~]$ grid-ca-sign -in /tmp/usercert_request_user.pem -out /tmp/usersigend_user.pem -force
```

Le résultat sera:

```
To sign the request
please enter the password for the CA key:
The new signed certificate is at: /home/globus/.globus/simpleCA//newcerts/06.pem
```

Maintenant nous allons copier le fichier signé *usersigned_user.pem* dans le fichier *usercert.pem* qui est vide en tapant la commande suivante :

```
[saida@poste ~]$ cp /tmp/usersigend_user.pem /home/saida/.globus/usercert.pem
```

Visualisation des propriétés du fichier *usercert.pem :*

```
[saida@poste ~]$ ls -l /home/saida/.globus/usercert.pem
```

Résultat:

```
-rw-r--r-- 1 saida saida 2683 jui 21 10:50 /home/saida/.globus/usercert.pem
```

Finalement, nous lançons la création du proxy en tapant la commande suivante :

```
[saida@poste ~]$ grid-proxy-init -debug -verify
```

Résultat:

```
User Cert File: /home/saida/.globus/usercert.pem
User Key File: /home/saida/.globus/userkey.pem
```

```
Trusted CA Cert Dir: /etc/grid-security/certificates
Output File: /tmp/x509up_u500
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=root
Enter GRID pass phrase for this identity:
Creating proxy .....++++++++++++
......++++++++++++
 Done
Proxy Verify OK
Your proxy is valid until: Mon Jul 21 22:55:09 2008
```

Ajout des autorisations

Comme précédemment, on crée le fichier grid-mapfile au niveau de la machine 'poste', et on ajout deux entrées pour les deux utilisateurs :'globus et saida' :

```
[root@poste ~]# /usr/local/globus-4.0.6/sbin/grid-mapfile-add-entry -dn
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus_poste" -ln globus
```

```
Modifying /etc/grid-security/grid-mapfile ...
New entry:
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus_poste " globus
(1) entry added
```

```
 [root@poste ~]# /usr/local/globus-4.0.6/sbin/grid-mapfile-add-entry -dn
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=root" -ln saida
```

Le résultat est

```
Modifying /etc/grid-security/grid-mapfile ...
New entry:
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=root" saida
(1) entry added
```

Ajout des autorisations inter-postes :

Cette étape se traduit par l'ajout des entrées pour chaque utilisateur d'un poste i dans un autre j. On donne un exemple qui permet à l'utilisateur 'globus' de la machine 'poste' d'être reliée avec le poste 4, on ajoute au map du poste 4 l'entrée composée du sujet de certificat des deux utilisateurs globus des hôts ajoutés.

```
[root@poste4 ~]# /usr/local/globus-4.0.6/sbin/grid-mapfile-add-entry -dn
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus_poste " -ln globus
```

De l'autre côté, on lance :

```
[root@poste ~]# /usr/local/globus-4.0.6/sbin/grid-mapfile-add-entry -dn
"/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus " -ln globus
```

5.1.6 Service 'gridFTP':

A. Configuration du service 'gridFTP':

Le service gridFTP est installé automatiquement lors du lancement du script d'installation de l'intergitiel.

Nous configurons le service gridFTP pour qu'il se lance au démarrage, et pour se faire, nous utilisons xinetd ou inetd. Dans notre cas, nous avons utilisé xinetd, nous lançons son installation comme suit:

```
[root@poste4 ~]# yum install xinetd
```

Résultat:

```
Dependencies Resolved
================================================================================
 Package              Arch       Version          Repository       Size
================================================================================
Installing:
 xinetd               i386       2:2.3.14-14.fc8  fedora           123 k
Transaction Summary
================================================================================
Install      1 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
Total download size: 123 k
Is this ok [y/N]: y
Downloading Packages:
(1/1): xinetd-2.3.14-14.f 100% |=========================| 123 kB    00:04
warning: rpmts_HdrFromFdno: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2
Importing GPG key 0x4F2A6FD2 "Fedora Project <fedora@redhat.com>" from /etc/pki/rpm-gpg/RPM-GPG-
KEY-fedora
Is this ok [y/N]: y
Importing GPG key 0xDB42A60E "Red Hat, Inc <security@redhat.com>" from /etc/pki/rpm-gpg/RPM-GPG-
KEY
Is this ok [y/N]: y
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing: xinetd                       ########################### [1/1]
Installed: xinetd.i386 2:2.3.14-14.fc8
Complete!
```

Nous créons un fichier de configuration 'gridftp' sous */etc/xinetd.d/*, avec la commande suivante:

```
[root@poste4 ~]#vim /etc/xinetd.d/gridftp
```

Et on ajoute le contenu suivant:

```
service gsiftp
{
instances = 100
socket_type = stream
wait = no
user = root
env += GLOBUS_LOCATION = /usr/local/globus-4.0.6
env += LD_LIBRARY_PATH=/usr/local/globus-4.0.6/lib
server = /usr/local/globus-4.0.6/sbin/globus-gridftp-server
server_args = -i   root     70 jui 15 14:59 gridftp.conf
log_on_success += DURATION
```

```
nice = 10
disable = no
}
```

Nous lançons xinetd comme suit:

```
[root@poste4 ~]# /etc/init.d/xinetd start
```

Résultat :

```
Démarrage de xinetd :                                  [  OK  ]
```

Ou avec

```
[root@poste4 ~] /etc/init.d/xinetd reload
```

Résultat :

```
Rechargement de la configuration :                     [  OK  ]
```

On ajoute le service gsiftp à l'ensemble des services locaux, au niveau du fichier *'/etc/services'* :

Nous ajoutons la ligne suivante à la fin du fichier dans la section *'Local services'*.

```
gsiftp          2811/tcp                        # GSI FTP
```

Une visualisation du fichier avec la commande 'tail' est possible:

```
[root@poste4 ~]# tail /etc/services
```

Le résultat est

```
iqobject       48619/udp               # iqobject
vboxd          20012/udp
binkp          24554/tcp               # binkp fidonet protocol
asp            27374/tcp               # Address Search Protocol
asp            27374/udp
dircproxy      57000/tcp               # Detachable IRC Proxy
tfido          60177/tcp               # fidonet EMSI over telnet
fido           60179/tcp
# Local services
gsiftp         2811/tcp                # GSI FTP
```

Nous continuons la configuration de gridFTP en créant le fichier *gridftp.conf* sous *'/etc/grid-security'* et aussi sous *'$GLOBUS_LOCATION/etc/'*.

```
[root@poste4 ~]# vi /etc/grid-security/gridftp.conf
```

Et nous mettrons le contenu suivant:

```
port 2811
allow_anonymous 1
anonymous_user globus
banner "bienvenue!"
```

```
[root@poste4 ~]# cp /etc/grid-security/gridftp.conf /usr/local/globus-4.0.6/etc/
```

Après la configuration de gridFTP, nous lançons le proxy de l'utilisateur globus:

```
[globus@poste4 ~]$ grid-proxy-init -debug -verify
```

Le résultat est:

```
User Cert File: /home/globus/.globus/usercert.pem
User Key File: /home/globus/.globus/userkey.pem
Trusted CA Cert Dir: /etc/grid-security/certificates
Output File: /tmp/x509up_u500
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA-poste2.lri.annaba/OU=lri.annaba/CN=globus
Enter GRID pass phrase for this identity:
Creating proxy .....++++++++++++
.++++++++++++
 Done
Proxy Verify OK
Your proxy is valid until: Tue Jul 15 00:58:16 2008
```

B. Lancement du service 'gridFTP'

On lance manuellement le serveur gridFTP comme suit:

```
[root@poste4 ~]# cd /usr/local/globus-4.0.6/sbin/
[root@poste4 sbin]# ./globus-gridftp-server -c ./gridftp.conf
```

Résultat:

```
Server listening at poste2.lri.net:56312
```

Le port 56312 est choisi automatiquement pour le serveur gridftp, on peut personnaliser le port en ajoutant l'option « -p numéro de port »;

Exemple**:**

[root@poste2 sbin]# ./globus-gridftp-server -c ./gridftp.conf -p 5000.

-Maintenant, on peut utiliser le protocole gsiftp pour l'envoi des fichiers:

Exemple :

```
[globus@poste4 ~]$ globus-url-copy -stripe gsiftp://poste4.lri.net:56312/tmp/gridftp_test
file:///tmp/copie_gridftp_test
```

5.1.7 Lancement du container des services web:

On crée le script permettant le lancement du container. Sous globus, nous exécutons la commande suivante:

```
[globus@poste4 ~]$ vim $GLOBUS_LOCATION/start-stop
```

Et on ajoute le contenu suivant:

```
#!/bin/sh
set -e
export GLOBUS_OPTIONS="-Xms256M -Xmx512M"
sh $GLOBUS_LOCATION/etc/globus-user-env.sh
cd $GLOBUS_LOCATION
case "$1" in
```

```
start)
$GLOBUS_LOCATION/sbin/globus-start-container-detâched -p 8443
;;
stop)
$GLOBUS_LOCATION/sbin/globus-start-container-detâched
;;
*)
echo "Usage : globus start|stop">&2
exit 1
;;
esac
exit 0
```

On donne le droit d'exécution pour ce fichier avec la commande :

```
[root@poste4 ~]$ chmod +x /usr/local/globus-4.0.6/start-stop
```

Sous root**,** nous créons un deuxième script sous *'/etc/init.d/'*, il va être utilisé pour l'appel du premier créé.

