

INTRODUCTION GENERALE	5
CHAPITRE I: LES GRILLES INFORMATIQUES.....	7
I.1 Problématique des grilles de calcul	8
I.1.1 Les besoins en calcul	8
I.1.2 Les besoins en stockage.....	9
I.1.3 Les besoins en bande passante	9
I.1.4 Les solutions retenues	10
I.1.5 Les inconvénients des supercalculateurs	12
I.2 Introduction aux grilles informatiques	14
I.2.1 Les caractéristiques des grilles informatiques	14
I.2.2 Les apports des grilles par rapport aux autres architectures	15
I.2.3 Les différents types de grilles informatiques	16
I.2.4 Cadre d'utilisation des grilles de calcul.....	18
I.2.5 Architecture d'une grille de calcul.....	20
I.2.6 Les interlogiciels de grille de calcul	23
I.3 Les limites des grilles de calcul	24
Conclusion	24
CHAPITRE II: ETUDE DE GLOBUS ET MISE EN PLACE DE LA GRILLE.....	25
II.1 Introduction	26
II.2 Motivation et concept.....	27
II.3 Les innovations de la version 4 de Globus.....	28
II.3.1 Les services web.....	28
II.3.2 Les nouveaux composants	29
II.4 Architecture de Globus version 4.....	29
II.4.1 Le service de sécurité	30
II.4.2 La gestion des données	32
II.4.3 La gestion des ressources	32
II.4.4 Le service d'information.....	33
II.5 Les limites de Globus.....	35
II.5.1 La sécurité.....	35
II.5.2 L'utilisation de Globus.....	35
II.6 Mise en place de la grille de calcul	35
II.6.1 Préparation à l'installation de Globus 4.0.1	35
II.6.2 Installation de l'interlogiciel Globus 4.0.1	38
II.7 Mise en place du site de Dakar dans le cadre du projet HP-UNESCO	40
II.7.1 Préparation à l'installation de l'interlogiciel.....	41
Conclusion	42
CHAPITRE III: SYSTEMES DE TRANSFERT DE DONNEES ET PROBLEMATIQUE DE RECHERCHE.....	43
III.1 INTRODUCTION	44
III.2 Les systèmes de transfert de fichiers.....	44
III.2.1 La structure des fichiers	44
III.2.2 LES MODES DE TRANSFERT DE FICHIERS	46

III.2.3	Quelques protocoles de transfert de fichier	47
III.3	Performances dans les systèmes de transfert	52
III.3.1	Les techniques de mesure de performance dans les systèmes de transfert	52
III.3.2	Les métriques de performance dans les systèmes de transfert	52
III.4	Problématique de recherche	55
III.4.1	Constat	55
III.4.2	Apport de notre mémoire	56
	Conclusion	57
	CHAPITRE IV: ETUDE DE PERFORMANCES ET PROPOSITION.....	58
IV.1	Les performances du Gridftp	59
IV.1.1	Les conditions des tests de performances	59
IV.1.2	Gridftp vs ftp	59
IV.1.3	Impact du parallélisme sur le transfert	60
IV.1.4	Impact de la taille des blocs	61
IV.1.5	Impact de la taille des buffers TCP	62
IV.1.6	Impact de la combinaison sur les performances	63
IV.2	Proposition d'optimisation du temps de transfert des fichiers.....	65
IV.2.1	Les problèmes du Gridftp	65
IV.2.2	Identification du problème par test	66
IV.2.3	Description de la proposition	70
	CONCLUSION ET PERSPECTIVES.....	76
	REFERENCES.....	78
	Liste des figures.....	80
	Liste des tableaux.....	80

INTRODUCTION GENERALE

Aujourd'hui les applications informatiques deviennent de plus en plus exigeantes en puissances de calcul, en capacités de stockage et aussi en bande passante. Face aux réseaux classiques ayant atteint leurs limites, il s'avère donc nécessaire de passer à une autre forme de collaboration entre les centres de recherche, les entreprises, d'où la naissance des grilles informatiques. Cette collaboration s'effectue par la mise en place d'un système virtuel composé d'un grand nombre de systèmes interconnectés et cela de manière complètement transparente. Ainsi deux éléments sont indispensables à l'apparition des grilles : l'abondance des ressources (de calcul et de stockage) et leur interconnexion.

Il est important de noter qu'au delà des besoins des capacités de traitement l'approche grille permet la co-allocation et la partage dynamique de toutes les ressources de systèmes hétérogènes indépendamment des unités organisationnelles. Ainsi la solution mise en place s'appuie sur la constitution d'organisations virtuelles (VO) regroupant des communautés d'utilisateurs partageant des ressources globalisées de traitement, de stockage, des instruments et des services largement distribués, quelles qu'en soient les organisations propriétaires.

Cependant, le modèle de grille diffère des technologies de cluster qui sont non seulement limités géographiquement mais aussi se basent sur un système de fichier de type NFS qui n'est pas adapté à des communications à grande échelle pour des raisons de sécurité. Le système de grille permet par contre à des machines appelées nœuds de joindre l'espace virtuel sur des réseaux à grande échelle (de type WAN). Ceci implique des contraintes de sécurité, de gestion de données et de tâches, de découverte et de surveillance de ressources etc. D'où la nécessité d'avoir un dispositif pour assurer ces services. Ces fonctions sont réalisées par un inter logiciel (tel que Globus).

D'autre part l'accès aux données occupe une place primordiale dans les systèmes de grilles. Les applications scientifiques nécessitent des transferts de données avec des volumes très importantes (térabits ou petabits).

Des protocoles de transfert et d'accès aux données ont été proposés. Cependant ils peuvent présenter quelques limites dues, d'une part à un mauvais dimensionnement des paramètres du protocole de transfert et d'autre part à une manque d'optimisation des fonctionnalités des services offerts par les protocoles.

L'objectif de ce mémoire est l'analyse des apports des interlogiciels et des services de grilles pour le calcul haute performance. Il comprend un important volet pratique, qui consiste à l'évaluation et la prise en charge de la plate-forme Globus (version 4, GT4), le principal standard d'interlogiciel pour les grilles informatiques et leurs applications. Cette pratique sera complétée, dans une seconde partie du travail, par l'étude et l'exécution d'un des principaux services pour les grilles proposés par l'interlogiciel Globus. Dans une seconde partie du travail, nous proposons l'étude et la réalisation des exécutions du service Gridftp (protocole de transfert de fichiers dans les grilles) entre un client et un serveur Globus, pour mettre en lumière les apports de ce service par rapport au service équivalent FTP, utilisé dans les environnements, systèmes et réseaux IP standards. La dernière partie du mémoire est consacrée à une amélioration des fonctionnalités de Gridftp.

Ainsi dans le chapitre I nous proposons un état de l'art sur les grilles informatiques. Le chapitre II est consacré à l'étude et la mise en place de l'interlogiciel Globus. Dans le chapitre III nous présentons la problématique de recherche liée aux systèmes de transfert dans les systèmes de grille. Nous terminerons ce mémoire, au chapitre IV par une étude de performance du protocole Gridftp et une proposition d'optimisation des temps de transfert.

Chapitre I: LES GRILLES

INFORMATIQUES

I.1 Problématique des grilles de calcul

Dans les années 70, on assiste à l'apparition de la simulation numérique par le calcul intensif ou HPC pour *High Performance Computing* [1]. Elle désigne un procédé selon lequel on exécute un programme informatique sur un ordinateur en vue de simuler un phénomène réel. Dès lors, le calcul intensif a pris une place essentielle dans la plupart des domaines de la recherche et de l'industrie. Il est indispensable au développement de la simulation numérique et de l'exploitation des données, outils dorénavant essentiels pour la recherche scientifique d'une part et d'autre pour l'industrie.

En effet, la maîtrise de la simulation et de l'exploitation des données conditionne le progrès dans de nombreux domaines scientifiques dont les matériaux, la chimie moléculaire, les sciences de la terre et du climat, l'énergie, la biologie, la bioinformatique, la physique, l'astrophysique, la mécanique des fluides, la combustion, les nano sciences et les nano technologies.

Dans l'industrie, il est nécessaire d'exploiter des modèles sophistiqués pour la conception et l'optimisation de produits et de procédés dans de nombreux domaines. Enfin, l'analyse de volumes considérables de données est souvent requise, c'est le cas en recherche pétrolière.