```
[root@poste4 ~]# vim /etc/init.d/globus-4.0.6
```

Et on ajoute le contenu suivant:

```
#!/bin/sh -e
case "$1" in
start)
su - globus /usr/local/globus-4.0.6/start-stop start
;;
stop)
su - globus /usr/local/globus-4.0.6/start-stop stop
;;
restart)
$0 stop
sleep 1
$0 start
;;
*)
printf "usage : $0 start|stop|restart|n">&2
exit 1
;;
esac
exit 0
```

On donne plus de priorité pour ce script ainsi le droit d'exécution avec la commande :

```
[root@poste4 ~]# chmod o+x /etc/init.d/globus-4.0.6
```

Maintenant, on lance le container comme suit:

```
[root@poste4 ~]# /etc/init.d/globus-4.0.6 start
```

Le résultat est

```
Starting Globus container. PID: 3554
```

Pour vérifier la réussite du lancement, nous visualisons le fichier « container.log » en exécutant la commande suivante:

```
[root@poste4 ~]# vim /usr/local/globus-4.0.6/var/container.log
```

Le résultat sera :

```
2008-07-13 11:35:37,100 ERROR service.ReliableFileTransferImpl [main,<init>:69] Unable to setup database
driver with pooling.Connection refused. Check that the hostname and port are correct and that the postmaster is
accepting TCP/IP connections.
2008-07-13 11:35:37,550 WARN  service.ReliableFileTransferHome [main,initialize:97] All RFT requests will
fail and all GRAM jobs that require file staging will fail.Connection refused. Check that the hostname and port
are correct and that the postmaster is accepting TCP/IP connections.
Starting SOAP server at: https://192.168.0.103:8443/wsrf/services/
With the following services:
[1]: https://192.168.0.103:8443/wsrf/services/AdminService
[2]: https://192.168.0.103:8443/wsrf/services/AuthzCalloutTestService
[3]: https://192.168.0.103:8443/wsrf/services/CASService
[4]: https://192.168.0.103:8443/wsrf/services/ContainerRegistryEntryService
[5]: https://192.168.0.103:8443/wsrf/services/ContainerRegistryService
[6]: https://192.168.0.103:8443/wsrf/services/CounterService
[7]: https://192.168.0.103:8443/wsrf/services/DefaultIndexService
[8]: https://192.168.0.103:8443/wsrf/services/DefaultIndexServiceEntry
[9]: https://192.168.0.103:8443/wsrf/services/DefaultTriggerService
[10]: https://192.168.0.103:8443/wsrf/services/DefaultTriggerServiceEntry
[11]: https://192.168.0.103:8443/wsrf/services/DelegationFactoryService
[12]: https://192.168.0.103:8443/wsrf/services/DelegationService
[13]: https://192.168.0.103:8443/wsrf/services/DelegationTestService
[14]: https://192.168.0.103:8443/wsrf/services/InMemoryServiceGroup
[15]: https://192.168.0.103:8443/wsrf/services/InMemoryServiceGroupEntry
[16]: https://192.168.0.103:8443/wsrf/services/InMemoryServiceGroupFactory
[17]: https://192.168.0.103:8443/wsrf/services/IndexFactoryService
[18]: https://192.168.0.103:8443/wsrf/services/IndexService
[19]: https://192.168.0.103:8443/wsrf/services/IndexServiceEntry
[20]: https://192.168.0.103:8443/wsrf/services/ManagedExecutableJobService
[21]: https://192.168.0.103:8443/wsrf/services/ManagedJobFactoryService
[22]: https://192.168.0.103:8443/wsrf/services/ManagedMultiJobService
[23]: https://192.168.0.103:8443/wsrf/services/ManagementService
[24]: https://192.168.0.103:8443/wsrf/services/NotificationConsumerFactoryService
[25]: https://192.168.0.103:8443/wsrf/services/NotificationConsumerService
[26]: https://192.168.0.103:8443/wsrf/services/NotificationTestService
[27]: https://192.168.0.103:8443/wsrf/services/PersistenceTestSubscriptionManager
[28]: https://192.168.0.103:8443/wsrf/services/ReliableFileTransferFactoryService
[29]: https://192.168.0.103:8443/wsrf/services/ReliableFileTransferService
[30]: https://192.168.0.103:8443/wsrf/services/RendezvousFactoryService
[31]: https://192.168.0.103:8443/wsrf/services/ReplicationService
[32]: https://192.168.0.103:8443/wsrf/services/SampleAuthzService
[33]: https://192.168.0.103:8443/wsrf/services/SecureCounterService
[34]: https://192.168.0.103:8443/wsrf/services/SecurityTestService
[35]: https://192.168.0.103:8443/wsrf/services/ShutdownService
[36]: https://192.168.0.103:8443/wsrf/services/SubscriptionManagerService
[37]: https://192.168.0.103:8443/wsrf/services/TestAuthzService
[38]: https://192.168.0.103:8443/wsrf/services/TestRPCService
[39]: https://192.168.0.103:8443/wsrf/services/TestService
[40]: https://192.168.0.103:8443/wsrf/services/TestServiceRequest
[41]: https://192.168.0.103:8443/wsrf/services/TestServiceWrongWSDL
```

```
[42]: https://192.168.0.103:8443/wsrf/services/TriggerFactoryService
[43]: https://192.168.0.103:8443/wsrf/services/TriggerService
[44]: https://192.168.0.103:8443/wsrf/services/TriggerServiceEntry
[45]: https://192.168.0.103:8443/wsrf/services/Version
[46]: https://192.168.0.103:8443/wsrf/services/WidgetNotificationService
[47]: https://192.168.0.103:8443/wsrf/services/WidgetService
[48]: https://192.168.0.103:8443/wsrf/services/gsi/AuthenticationService
[49]: https://192.168.0.103:8443/wsrf/services/mds/test/execsource/IndexService
[50]: https://192.168.0.103:8443/wsrf/services/mds/test/execsource/IndexServiceEntry
[51]: https://192.168.0.103:8443/wsrf/services/mds/test/subsource/IndexService
[52]: https://192.168.0.103:8443/wsrf/services/mds/test/subsource/IndexServiceEntry
```

Les erreurs existantes, vont être corrigées après la configuration de la base de données utilisée par RFT.

5.1.8 Configuration du RFT

A. Création du fichier pg_hba.conf

Le fichier pg_hba.conf est un fichier de configuration pour postgressql, il va contenir par la suite l'entrée qui autorise à l'utilisateur globus d'utiliser la base de données rftDatabase à partir du poste4.

On crée le chemin suivant *'/var/lib/pgsql/data'*

```
[root@poste4 ~]# mkdir  /var/lib/pgsql/
[root@poste4 ~]# mkdir  /var/lib/pgsql/data/
```

On édite le fichier pg_hba.conf

```
[root@poste4 ~]# vim /var/lib/pgsql/data/pg_hba.conf
```

On ajoute la ligne suivante :

```
 host rftDatabase globus 192.168.0.101 255.255.255.0 md5
```

B. Création d'un utilisateur globus sous postgres

Nous lançons le serveur postgressql :

```
[postgres@poste4 ~]$ /usr/local/pgsql/bin/postmaster -i -D /usr/local/pgsql/data/
```

Le résultat est le suivant:

```
LOG:  database system was interrupted at 2008-07-13 13:09:54 CET
LOG:  checkpoint record is at 0/42E904
LOG:  redo record is at 0/42E904; undo record is at 0/0; shutdown FALSE
LOG:  next transaction ID: 0/622; next OID: 24579
LOG:  next MultiXactId: 1; next MultiXactOffset: 0
LOG:  database system was not properly shut down; automatic recovery in progress
LOG:  record with zero length at 0/42E94C
LOG:  redo is not required
LOG:  database system is ready
```

On utilise la base de données *test* créée précédemment :

```
[postgres@poste4 ~]$ /usr/local/pgsql/bin/psql test
```

Résultat:

```
Welcome to psql 8.2.6, the PostgreSQL interactive terminal.
Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit
test=#
```

Nous lançons la création de l'utilisateur globus sous postgres en lançant la requête suivante :

```
test=# CREATE USER globus WITH PASSWORD 'globus' CREATEDB;
```

Résultat :

```
CREATE ROLE
```

C. Création de la base de données rftDatabase

Après la création de l'utilisateur, on va lancer la création de la base de données rftDatabase :

```
[globus@poste4 ~]$ /usr/local/pgsql/bin/createdb rftDatabase
```

Résultat :

```
CREATE DATABASE
```

On exécute le script *rft_schema.sql* qui nous permet de créer les schémas de la base de données

```
[globus@poste4 ~]$  /usr/local/pgsql/bin/psql -d rftDatabase -f /usr/local/globus-
4.0.6/share/globus_wsrf_rft/rft_schema.sql
```

Le résultat sera:

```
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:6: NOTICE:  CREATE TABLE / PRIMARY
KEY will create implicit index "requestid_pkey" for table "requestid"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:11: NOTICE:  CREATE TABLE / PRIMARY
KEY will create implicit index "transferid_pkey" for table "transferid"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:30: NOTICE:  CREATE TABLE / PRIMARY
KEY will create implicit index "request_pkey" for table "request"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:65: NOTICE:  CREATE TABLE / PRIMARY
KEY will create implicit index "transfer_pkey" for table "transfer"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:71: NOTICE:  CREATE TABLE / PRIMARY
KEY will create implicit index "restart_pkey" for table "restart"
CREATE TABLE
CREATE TABLE
CREATE INDEX
```

On vérifie les headers du fichier *jndi-config.xml* par la commande :

```
[globus@poste4 globus_wsrf_rft]$  grep -C 3 password $GLOBUS_LOCATION/etc/globus_wsrf_rft/jndi-
config.xml
```

Le résultat sera :

```
</parameter>
        <parameter>
          <name>
          password
          </name>
          <value>
          foo
```

D. Test de fonctionnement du RFT

Avant de tester le bon fonctionnement de rft, on doit relancer le container des web services avec l'option restart s'il est déjà lancé, sinon le lancer de nouveau avec l'option start.

```
[root@poste4 ~]# /etc/init.d/globus-4.0.6 restart
```

Le résultat sera :

```
Stale pid file detected. It will be removed
Starting Globus container. PID: 3554
```

On teste le bon fonctionnement de RFT. On lance la commande qui fait référence au fichier *runtests.xml*

```
[globus@poste4 ~]$ ant -Dtests.jar=/usr/local/globus-4.0.6/lib/globus_wsrf_rft_test.jar -f /usr/local/globus-
4.0.6/share/globus_wsrf_rft_test/runtests.xml
```

Le résultat sera comme suit:

```
Buildfile: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/runtests.xml
init:
   [delete] Deleting directory /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/classes
    [mkdir] Created dir: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/classes
    [unjar] Expanding: /usr/local/globus-4.0.6/lib/globus_wsrf_rft_test.jar into /usr/local/globus-
4.0.6/share/globus_wsrf_rft_test/tests/classes
runServer:
runTests:
_runCustomTests:
    [junit] Running org.globus.transfer.reliable.service.test.PackageTests
    [junit]     GSSException: Expired credentials detected)
    [junit] Tests run: 18, Failures: 0, Errors: 17, Time elapsed: 11,096 sec
    [junit] Test org.globus.transfer.reliable.service.test.PackageTests FAILED
    [junit] Running org.globus.transfer.reliable.service.test.client.PackageTests
    [junit] Tests run: 2, Failures: 2, Errors: 0, Time elapsed: 0,322 sec
    [junit] Test org.globus.transfer.reliable.service.test.client.PackageTests FAILED
    [junit] Running org.globus.transfer.reliable.service.test.connection.PackageTests
    [junit] Tests run: 81, Failures: 0, Errors: 81, Time elapsed: 1,065 sec
    [junit] Test org.globus.transfer.reliable.service.test.connection.PackageTests FAILED
BUILD SUCCESSFUL
Total time: 15 seconds
```

On peut exécuter un nouveau test qui fait référence au fichier mentionné précédemment par la commande suivante :

```
[globus@poste2 ~]$ ant -f /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/runtests.xml generateTestReport
```

Le résultat sera:

```
Buildfile: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/runtests.xml
generateTestReport:
   [delete] Deleting directory /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/test-reports-html
    [mkdir] Created dir: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/test-reports-html
[junitreport] Processing /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/test-reports/TESTS-
TestSuites.xml to /tmp/null6835105
[junitreport] Loading stylesheet jar:file:/usr/local/outils/apache-ant-1.7.0/lib/ant-
junit.jar!/org/apache/tools/ant/taskdefs/optional/junit/xsl/junit-frames.xsl
[junitreport] Transform time: 1050ms
[junitreport] Deleting: /tmp/null6835105
BUILD SUCCESSFUL
Total time: 1 second
```

5.1.9 Configuration du service GRAM

A. Édition du fichier '/etc/sudoers'.

Après avoir installer gridFTP et le RFT, il est maintenant possible de lancer l'installation du gestionnaire des ressources GRAM. Pour se faire, on modifie le fichier *'/etc/sudoers'* :

```
[root@poste4 ~]# vim /etc/sudoers
```

On ajoute :

```
globus ALL=(user1,user2) NOPASSWD: /usr/local/globus-4.0.6/libexec/globus-gridmap-and-
execute
-g /etc/grid-security/grid-mapfile /usr/local/globus-4.0.6/libexec/globus-job-manager-script.pl
*
globus  ALL=(user1,user2) NOPASSWD: /usr/local/globus-4.0.6/libexec/globus-gridmap-
and-execute -g /etc/grid-security/grid-mapfile /usr/local/globus-4.0.6/libexec/globus-gram-
local-proxy-tool *
```

B. Test du fonctionnement de GRAM:

Avec la modification apportée précédemment, nous pouvons lancer GRAM et soumettre des jobs, par exemple, on lance la commande suivante :

```
[globus@poste4 ~]$ globusrun-ws -submit -c /bin/true
```

Le résultat est:

```
Submitting job...Done.
Job ID: uuid:b39456d8-526b-11dd-a4f2-001ec92c2a43
Termination time: 07/16/2008 12:44 GMT
Current job state: Active
Current job state: CleanUp
Current job state: Done
Destroying job...Done.
```

5.1.10 Configuration de gridFTP, RFT et GRAM sous les autres machines :

Après l’installation du certificat sur les autres machines (machine, machine2 et machine3) voir (section 4.4), on procède à la configuration des autres services (gridFTP, lancement des web services, RFT, GRAM) de la même façon que pour la machine4 (serveur du CA).

5.2 Interface Graphique de COG Kit

Après installation de Globus, on a téléchargé le Kit COG [90], et après installation et configuration avec la commande ‘./cog-setup’, on a lancé son interface graphique avec la commande ‘./cog-desktop’ qui se situe dans le bin de la racine. La capture d’écran illustrée par la figure 5-3, montre l’utilisation de ‘cog-gridftp’ en mode graphique pour le transfert des fichiers.

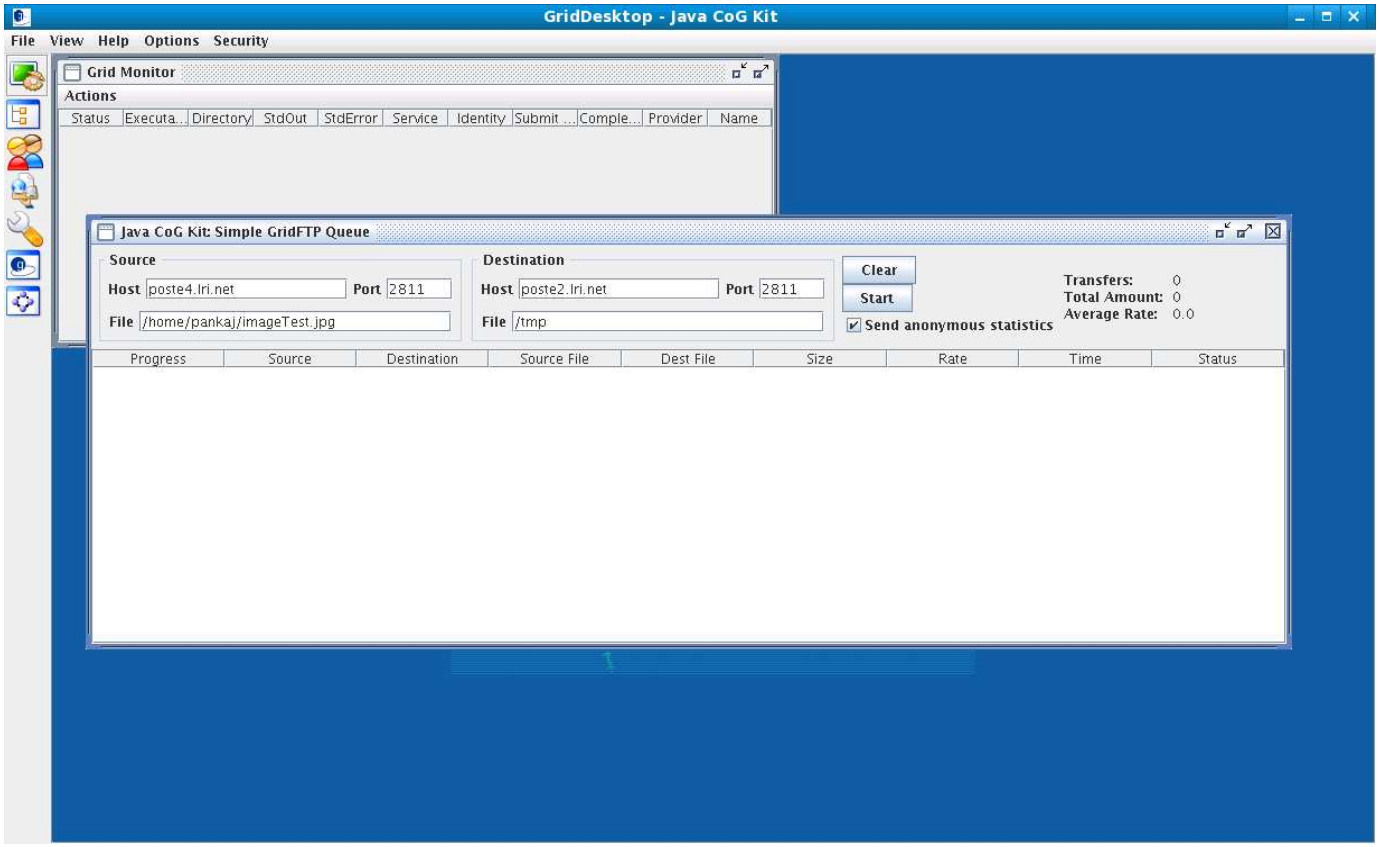


Figure 5-3: Interface graphique COG.

5.3 Conclusion

L’installation et la configuration de globus sont des tâches qui nécessitent beaucoup de précision. Dans cette expérience, l’installation a pris un temps considérable, celà est du aux problèmes liés à l’authentification des utilisateurs de la grille. Ce que l’on peut ****c’est que si l’autorité de certification AC est bien installée, et que les fichiers ‘*cert.pem’ et ‘*key.pem’ sont bien définis, on évitera une grande partie des erreurs qui peuvent se produirent durant le processus d’installation.

ANNEXE B.

6. ANNEXE B

6.1 Traitement des images

6.2.1 Structure d'une région

Une région est représentée principalement par des informations sur l'ensemble des pixels qui la composent, sa position et son voisinage. La partie du code qui suit illustre la structure utilisée l'introduction des méthodes (seteurs, geteurs et compare).

La JVM utilisée est la 1.4, dans laquelle les listes ne sont pas typées. On a utilisé des ArrayList pour la représentation des listes d'objet, et pour la lecture, on parse vers le type demandé.

```
import java.util.ArrayList;

public class Region implements Comparable{



        int xPosDebut;

        int yPosDebut;



        int etiquette;

        double moyenne;

        double variance;

        ArrayList couleurs ;

        //

        int nbrPixels;

        ArrayList voisins=new ArrayList();

        ArrayList listePixels=new ArrayList();



        Region(){}

        Region(int xPosDebut,int yPosDebut,int xPosFin,int yPosFin,int type,int etiquette){
```

```
            this.xPosDebut=xPosDebut;

            this.yPosDebut=yPosDebut;

            this.etiquette=etiquette;

            //nbrPixels

            this.nbrPixels=(xPosFin+1-xPosDebut)*(yPosFin+1-yPosDebut);

            voisins=new ArrayList();



    }
```

6.2.2 Structure de fichier .seg

Les fichiers .seg sont utilisés pour représenter l'ensemble des régions composantes de l'image, ils sont représentés par un objet de type ArrayList avec le contenu de type Région.

6.2.3 Calcul des caractéristiques visuelles d'une image

Méthodes pour le calcul des caractéristiques extraites de la matrice de cooccurrence définie par Haralick :

L'énergie :

```
private double getEnergie(double[][] cooccurreceMatrix) {

            double energie=0;

            for(int j=0;j<SizeM;j++) { //SizeM : taille de la matrice de cooccurrence,
générealement a la valeur 256 (image en niveau de gris)

                for(int i=0;i<SizeM;i++) {

                    energie += cooccurreceMatrix [i][j]*cooccurreceMatrix[i][j];

                }

            }

            return energie;
```

```
        }
```

L'entropie :

```
        private double getEntrepie(double[][] cooccurreceMatrix) {

                double entropy=0;

                for(int j=0;j<SizeM;j++) { //SizeM : taille de la matrice de cooccurrence,
généralement a la valeur 256 (image en niveau de gris)

                        for(int i=0;i<SizeM;i++) {

                        if (cooccurreceMatrix [i][j]==0) continue;

                        entropy+= cooccurreceMatrix [i][j]*Math.log(cooccurreceMatrix [i][j]);

                }

            }

                return -entropy;

        }
```

Le contraste :

```
        private double getContrast(double[][] cooccurreceMatrix) {

            double contrast=0;

            for(int j=0;j<SizeM;j++) {//SizeM : taille de la matrice de cooccurrence,
généralement a la valeur 256 (image en niveau de gris)

                        for(int i=0;i< SizeM;i++) {

                                contrast+=(i-j)*(i-j)* cooccurreceMatrix [i][j];

                        }

                }

                return contrast;
```

```
}
```

Le moment inverse de différence :

```
private double getMomentInverseDiff(double[][] cooccurreceMatrix) {

        double momInvDiff=0;

        double coef;

        for(int j=0;j< SizeM;j++) {//SizeM : taille de la matrice de cooccurrence, généralement a la valeur 256 (image en niveau de gris)

                for(int i=0;i< SizeM;i++) {

                        coef = 1.0/(1.0+(i-j)*(i-j));

                        momInvDiff +=coef* cooccurreceMatrix [i][j];

                }

        }

        return momInvDiff;