L'augmentation perpétuelle de la quantité de données à traiter, due à la convergence des technologies de l'information, tend à rendre l'utilisation d'ordinateur classique obsolète.

Dés lors, certains besoins liés à ces recherches s'imposent et parmi lesquels on peut citer : les besoins en calcul, les besoins en stockage les besoins en bande passante.

I.1.1 Les besoins en calcul

De plus en plus, on veut fournir des résultats scientifiques dans les meilleurs délais possibles. Cela pose évidemment des problèmes de localisation des données, de temps d'accès aux ressources ainsi que de temps de traitement associé. Si l'on multiplie ce problème à un groupe de travail, à une entreprise, à un pays, voire même à tout l'Internet on obtient très rapidement des volumes de traitement très grands [2]. Il est

donc impossible de se reposer sur une architecture classique pour répondre à ces besoins. D'autant plus que les besoins en puissance de calcul s'imposent de plus en plus. D'où la nécessité de trouver des solutions pour résoudre certains problèmes scientifiques.

L'expérience du Large Hadron Collider en est une parfaite illustration. Elle consiste à calculer l'énergie dégagée lors des chocs de protons se déplaçant à la vitesse de la lumière [1]. Dès lors question est de savoir comment disposer de la puissance de calcul afin d'analyser ces données. La réponse à cette question est la mise en place d'une grille de calcul.

I.1.2 Les besoins en stockage

Dans certaines applications, tant dans le domaine de la recherche que dans le domaine de l'industrie, ce sont les données qui structurent la démarche par la gestion de volumes considérables, par l'exploration de ceux-ci, par les problèmes complexes qu'il faut résoudre pour en extraire les mesures exploitables, ou enfin par les traitements statistiques auxquels il faut les soumettre [2].

Les données collectées doivent être archivées et stockées pour être ensuite analysées dans le cadre de la recherche. Ces données sont nombreuses et complexes. Même avec la technologie actuelle, une machine unique n'est pas assez puissante pour analyser et stocker toutes les données. Donc, la nécessité de mettre en place des infrastructures se dotant de grands systèmes de stockage de données s'impose [1].

I.1.3 Les besoins en bande passante

Devant les très gros volumes de données à traiter, à stocker mais surtout à acheminer à travers des espaces géographiques distribués les équipements réseaux ne pouvaient plus offrir la bande passante nécessaire. Donc, il s'avère nécessaire de mettre en place de nouvelles infrastructures pour répondre à ces besoins en bande passante.

Le projet ERS (Européen Remote Sensing Satellite System) de l'agence spatiale Européenne (ESA) est un des plus importants programmes d'observation de la terre concernant la climatologie, l'océanographie ou la gestion des ressources ERS. Il

consiste en l'envoi de deux satellites qui doivent envoyer environ 100 Gbytes de données par jour pour la mission (ERS ½) avec une vitesse de transmission de supérieure à 200Mbytes/s [1].D'où la nécessité de bande passante.

I.1.4 Les solutions retenues

Devant ces besoins en stockage, en calcul et en bande passante diverses solutions ont été adoptées parmi lesquelles l'informatique parallèle et l'informatique distribuée. Ainsi, avec l'avènement de l'informatique parallèle et distribuée on assiste aussi à l'avènement de nouveaux types d'ordinateurs les clusters (architecture distribuée) et les multiprocesseurs (architecture parallèle). On utilisera le terme supercalculateur pour désigner ces types d'ordinateurs. Ces supercalculateurs ont permis de résoudre plusieurs problèmes liés :

- Au besoin de calcul : la mise en commun de plusieurs unités de calcul permet d'effectuer des calculs parallèles en des temps plus courts. De ce fait on acquiert des puissances de calcul énormes,
- Au besoin de stockage : la mise en commun de plusieurs unités de stockage et l'accès distant permettent de stocker de très grands volumes de données et d'y accéder en des temps très rapides.

I.1.4.1 L'informatique parallèle ou le parallélisme

Appelée aussi calcul parallèle, elle consiste en l'exécution simultanée d'une même tâche, partitionnée et adaptée afin de pouvoir être répartie entre plusieurs processeurs en vue de traiter plus rapidement des problèmes plus grands. Le terme ordinateur parallèle ou multiprocesseur est parfois utilisé en référence à un ordinateur qui possède plus d'une unité de traitement et qui peut effectuer un traitement parallèle.