}
```

6.2 Application

6.2.1 Structure de la table des images indexées

Cette table contient la liste des images qui ont été indexées, elle est composée de l'ID de l'image, l'ID de la classe qui l'appartient, le pourcentage d'appartenance à la classe désignée précédemment, le chemin physique de l'image, sa résolution, la taille physique et enfin une lise de couleurs séparées par un signe dollar.

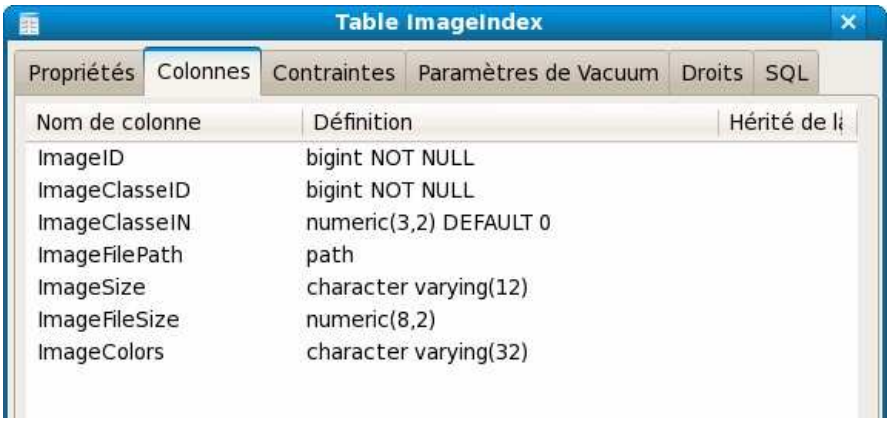


Figure 6-1: Structure de la table Index des images.

Un exemple simplifié d'une requête SQL générée à partir de l'analyse de la phrase 'Voiture rouge' est : *SELECT * FROM INDEX WHERE ImageClasse LIKE '%VOITURE%' AND ImageColors LIKE'%ROUGE%' ORDER BY ImageClasseIN*

Comme une image peut appartenir à plusieurs classes d'images, on peut trouver plusieurs entrées pour une seule image dans la table précédente.

6.2.2 Environnement de développement

Afin de pouvoir développer notre application de grille en Java, l'environnement illustré par la figure 6-2 doit être dotée d'un serveur de base de données (PostgreSQL), d'un serveur sur lequel on peut déployer notre application (ANT), d'une JVM bien entendu et enfin, d'un middleware de grille qui est représenté par le ToolKit Globus.

Figure 6-2 : environnement utilisé pour le développement.

L'application développée est composée de trois parties principales : le côté client, l'ensemble des ressources, et enfin le côté serveur qui va être déployé sur le container. La figure 6-3 illustre les principales composantes simplifiées d'une application de grille.

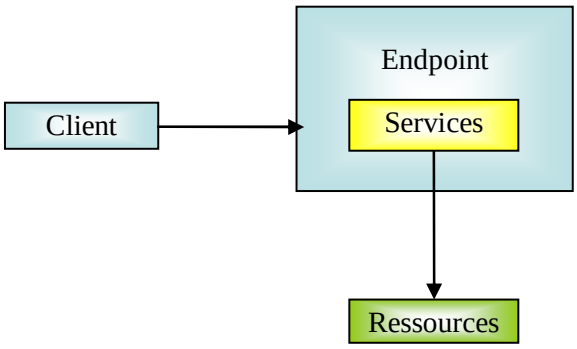


Figure 6-3: structure générale de l'application.

Pour pouvoir gérer les ressources, on doit implémenter une classe Ressources Home qui est utilisée principalement pour la localisation des ressources dans la grille (données). Et afin de pouvoir créer une multitude de ressources pour l'application, on utilise une classe Factory qui instancie la classe précédente (Ressources Home) pour la création effective de la ressource, la figure 6-4, montre la structure générale pour la création du service de segmentation.

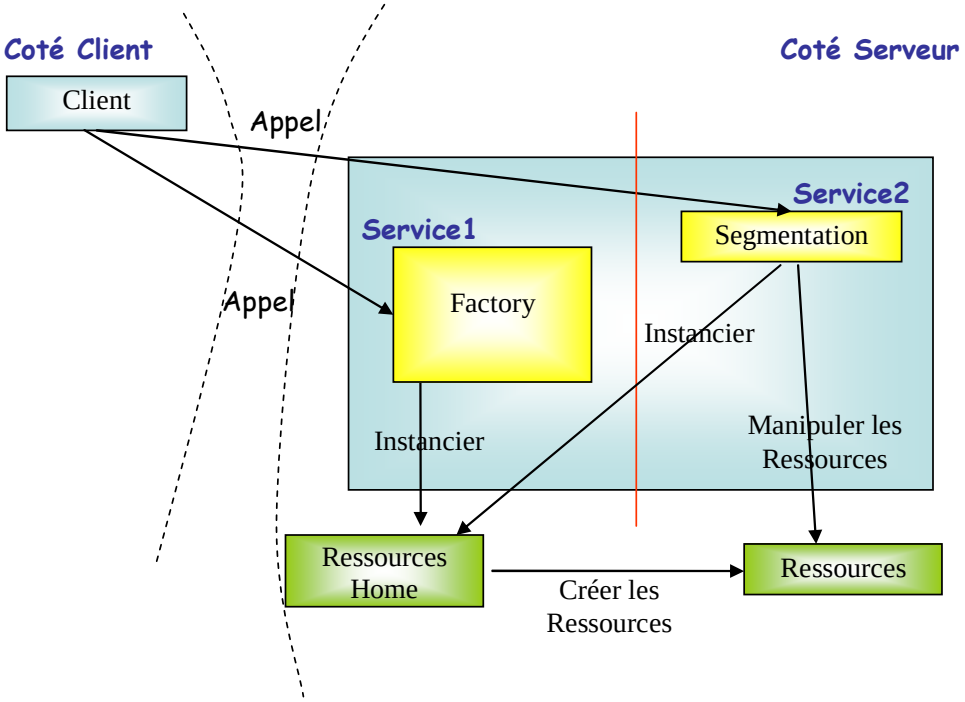


Figure 6-4: structure de service de segmentation.

6.2.3 Outil d'aide au développement

Plusieurs outils de développement des Services Web sont disponibles et peuvent aider au développement des services de grille avec un serveur TomCat, on peut citer eclipse (IBM), NetBeans (SUN).

6.2.4 Etapes de développement de service de segmentation

Le développement passe par cinq étapes principales :

A. Définition de l'interface du service

L'interface représente une classe dans laquelle on spécifie l'ensemble de méthodes que le service peut mettre en œuvre.

Dans le cas des services Web et Services de grille, la définition de l'interface se fait avec le langage descriptif WSDL, il repose sur la notation XML. WSDL permet de décrire les méthodes, leurs paramètres et les types de données utilisées ainsi que les messages d'entrées et de sorties. Il est composé des parties suivantes :

portType : Définit le service, les opérations et les Message échangés pour chacune d'eux.

Message: Définit les messages échangés et leurs paramètres.

Types : Définit les méthodes, les types de paramètres du service et les ressources disponibles ainsi que leurs type.

Binding : Précise le protocole utilisé et le format de message.

Le code suivant représente l'interface simplifié du service de segmentation

```
<?xml version="1.0" encoding="UTF-8"?>

<definitions name="AprioriService"


targetNamespace="http://www.globus.org/namespaces/examples/core/SegmentationService_instance
"

   xmlns="http://schemas.xmlsoap.org/wsdl/"

   xmlns:tns="http://www.globus.org/namespaces/examples/core/SegmentationService_instance"

   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

   xmlns:wsrp="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-
01.xsd"

   xmlns:wsrpw="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-
01.wsdl"
```

```
    xmlns:wsdlpp="http://www.globus.org/namespaces/2004/10/WSDLPreprocessor"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<wsdl:import
    namespace=
    "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.wsdl"
    location="../../wsrf/properties/WS-ResourceProperties.wsdl" />

<!--============================================================

                T Y P E S

 ============================================================-->
<types>
<xsd:schema
targetNamespace="http://www.globus.org/namespaces/examples/core/SegmentationService_instance
"
    xmlns:tns="http://www.globus.org/namespaces/examples/core/SegmentationService_instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

        <!-- REQUESTS AND RESPONSES -->

        <xsd:element name="segmentation" type="xsd:string"/>
        <xsd:element name="segmentationResponse">
```

```
            <xsd:complexType/>
        </xsd:element>

        <xsd:element name="getValueRP">
            <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getValueRPResponse" type="xsd:string"/>

        <!-- RESOURCE PROPERTIES -->

        <xsd:element name="Value" type="xsd:string"/>
        <xsd:element name="LastOp" type="xsd:string"/>

        <xsd:element name="SegmentationResourceProperties">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="tns:Value" minOccurs="1" maxOccurs="1"/>
                <xsd:element ref="tns:LastOp" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
        </xsd:element>

</xsd:schema>
</types>
```

```
<!--=========================================================

                M E S S A G E S

 ========================================================-->
<message name="SegmentationInputMessage">
        <part name="parameters" element="tns:segmentation"/>
</message>
<message name="SegmentationOutputMessage">
        <part name="parameters" element="tns:segmentationResponse"/>
</message>

<message name="GetValueRPInputMessage">
        <part name="parameters" element="tns:getValueRP"/>
</message>
<message name="GetValueRPOutputMessage">
        <part name="parameters" element="tns:getValueRPResponse"/>
</message>




<!--=========================================================

                P O R T T Y P E
```

```
 ============================================================-->

<portType name="SegmentationPortType"

  wsdlpp:extends="wsrpw:GetResourceProperty"

  wsrp:ResourceProperties="tns:SegmentationResourceProperties">


        <operation name="segmentation">

              <input message="tns:SegmentationInputMessage"/>

              <output message="tns:SegmentationOutputMessage"/>

        </operation>


        <operation name="getValueRP">

              <input message="tns:GetValueRPInputMessage"/>

              <output message="tns:GetValueRPOutputMessage"/>

        </operation>


</portType>


</definitions>
```

B. Implémentation de l'interface

Cette étape est traduite par la création des classes suivantes : SegmentationService, SegmentationQNames, SegmentationResource, segmentationProcess, SegmentationResourceHome, SegmentationFactoryService.

La figure 6-5, montre la structure de la classe SegmentationService, les classes importées et les méthodes développées.

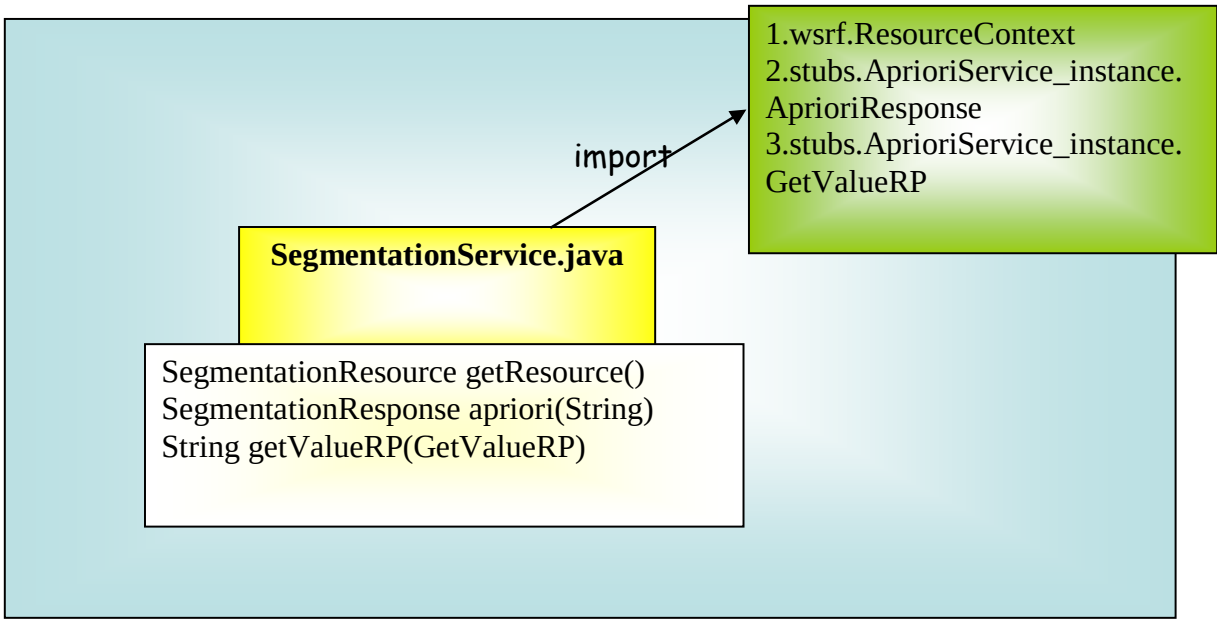


Figure 6-5: classe de service de segmentation.

La figure 6-6 montre la classe SegmentationQName.

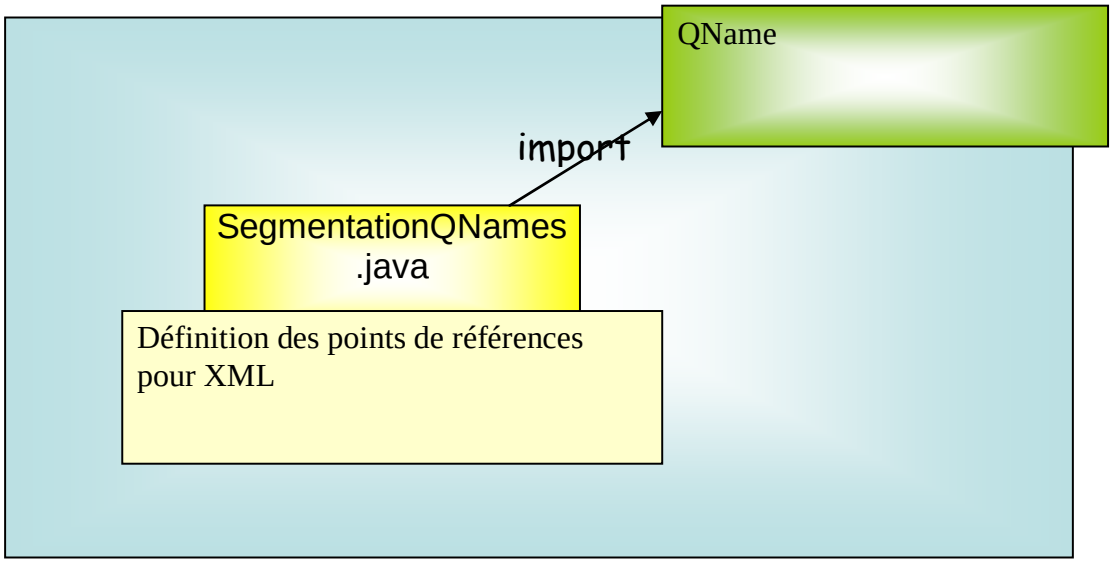


Figure 6-6: classe SegmentationQName.

La figure 6-7 montre la classe Ressource.

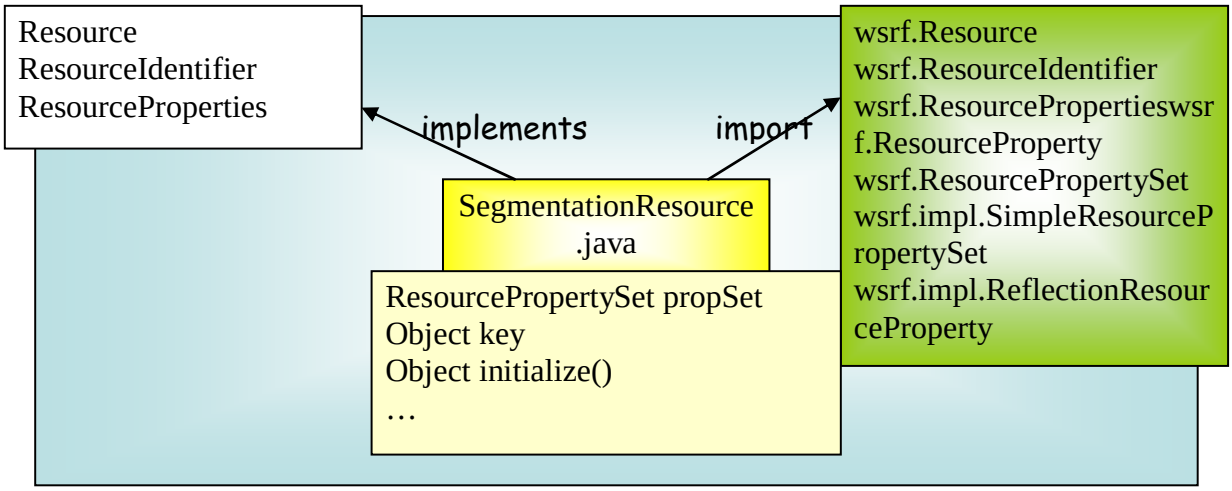


Figure 6-7: classe SegmentationRessource.

La figure 6-8 montre la classe qui implémente l'algorithme de segmentation.

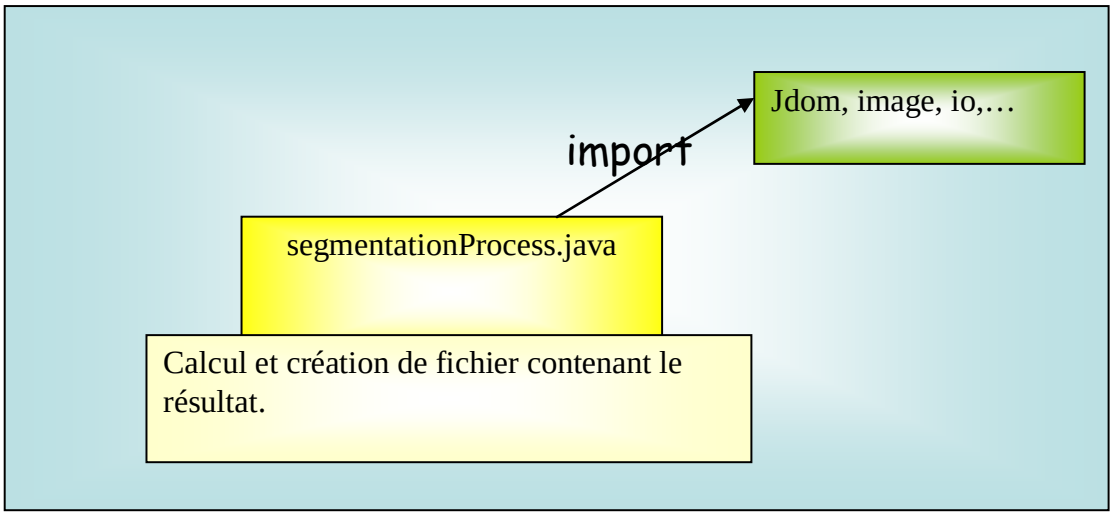


Figure 6-8: classe segmentationProcess.

La figure 6-9 montre la classe Factory.

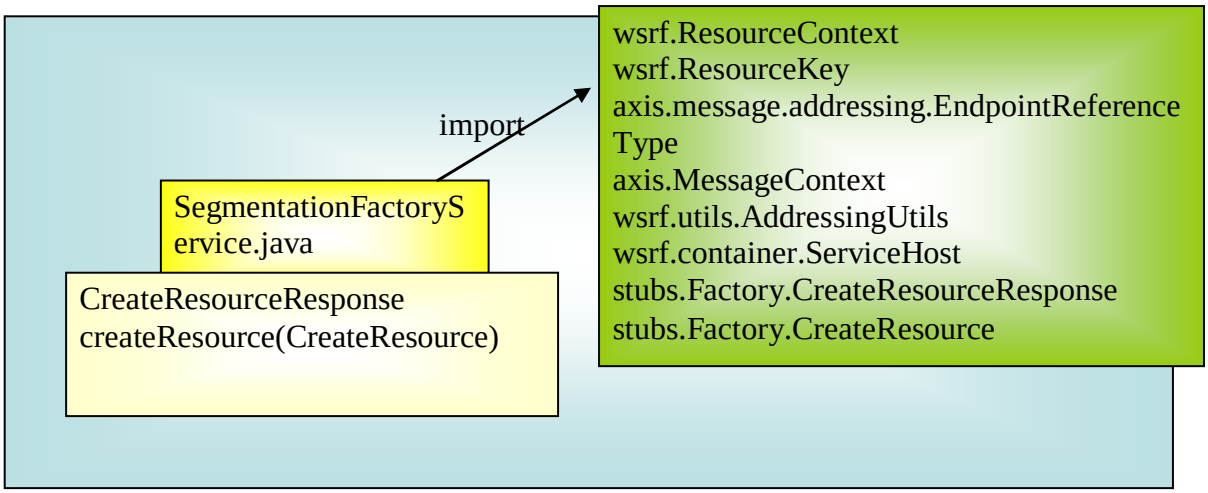


Figure 6-9: classe SegmentationFactoryService.java

La figure 6-10 montre la classe SegmentationRessourceHome.

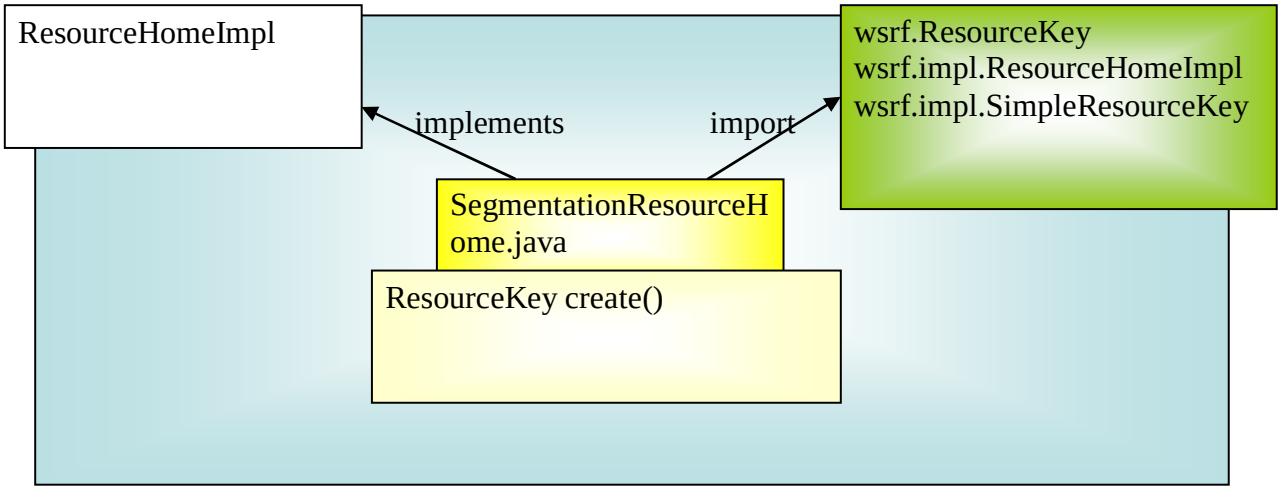


Figure 6-10: classe SegmentationRessourceHome

La figure 6-11 montre l'interaction entre toutes les classes Java qui composent le service de segmentation.

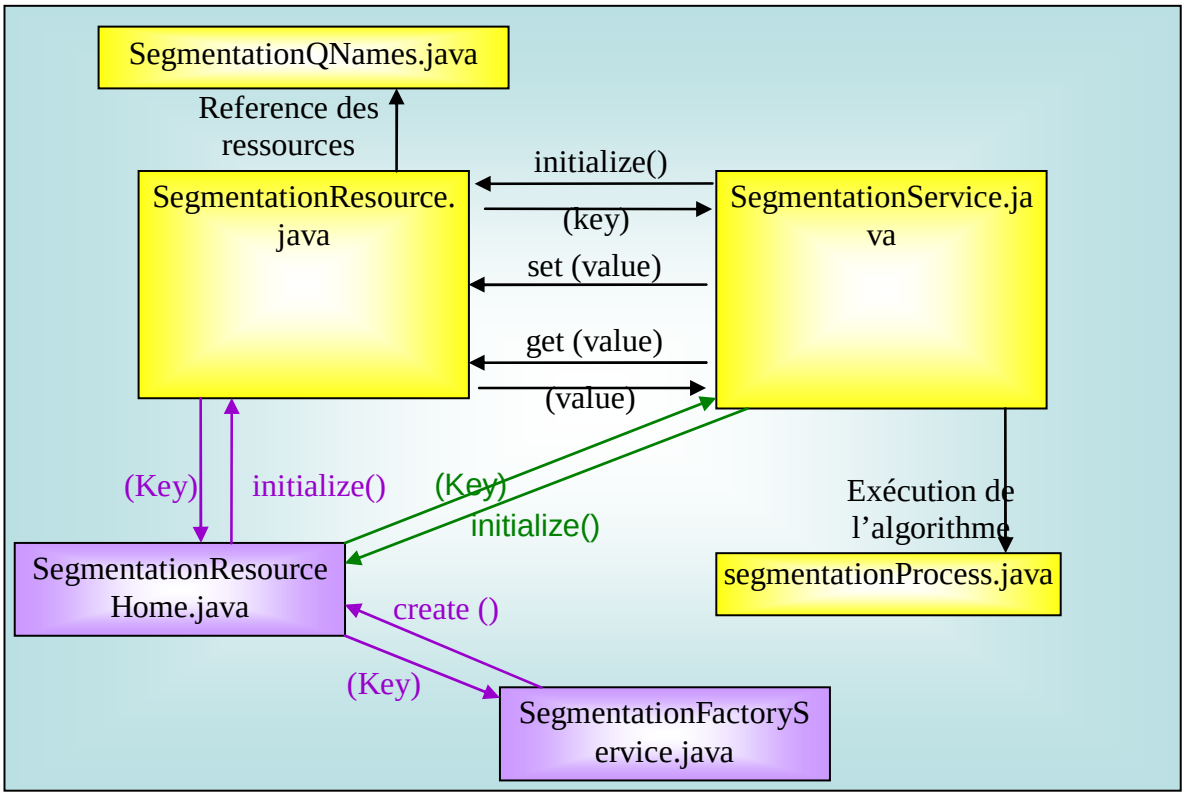


Figure 6-11: interaction entre classes de segmentation.

C. Développement Côté client

Il est composé d'interfaces graphiques, il lance le service d'information et de Synchronisation (SIS), ce dernier lance le service de segmentation en lui spécifiant les ressources sur lesquelles il va travailler.

La figure 6-12 illustre l'interface graphique qui instance le service SIS et qui permet de lui passer les paramètres (source de données, propriétés..).

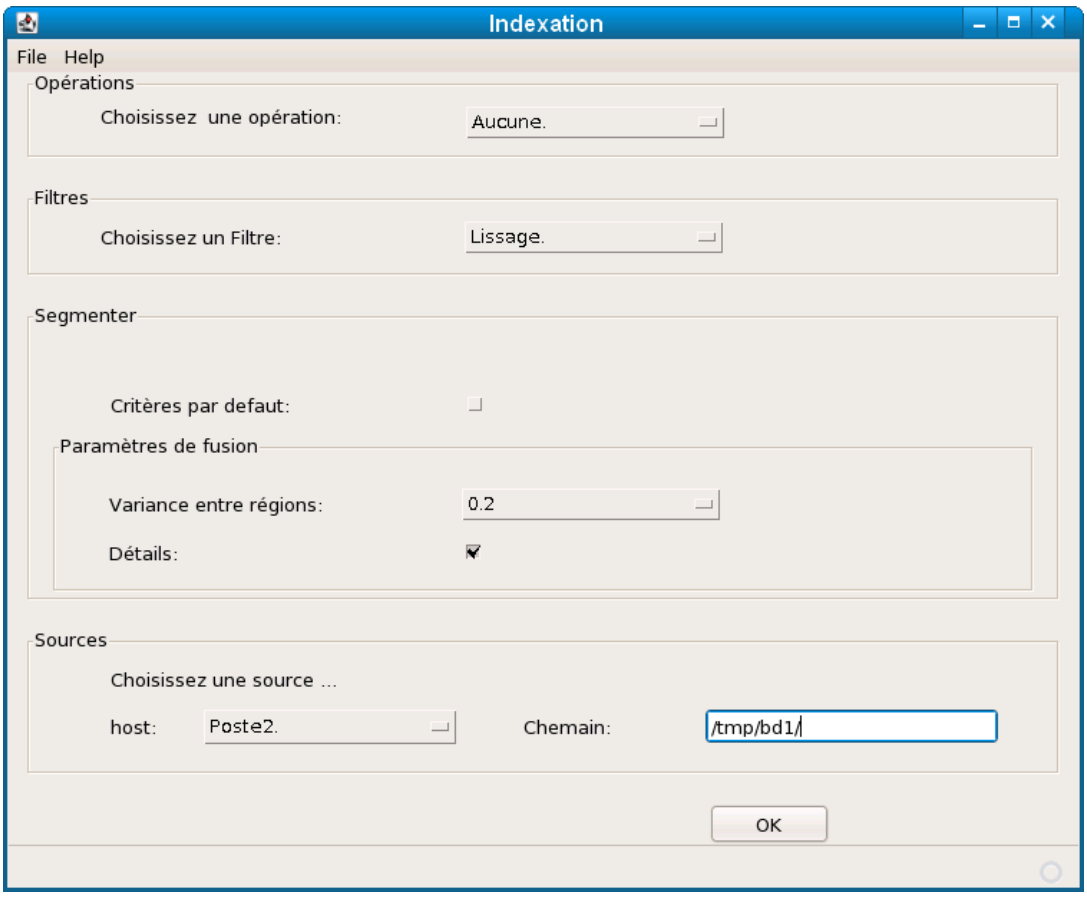


Figure 6-12: interface client pour effectuer une indexation d'une source de données.

La figure 6-13 montre l'interface graphique utilisée pour la recherche, l'utilisateur peut spécifier une requête image ou introduire des mots clés.

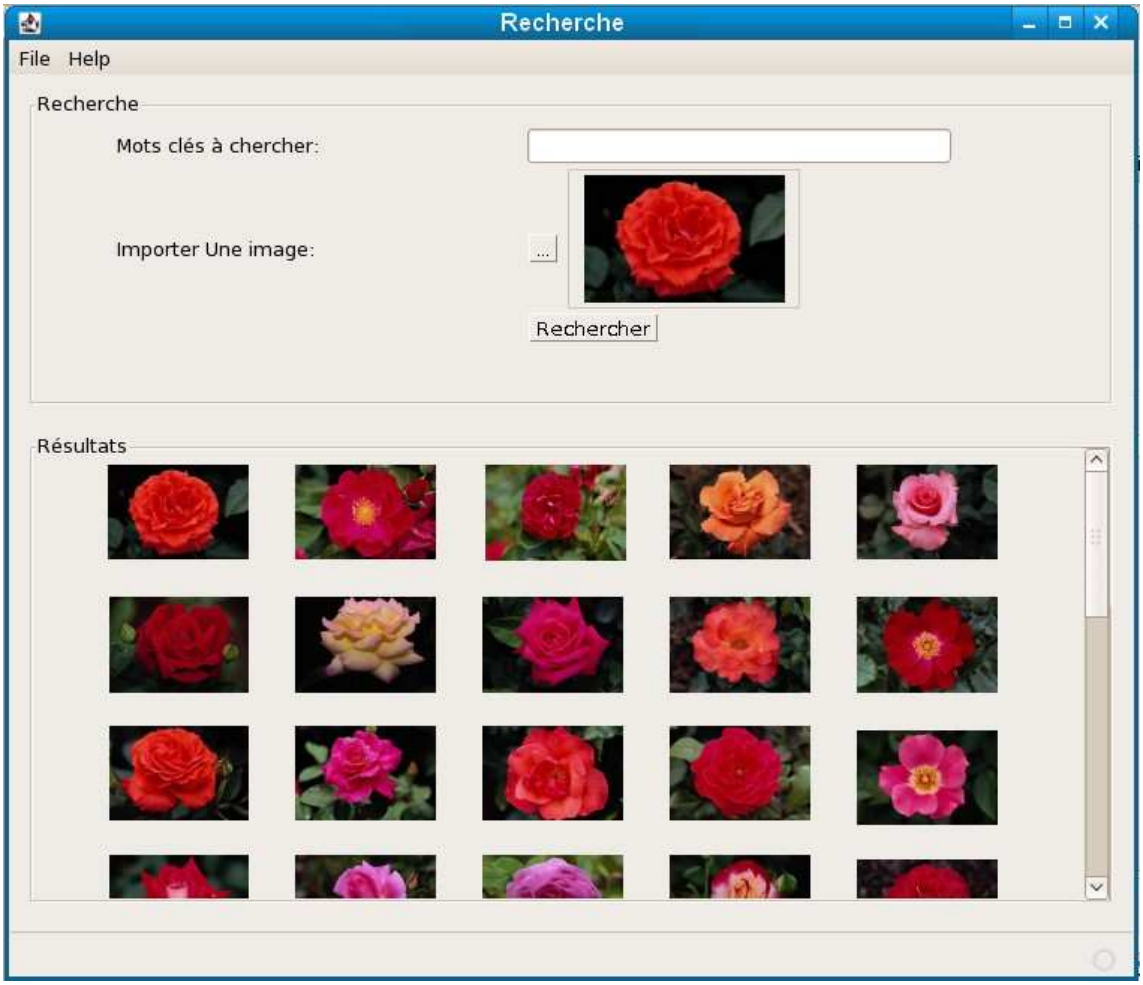


Figure 6-13: interface graphique pour la recherche.

D. Configuration de déploiement du service

Se traduit par la création d'un fichier 'deploy-server.wsdd', ce type de fichier wsdd (Web Service Deployment Descriptor) permet de décrire la façon avec laquelle le service sera déployé. Il fournit les informations suivantes :

- Le nom du service avec le quel on peut extraire l'URI.
- Le chemin du répertoire contenant le code source du service.
- Le chemin de l'interface WSDL du service à partir duquel les fichiers WSDL sont ajoutés à $GLOBUSLOCATION/share/schema/.
- Paramètres de configuration utiles pour le déploiement et la façon avec laquelle il doit être déployé.

Création d'un fichier deploy-jndi-config.xml : utilisé pour la désignation de multiple ressources pour les services.

E. Génération du fichier GAR

Le fichier GAR est une structure similaire au fichier WAR pour les applications J2EE, il contient le binaire des classes du service ainsi que les fichier de descriptions de déploiement.

On lance la commande globus-build-service avec le nom du service comme paramètre. Exemple de génération du GAR du service factory : globus-build-service factory.

Le déploiement se fait par la commande ‘globus-deploy-gar’, et on ajoute le nom du GAR généré précédemment. On redémarre le conteneur, et le service sera prêt à être utilisé.

6.2.5 Conclusion

Le développement des services de grilles nécessite des connaissances profondes, et durant l’implémentation des services, on a trouvé qu’il y a un manque d’outils de développement et de tests efficaces. Les services de ce projet ont été développés en Java avec l’importation de quelques librairies fournies par Globus. L’utilisation d’autres langages de programmation nous permet de bénéficier d’autres fonctionnalités tel que le langage C qui fournit plusieurs fonctions de traitement des images, et qui peuvent élargir le contexte de ce travail. Il existe une possibilité pour instancier des classes écrites avec d’autre langage à partir d’une classe Java, cela aussi ouvre beaucoup de chemins pour enrichir l’application.

Conclusion et perspectives

Ce mémoire a porté sur la conception et le développement d'un système de recherche d'images à base de contenus, et l'exploitation d'infrastructure de grille de calcul pour mettre en œuvre des procédures d'analyse de grande quantité de données. L'amélioration des performances des plateformes CBIR est la principale motivation de ce travail.

Le travail réalisé a mené à la réalisation d'un manuel d'installation et de configuration de l'Intergitiel Globus nécessaire pour la mise en place d'une grille de calcul. Un but aussi atteint et qui n'est pas négligeable est qu'on a acquit une expérience dans le contexte de développement des services de grilles, et qui ouvre la porte vers l'enrichissement de la plateforme développée.

Avec ce travail, on a arrivé à minimiser le temps de segmentation d'une base qui contient 10000 images en 25% du temps écoulé pour un traitement séquentiel, ce qui représente un bon début pour d'autres travaux locaux qui porteront sur le griding.