I.1.4.1.A Les architectures parallèles

La principale architecture parallèle est le multiprocesseur. Il est caractérisé par plusieurs unités de traitement qui ont une mémoire commune. On parle ainsi

d'architecture à mémoire partagée. Pour communiquer, les processeurs utilisent différentes méthodes parmi lesquelles on peut citer :

- le sémaphore,
- l'exclusion mutuelle.

Blue Gene/L - eServer Blue Gene est un exemple de multiprocesseur. Il comporte près de 131072 processeurs et a une puissance réelle de 280600 flops [15].

I.1.4.1.B Les types de multiprocesseur

Il existe plusieurs types d'ordinateurs (ou de processeurs) parallèles, caractérisés, principalement, par différents modèles d'interconnexions entre les processeurs et la mémoire. La classification la plus populaire est la taxonomie de Flynn [19], qui catégorise les ordinateurs parallèles selon le type d'organisation du flot de données et du flot d'instructions. Elle distingue différents types de machines :

- SISD (single instruction single data): à flot d'instructions et de données unique,
- SIMD (single instruction multiple data) : à flot d'instructions uniques agissant sur des données multiples,
- MIMD (multiple instruction multiple data) : à flot d'instructions et de données multiples,
- MISD (multiple instruction single data): à flot d'instruction multiples agissant sur des données uniques.

I.1.4.2 L'informatique distribuée

La deuxième solution trouvée pour résoudre ces besoins en stockage, en calcul et en bande passante est l'informatique distribuée appelée système distribué ou calcul distribué.

Un système distribué ou repartitionné est composé de plusieurs systèmes calculatoires autonomes sans mémoire physique commune qui communiquent par l'intermédiaire d'un réseau.

Les principales architectures distribuées sont :

- Le cluster : appelé aussi grappe est une collection de machines interconnectées, utilisées comme une ressource de calcul unifiée.

Une grappe « Beowulf » se définit par les propriétés suivantes :

- composants à grande diffusion,
 - composants réseau à faible coût,
 - système d'exploitation « open source »,
 - hardware non propriétaire,
 - logiciel « open source ».
- Le centre de calcul : c'est un environnement distribué pouvant interconnecter des clusters, des multiprocesseurs.

I.1.4.3 Informatique distribuée vs Informatique parallèle

	Mémoire	Processeurs	Communication
Informatique parallèle	Partagée (unique)	Un ou plusieurs	Mutex, Sémaphore
Informatique distribuée	Mémoire distribuée (plusieurs)	Un ou plusieurs	Appel de procédure, envoi de message

Tableau 1. Parallélisme vs distribution

I.1.5 Les inconvénients des supercalculateurs

Les supercalculateurs sont des monstres de puissance, mais ce sont aussi des gouffres à énergie et de véritables nids à chaleur. En effet, l'accroissement de la performance a aussi des conséquences de plus en plus préoccupantes : elle a conduit à une augmentation de la puissance consommée et de la chaleur dissipée. Un microprocesseur de dernière génération peut consommer jusqu'à 130 watts, ce qui se traduit par un dégagement de chaleur considérable. Cet accroissement de la

consommation électrique et donc du dégagement de chaleur ne peuvent se poursuivre faute de dispositif d'évacuation de chaleur.

Aujourd'hui, les chercheurs se trouvent confrontés à de nombreux obstacles liés aux exigences énergétiques des supercalculateurs. Ces exigences ne font qu'augmenter si on se réfère à la loi de Moore [1]. En effet, la Loi de Moore est relative à la croissance des performances de nos ordinateurs. Énoncée il y a plus de trente ans, elle est toujours valable et stipule que : « Le nombre de transistors sur une puce de circuit intégré double tous les dix-huit mois ».

Un autre problème lié à ces supercalculateurs est d'ordre économique. En effet, ils coûtent très chers. Et à cela s'ajoutent des coûts liés à leur maintenance. Donc, il n'est pas facile de s'acquiescer de ces types de machines. Seule les centres de recherche et les grandes industries ont le privilège d'en posséder.