La gestion de la grille représente un grand défi pour plusieurs projets. Les résultats obtenus sur la grille de test ont prouvé la bonne gestion gérée par les services de la plateforme. La bonne gestion d'exécution et la gestion des données sont bien traduites par la minimisation de temps d'indexation des images. La qualité des résultats de la recherche des images, repose sur les algorithmes implémentés durant la phase de prétraitement et la phase d'analyse des images, ce qui entre dans un autre domaine de recherche. Pour cela, on a implémenté les méthodes les plus utilisées en imagerie, et on a montré la façon avec laquelle on peut augmenter la plateforme.

L'implémentation réelle de la plateforme exclut le module de retour de pertinence pour des raisons liées avec leurs temps de développement et le temps de recherche nécessaire.

La plateforme réalisé manque un système efficace pour permettre des ajouts sous forme de plugin, dans les travaux qui suit, on va focaliser sur ce point de plugin et aussi sur la réalisation d'outils d'aide aux développements des services de grille avec plusieurs langages de programmation.

Le travail réalisé ouvre la voie vers la mise en place d'une grille à l'échelle de l'environnement informatique de l'université pour fournir une grille éprouvée. Avec de domaine de travail, plusieurs recherches en physique, en biologie et en informatique vont largement bénificier des performances que les grilles peuvent fournir.

Références

[1] I. Foster, C. Kesselman, S. Tuecke. “The Anatomy of the Grid: Enabling Scalable Virtual Organizations”. International J. Supercomputer Applications, 15(3), 2001.

[2] I. Foster, C. Kesselman, “The Grid: Blueprint for a New Computing Infrastructure”. Morgan Kaufmann, 1999.

[3] Borja Sotomayor and Lisa Childers. “Globus® Toolkit 4: Programming Java Services”. Morgan Kaufmann publisher. 2006.

[4] Chetty, M. et R. Buyya, Weaving Computational Grids: How Analogous Are they with Electrical Grids, Computing in Science & Engineering, July/August 2002.

[5] I. Foster et C. Kesselman editors. "The Grid 2: Blueprint for a New Computing Infrastructure". Morgan Kaufmann publisher. 2004.

[6]Condor. http://www.cs.wisc.edu/condor/.

[7]http://www.npac.syr.edu/factoring.html.

[8] Foster, I., Geisler, J., Nickless, W., Smith, W. and Tuecke, S. A997) Software infrastructure for the I-WAY high performance distributed computing experiment. Proc. 5th IEEE Symposium on High Performance Distributed Computing, 1997.

[9] SETI@Home, http://setiathome.ssl.berkeley.edu/.

[10] Distributed Net, http://www.distributed.net/.

[11] Foster, I. and Kesselman, C. A997 Globus: A metacomputing infrastructure toolkit. International Journal of Supercomputer Applications, 11B, 1997.

[12] Grimshaw, A. et al. A997 The legion vision of a worldwide virtual computer. Communications of the ACM, 40A.

[13] David De Roure, Mark A. Baker, Nicholas R. Jennings, and Nigel R. Shadbolt. The evolution of the Grid. Grid Computing: Making the Global Infrastructure a Reality. 2003.

[14] Daniel Minoli. “A Networking Approach to Grid Computing”. Publié par John Wiley & Sons, Inc., Hoboken, New Jersey. 2005.

[15]D.Snelling J.Almond. Unicore: secure and uniform access to distribuated resources via the www, October 1998.

[16]D. Erwin (Ed.), UNICORE Plus final Report - Uniform Interface to Computing Resources, Forschungszentrum Jülich, 2003.

[17] The Open Grid Services Architecture and Data Grids – with Leanne Guy, in Grid Computing: Making The Global Infrastructure a Reality (Edited by Fran Berman), p 395ff. John Wiley & Sons 2003.

[18]Dennis Gannon, Kenneth Chiu, Madhusudhan Govindaraju, Aleksander Slominski. Department of Computer Science. Indiana University, Bloomington IN 47405 “An Revised of The Open Grid Services Architecture“. October 25, 2002

[19] David Booth, W3C Fellow / Hewlett-Packard, Hugo Haas, W3C, Francis McCabe, Fujitsu Labs of America, Eric Newcomer (until October 2003), Iona Michael Champion (until March 2003), Software AG, Chris Ferris (until March 2003), IBM, David Orchard (until March 2003), BEA. “SystemsWeb Services Architecture”. W3C Working Group Note 11 February 2004

[20] Maozhen Li Brunel University UK, Mark Baker University of Portsmouth UK. “The Grid Core Technologies”. 2005.

[21] Steve Graham (IBM) (Editor), Karl Czajkowski (Globus / USC/ISI), Donald F Ferguson (IBM), Ian Foster (Globus / Argonne), Jeffrey Frey (IBM), Frank Leymann (IBM), Tom Maguire (IBM), Nataraj Nagaratnam (IBM), Martin Nally (IBM), Tony Storey (IBM), Igor Sedukhin (Computer Associates International), David Snelling (Fujitsu Laboratories of Europe), Steve Tuecke (Globus / Argonne), William Vambenepe (Hewlett-Packard), Sanjiva Weerawarana (IBM). “Web Services Resource Properties (WS-ResourceProperties)” Version 1.1 03/05/2003

[22] I. Foster, C. Kesselman, J. Nick, S. Tuecke, Open Grid Service Infrastructure WG, Global Grid Forum. “The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration”. June 22, 2002.

[23] globus.org

[24] Ian Foster (Globus Alliance / Argonne National Laboratory) (Editor), Jeffrey Frey (IBM) (Editor), Steve Graham (IBM) (Editor), Steve Tuecke (Globus Alliance / Argonne National Laboratory) (Editor), Karl Czajkowski (Globus Alliance / USC ISI). Don Ferguson (IBM), Frank Leymann (IBM), Martin Nally (IBM),Igor Sedukhin (Computer Associates International), David Snelling (Fujitsu Laboratories of Europe), Tony Storey (IBM), William Vambenepe (Hewlett-Packard), Sanjiva Weerawarana (IBM). “Modeling Stateful Resources with Web Services, Version 1.1”. 03/05/2004.

[25]Ian Foster. “Globus Toolkit Version 4: Software for Service-Oriented Systems”.2005.

[26] Lee Liming, University of Chicago, Argonne National Laboratory. “Globus Primer: Introduction to Globus Software”. OSGCC08.

[27] Ian Foster. “Globus Toolkit Version 4: Software for Service-Oriented Systems”.J.Comput.Sci. & Technol., July 2006, Vol.21, N.4, pp.513-520.

[28] W. Allcock, J. Bresnahan, R. Kettimuthu, M. Link, C. Dumitrescu, I. Raicu, I. Foster, The Globus Striped GridFTP Framework and Server, SC'05, ACM Press, 2005.

[29] Communicating Security Assertions over the GridFTP Control Channel. Rajkumar Kettimuthu, Liu Wantao, Frank Siebenlist and Ian Foster. 4th IEEE International Conference on e-Science, December 2008. 426-427.

[30] RELIABLE FILE TRANSFER IN GRID ENVIRONMENTS. Ravi K Madduri, Argonne National Laboratory, Illinois Institute of Technology. Cynthia S. Hood, Illinois Institute of Technology. William E. Allcock, Argonne National Laboratory. Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN’02). 0742-1303/02 17.00 © 2002 IEEE.

[31] Performance and Scalability of a Replica Location Service. A.L. Chervenak, N. Palavalli, S. Bharathi, C. Kesselman, R. Schwartzkopf. Proceedings of the International IEEE Symposium on High Performance Distributed Computing (HPDC-13), June 2004.

[32] Protocols and Services for Distributed Data-Intensive Science. B. Allcock, S. Tuecke, I. Foster, A. Chervenak, and C. Kesselman. ACAT2000 Proceedings, pp. 161-163, 2000.

[33] Wide Area Data Replication for Scientific Collaborations. A. Chervenak, R. Schuler, C. Kesselman, S. Koranda, B. Moe. Proceedings of 6th IEEE/ACM International Workshop on Grid Computing (Grid2005), November 2005.

[34] Numerical Relativity in a Distributed Environment. W. Benger, I. Foster, J. Novotny, E. Seidel, J. Shalf, W. Smith, P. Walker. Ninth SIAM Conference on Parallel Processing for Scientific Computing, April 1999.

[35] A National-Scale Authentication Infrastructure. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer, V. Welch. IEEE Computer, 33(12):60-66, 2000.

[36] GT4 GRAM: A Functionality and Performance Study. M. Feller, I. Foster, and S. Martin. TERAGRID 2007 CONFERENCE, MADISON, WI (SUBMITTED).

[37] http://www.mcs.anl.gov/~smartin/gram/

[38] Performance Analysis of the Globus Toolkit Monitoring and Discovery Service, MDS2. X. Zhang and J. Schopf. Proceedings of the International Workshop on Middleware Performance (MP 2004), part of the 23rd International Performance Computing and Communications Workshop (IPCCC), pp 843- 849 April 2004.

[39] http://www.mcs.anl.gov/~schopf/Pubs/MDS4.PDJuly05.pdf (MDS4 and Project Deployments)

[40] R. M. Haralick, K. Shanmugan, and I. Distein, "Textural features for image classification," IEEE Trans. Syst., vol. SMC-3, pp. 610-621, June 1973.

[41] Comparaison Entre la Matrice de Cooccurrence et la Transformation en Ondelettes pour la Classification Texturale des Images SPOT-XS Abdelmounaime Safia, Tahar Iftene. CISTEMA2003.

[42] IRichard E. Woods. "Digital Image Processing". 2001.

[43] A. K. J ain. Fundamentals of Digital Image Processing. Prentice-Hall Information and System Sciences Series. Prentice-Hall International, London,
1989.

[44] Shim, S., Choi, T., 2003. Image indexing by modified color cooccurrence matrix. In: Proc. Internat. Conf. on Image Processing.

[45] A. Vadivel, Shamik Sural, A. K. Majumdar. An Integrated Color and Intensity Co-occurrence Matrix, Pattern Recognition Letters, Vol. 28, No. 8. (1 June 2007), pp. 974-983.

[46] www.ensta.fr/~manzaner/Cours/IAD/AM_Segmentation.pdf

[47] Malika CHARRAD. Techniques d'extraction de connaissances appliquées aux données du Web, Ecole Nationale des Sciences de l'Informatique, Université de la Manouba, Tunis ,2005

[48] Brochu, F. et al. (2009) "Ganga: a tool for computational-task management and easy access to Grid resources", published in arXiv:0902.2685v1, submitted to Comp. Phys. Comm, 2009.

[49] Dr Chris Town and Dr Karl Harrison. "Large-scale Grid Computing for Content-based Image Retrieval". ISKO UK 2009 conference, 22-23 June.

[50] "Medical Image Processing Workflow Support on the EGEE Grid with Taverna"

Ketan Maheshwari University of Nice, France. Paolo Missier University of Manchester, UK. Carole Goble University of Manchester, UK.Johan Montagnat University of Nice, Frnace. 2009.

[51] T. Oinn, M. Greenwood, M. Addis, and M. N. A. et al. Taverna: Lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experience, 18(10):1067-1100, August 2006.

[52] http://www.mygrid.org.uk/ myGrid

[53] D. Lingrand, J. Montagnat, and T. Glatard. Estimating the execution context for rening submission strategies on production grids. In Assessing Models of Networks and Distributed Computing Platforms (ASSESS, CC-grid'08), pages 753-758, Lyon, May 2008. IEEE.

[54] Bo Song; Guangwen Yang; Qiming Fang, "Improving the Performance of MPI Applications over Computational Grid" Grid, Page(s):799 - 806, 16-18 Aug. 2007.

[55] M. J. Pitkanen, Xin Zhou, A. Hyvarinen, H. Muller. Using the Grid for Enhancing the Performance of a Medical Image Search Engine. Computer-Based Medical Systems, 2008. CBMS '08. 21st IEEE

International Symposium on In Computer-Based Medical Systems, 2008. CBMS '08. 21st IEEE International Symposium on (2008), pp. 367-372.

[56] http://www.sim.hcuge.ch/medgift/

[57] Mattias Ellert et al., ”Advanced Resource Connector middleware for lightweight computational Grids”, Future Generation Computer Systems, 23 (2007).

[58] http://www.nordugrid.org/

[59] http://www.tcs.hut.fi/˜aehyvari/gridjm/

[60] M. W. Vannier, E. V. Staab, and L. C. Clarke. Medical image archives – present and future. In H. U. Lemke, M. W. Vannier, K. Inamura, A. G. Farman, and J. H. C. Reiber, editors, Proceedings of the International Conference on Computer–Assited Radiology and Surgery (CARS 2002), pages 565–576, Paris, France, June 2002.