En outre avec les projets actuels et ceux de l'avenir les puissances de calcul deviennent de plus en plus énormes. De plus en plus, un calculateur unique, une grappe d'ordinateurs normaux ou même un supercalculateur spécialisé, ne suffit pas pour les calculs que les scientifiques veulent effectivement faire.

La solution est donc de définir un nouveau concept de collaboration dans lequel il serait possible d'agréger la capacité de plusieurs ressources (ordinateurs, stations de travail, grappes, matériels spécifiques) afin de réaliser un seul et même service.

En plus, ces ressources doivent pouvoir être éparpillées sur des espaces géographiques différents et appartenir à des institutions différentes. C'est cet «agrégat » qu'on appelle **grille informatique**.

Le concept de grilles informatiques a fait ses preuves, des grilles de production sont maintenant opérationnelles en entreprises et entre organismes. Au delà d'une solution pour les calculs intensifs, l'approche grille permet la constitution d'organisations virtuelles (VO) regroupant des communautés d'utilisateurs partageant des ressources globalisées de traitement, de stockage, des instruments et des services largement distribuées, quelles qu'en soient les organisations propriétaires.

Le concept se veut une sur extension de l'informatique répartie et parallèle en allant au delà des performances et des services offerts par ces technologies. Donc, il est nécessaire d'adapter de nouvelles approches sur le plan algorithmique et applicatif afin

de relever les défis de grilles dont les principaux sont la performance et les services offerts d'où l'utilisation d'un ensemble de logiciels appelés interlogiciel. Ce dernier unifie l'accès à des ressources informatiques hétérogènes.

I.2 Introduction aux grilles informatiques

Le paradigme de grilles informatiques a été posé en 1998 dans [2]. C'est une notion qui s'inspire beaucoup de la grille d'électricité (*power Grid*).

Par analogie à cette grille d'électricité, la notion de grille informatique est définie comme étant : une infrastructure matérielle et logicielle fournissant un accès fiable, cohérent à un taux de pénétration élevé et bon marché à des capacités de traitement, de calcul et de stockage.

Mais au delà des besoins en capacité de traitement, l'approche grille permet la co-allocation et le partage dynamique de toutes les ressources de systèmes hétérogènes indépendamment des unités organisationnelles (départements d'entreprises, institutions et entreprises). Des organisations virtuelles (VO) permettent maintenant de définir en tant que de besoin des communautés d'utilisateurs partageant suivant des modalités communes d'utilisation, des sous-ensembles de ces ressources globalisées de traitement, de stockage, des logiciels, des instruments et des services largement distribuées, quelles que soient les organisations propriétaires.

I.2.1 Les caractéristiques des grilles informatiques

Les grilles informatiques possèdent quatre principales caractéristiques qui sont :

- existence de plusieurs domaines administratifs : Les ressources appartiennent à des organisations qui ont des localisations différentes. En outre, chaque organisation a sa propre politique d'administration (gestion et sécurité) de ses ressources,
- hétérogénéité des ressources : Les ressources dans une grille sont de natures différentes. Cette différence de nature peut être matérielle comme logicielle,
- nombre important de ressources : une grille informatique peut aller de quelques dizaines de ressources jusqu'à des dizaines de millions de ressources,

- nature dynamique des ressources_: les ressources dans grilles peuvent varier en tout temps.

I.2.2 Les apports des grilles par rapport aux autres architectures

Il faut noter que les grilles informatiques reposent principalement sur les architectures actuelles (informatiques distribuée et parallèle). Cependant, les grilles informatiques offrent des avantages que les technologies actuelles ne sont pas en mesure d'assurer. Parmi ces avantages on peut citer :

- L'exploitation des ressources inutilisées : des études ont montré que les postes de travail et les ordinateurs de bureau sont inactifs la plupart du temps. En effet, les pourcentages d'utilisation indiquent un taux d'utilisation de 5% pour les machines grand public et 30% pour les machines en milieu académique et industriel. Ainsi, les grilles de calcul permettront d'utiliser les cycles processeurs (temps mis entre deux tops d'horloges) pendant lesquelles les machines sont inactives pour faire tourner des applications exigeantes en matière de puissance dont les machines dédiées ne parviennent pas à assurer.
- La fourniture d'une importante capacité de calcul : l'une des caractéristiques principales des grilles de calcul est la possibilité de fournir de grandes puissances de calcul. En effet la technologie du ***Grid Computing*** s'inspire beaucoup de l'informatique parallèle et distribuée. Plus précisément avec la révolution de l'algorithmique les applications sont écrites d'une façon à pouvoir exploiter parallèlement des ressources (clusters, machines multiprocesseur, matériels spécifiques, etc.). Les grilles de calcul peuvent de la même manière fournir des ressources dont l'utilisation pourra se faire en parallèle.
- La virtualisation de certaines ressources : la virtualisation est l'ensemble des techniques matérielles et/ou logicielles qui permettent de faire fonctionner sur une seule machine plusieurs systèmes d'exploitation et/ou plusieurs applications, séparément les uns des autres, comme s'ils fonctionnaient sur des machines physiques distinctes. En virtualisant les ressources, une grille pourra fournir l'accès à des ressources spéciales comme des équipements spécifiques (microscope électronique,

capteur ...) ou des logiciels dont le prix de la licence est élevé. Ainsi ces ressources exposées à tous les utilisateurs seront mieux utilisées et partagées et ainsi on évitera d'avoir recours à installer du nouveau matériel ou acheter de nouvelles licences.

- La fiabilité et disponibilité des ressources : les ressources de la grille appartiennent à des organisations virtuelles dispersées géographiquement. De ce fait la dispersion des ressources et aussi leur disponibilité en grande quantité permettent à la grille d'assurer la continuité du service si certaines ressources deviennent inaccessibles. Les logiciels de contrôle et de gestion de la grille seront en mesure de soumettre la demande de calcul à d'autres ressources [12].

I.2.3 Les différents types de grilles informatiques

Selon les besoins et les résultats attendus on peut classer les grilles informatiques en : grille d'information, grille de données et grille de calcul.

I.2.3.1 La grille d'information

Ce type de grille est conçu pour le partage de connaissances qui sont partagées à travers un réseau qui permet aux utilisateurs d'y accéder. La grille doit aussi permettre la recherche d'information de façon efficace et fiable grâce aux moteurs de recherche. Une des applications est le WEB. L'accès à l'information se fait à partir d'une adresse http ou à partir d'un moteur de recherche (Google par exemple).

I.2.3.2 La grille de données

On l'appelle aussi grille de stockage (Data Grid). Elle est conçue pour le stockage des données. Elle permet de découper les données et de les ranger dans plusieurs machines.

En outre, elle doit mettre en place des mécanismes de recherche, d'indexation, de sécurité pour permettre un accès fiable et permanent. Il est important que la répartition des données soit transparente pour l'utilisateur.

En effet, il est impossible de savoir précisément où va être stockée chaque donnée. Un fichier peut avoir été découpé en plusieurs morceaux et réparti sur plusieurs machines. Il peut aussi avoir été répliqué sur plusieurs entités du réseau pour permettre une meilleure disponibilité. Il faudra alors avoir un mécanisme qui répartit la charge sur chaque serveur.

Une des applications des grilles de données est le cas Napster. C'est une grille de stockage à cheval entre les modes de déploiement P2P et le client/Serveur.

En effet il existe un serveur Napster qui stocke des données mais également des clients Napster qui stockent des données.

I.2.3.3 Les grilles de calcul

Ce type de grille va essayer quand à elle de résoudre le problème du manque de puissance de calcul afin de permettre de réaliser des opérations compliquées dans des délais raisonnables. Il faut noter que les calculs peuvent bel et bien se faire sur une seule et même machine. Cependant, les durées de traitement risquent d'être énormes (des années pour faire la cryptanalyse de certaines clés). Donc, la grille de calcul permet de fédérer plusieurs machines et de répartir les calculs sur les machines. Mais cette répartition est régulée par plusieurs concepts de construction de la grille.

Dans la suite de notre étude on se focalisera uniquement sur les grilles de calcul qui constituent les grilles les mieux adaptées pour les mesures de performance.

I.2.4 Cadre d'utilisation des grilles de calcul

Les grilles de calcul offrent plusieurs possibilités d'application tant les domaines pouvant bénéficier d'une telle puissance et infrastructure. Dans cette partie nous allons présenter quelques domaines d'application de ces grilles de calcul.