[61] M. Costa Oliveira, W. Cirne, and P. M. de Azevedo Marques. Towards applying content–based image retrieval in clinical routine. Future Generation Computer Systems, 23:466–474, 2007.

[62] H. J. Lowe, I. Antipov, W. Hersh, and C. Arnott Smith. Towards knowledge–based retrieval of medical images. The role of semantic indexing, image content representation and knowledge–based retrieval. In Proceedings of the Annual Symposium of the American Society for Medical Informatics (AMIA), pages 882–886, Nashville, TN, USA, October 1998.

[63] H. D. Tagare, C. Ja□e, and J. Duncan. Medical image databases: A content–based retrieval approach. Journal of the American Medical Informatics Association, 4(3):184–198, 1997.

[64] H. M¨uller, M. Pitkanen, X. Zhou, A. Depeursinge, J. Iavindrasana, and A. Geissbuhler. Knowarc: Enabling grid networks for the biomedical research community. In Healthgrid 2007, pages 261–268, Geneva, Switzerland, April 2007.

[65] M. Litzkov, M. Livny, and M. Mutka. Condor — a hunter of idle workstations. In Proceedings of the 8th international conference on distributed computing, pages 104–111, San Jose, California, USA, June 1988.

[66] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. The International Journal of Supercomputer Applications and High Performance Computing, 11(2):115–128, Summer 1997.

[67] M. Romberg. The unicore grid infrastructure. Scientific Programming, 10(2):149–157, 2002.

[68] M. Ellert, M. Grønager, A. Konstantinov, B. K´onya, J. Lindemann, I. Livenson, J. Langgaard Nielsen, M. Niinim¨aki, O. Smirnova, and A. W¨a¨an¨anen. Advanced resource connector middleware for lightweight computational grids. Future Generation computer systems, 23(2):219–240, 2007.

[69] S. G. Erberich, J. C. Silverstein, A. Chervenak, R. Schuler, M. D. Nelson, and C. Kesselman. Globus medicus – federation of dicom medical imaging devices into healthcare grids. In Healthgrid 2007, pages 269–278, Geneva, Switzerland, April 2007.

[70] I. Blanquer, V. Hernandez, D. Segrelles, and E. Torres. Trencadis – secure architecture to share and manage dicom objects in a ontological framework based on ogsa. In Healthgrid 2007, pages 115–124, Geneva, Switzerland, April 2007.

[71] H. M¨uller, A. Garcia, J.-P. Vall´ee, and A. Geissbuhler. Grid computing at the university hospitals of geneva. In Proceedings of the 1st healthgrid conference, pages 264–276, Lyon, France, January 2003.

[72] H. M¨uller, M. Pitkanen, X. Zhou, A. Depeursinge, J. Iavindrasana, and A. Geissbuhler. Knowarc: Enabling grid networks for the biomedical research community. In Healthgrid 2007, pages 261–268, Geneva, Switzerland, April 2007.

[73] J. Montagnat, V. Breton, and I. E. Magnin. Partitioning medical image databases for content–based queries on a grid. International Journal of Supercomputer Applications, 44(2):154–160, 2005.

[74] Xin Zhou, Mikko Juhani Pitkanen, Adrien Depeursinge, Henning Müller, A Medical Image Retrieval Application Using Grid Technologies To Speed Up Feature Extraction, ICT4Health, Manila, Phillipines, 2008.

[75] http://www.vmware.com/

[76] http://www.gnu.org/software/gift/

[77] http://www.imageclef.org/

[78] http://www.tcs.hut.fi/˜aehyvari/gridjm/

[79] Montagnat, F. Bellet, H. Benoit-Cattin, V. Breton, L. Brunie, H Duque, Y. Legre, I. Magnin, L. Maigne, S. Miguet, J. Pierson, L Seitz, T Tweed.Medical images simulation, storage and processing on the European DataGrid testbed. J Journal of Grid Computing 4(2):387-400, Springer Verlag, ISSN 1570-7873. 2004.

[80] EDG (2001). European DataGrid IST project, FP5, jan. 2001-feb. 2004, http://www.edg.org/.

[81] Cécile Germain, Vincent Breton, Patrick Clarysse, Bertrand Delhay, Yann Gaudeau, Tristan Glatard, Emmanuel Jeannot, Yannick Legré, Johan Montagnat, Jean-Marie Moureaux, Angel Osorio, Xavier Pennec, Joël Schaerer, Romain Texier. "Grid Analysis of Radiological Data" in Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare, chapter XIX, IGI edition, 2009

[82] C. Germain, V. Breton, P. Clarysse, Y. Gaudeau, T. Glatard, E. Jeannot, Y. Legré, C. Loomis, J. Montagnat, J-M Moureaux, A. Osorio, X. Pennec, R. Texier. Grid-enabling medical image analysis. CCGrid 2005 Bio-Grid workshop, IEEE Press. Extended version to appear in Journal of Clinical Monitoring and Computing.

[83] www.aci-agir.org

[84] Argonne National Laboratory, MPICH2, http://www.mcs.anl.gov/mpi/

[85] William Gropp and Ewing Lusk. An abstract device definition to support the implementation of a high-level point-to-point message-passing interface. Preprint MCS-P342-1193, Argonne National Laboratory, 1994.

[86] D. BUNTINAS, G. MERCIER et W. GROPP. « Implementation and Shared-Memory Evaluation of MPICH2 over the Nemesis Communication Subsystem ». Dans Recent Advances in Parallel Virtual Machine and Message Passing Interface : Proc. 13th European PVM/MPI Users Group Meeting, Bonn, Germany, septembre 2006.

[87] N. Balaji, N. Ramaraj and S. Sadaiappan Enhancing Computational Spped for Search Application Through High Performance Grid Using Globus Tool Kit International Journal of Computational Intelligence Research ISSN 0973-1873 Volume 5, Number 1 (2009), pp. 45–56

[88] Jens Hofmann, Norbert Th. M¨uller, and Kasyap Natarajan. Parallel versus Sequential Task-Processing : A New PerformanceModel in Discrete Time. Universit¨at Trier, Mathematik/Informatik, Forschungsbericht, 96-46, 1996.

[89] http://wang.ist.psu.edu/

[90] http://wiki.cogkit.org/

[91] http://www.worldcommunitygrid.org/

[92] http://lhc.web.cern.ch/lhc/

[93] http://www.eurogrid.org/wp2.html

[94] http://www.corba.org/

[95] http://www.jini.org/