I.2.4.1 Le calcul distribué

Les applications de calcul distribuées sont d'excellentes candidates pour être utilisées dans les grilles de calcul. Grâce à un nombre très important de ressources de calcul elles peuvent résoudre des problèmes qui leur étaient inaccessibles auparavant.

Ainsi, les expériences du LHC nécessiteront de l'ordre de 100 000 processeurs pour traiter la masse d'informations générées par le collisionneur. Il est évidemment illusoire d'espérer ou de disposer d'une telle puissance de calcul sur un seul et même site [1].

L'utilisation d'une grille se révèle dès lors indispensable. Parmi les principaux défis que doit relever l'architecture de la grille pour de telles applications, citons : l'ordonnancement à grande échelle des processus, la souplesse des algorithmes et protocoles qui doivent être capables de gérer un nombre de nœuds pouvant aller de la dizaine à des centaines voire des milliers de fermes de machines, la tolérance des algorithmes aux temps de latence inhérents à la taille de la grille, atteindre et maintenir un haut niveau de performances dans un système très hétérogène.

I.2.4.2 Supercalculateur virtuel ou High-Throughput Computing

Le but ici n'est pas de rechercher la performance pure comme dans le cas précédent mais de rentabiliser au maximum des ressources en récupérant les cycles processeurs non utilisés quand celui-ci est oisif (idle). On pense bien entendu aux stations de travail qui sont généralement très largement sous utilisées, ne serait-ce qu'en dehors des heures de bureau.

I.2.4.3 Le calcul à la demande

Ce type d'applications utilise la grille afin de satisfaire des besoins à court terme en ressources, tels qu'ils ne peuvent être satisfaits en local pour des raisons pratiques ou de rentabilité. Ces ressources peuvent être du temps de calcul, des logiciels, des données, des capteurs spécialisés, etc.

Contrairement au calcul distribué, le but recherché ici est plutôt la rentabilité que les performances pures.

Pour rendre cette nouvelle forme d'utilisation des ressources possible, il est primordial de développer des systèmes performants et sécurisés de facturation (établir le montant à imputer à l'utilisateur en fonction des différentes ressources qu'il a employées sur la grille) et de paiement.

I.2.4.4 Traitement massif de données

Dans ce type d'applications, le but est d'extraire de nouvelles informations à partir de grandes bases de données distribuées géographiquement. Généralement, ces types de traitement sont également de grands consommateurs de puissance de calcul et de bande passante.

L'expérience du LHC est sans doute la meilleure illustration de ce type de problème. En effet il est indispensable de faire la distribution géographique des données (15 péta octets).

I.2.4.5 L'informatique collaborative

Le but des applications collaboratives est de permettre et favoriser les interactions entre les personnes. Elles sont souvent structurées sous forme d'espaces virtuels partagés entre les utilisateurs. La plupart de ces applications permettent de partager des ressources comme par exemple des données ou des simulations. Elles partagent alors bien souvent des caractéristiques avec les autres grands types d'applications décrites précédemment. Les expériences du LHC sont ici encore une bonne illustration de ce type d'applications.

En effet, ce n'est pas moins de quelque 5000 chercheurs repartis dans plus de 500 universités et instituts qui devront pouvoir accéder aux résultats des différentes expériences [12].

I.2.5 Architecture d'une grille de calcul

La quantité importante de données à traiter et stocker, mais surtout la nature dynamique des ressources au niveau de la grille, montre que l'interopérabilité est un facteur très important que la grille doit tenir en compte. L'architecture de la grille, en plus de définir les composants fondamentaux de la grille et leurs fonctions, met aussi en place des mécanismes pour assurer cette interopérabilité. Ces mécanismes sont entre autres des protocoles, des services, des APIs. De ce fait, dans une architecture de grille on trouve différentes couches permettant :

- l'intégration des différents composants matériels et logiciels en une ressource globale à travers le réseau,
- l'implémentation d'interlogiciel offrant une vue transparente et consistante à cette ressource,
- le développement d'outils permettant le contrôle et la gestion de l'infrastructure et des applications,
- le développement d'applications exploitant cette infrastructure.

Ainsi, l'architecture d'une grille de calcul comprend trois couches (voir figure 1).

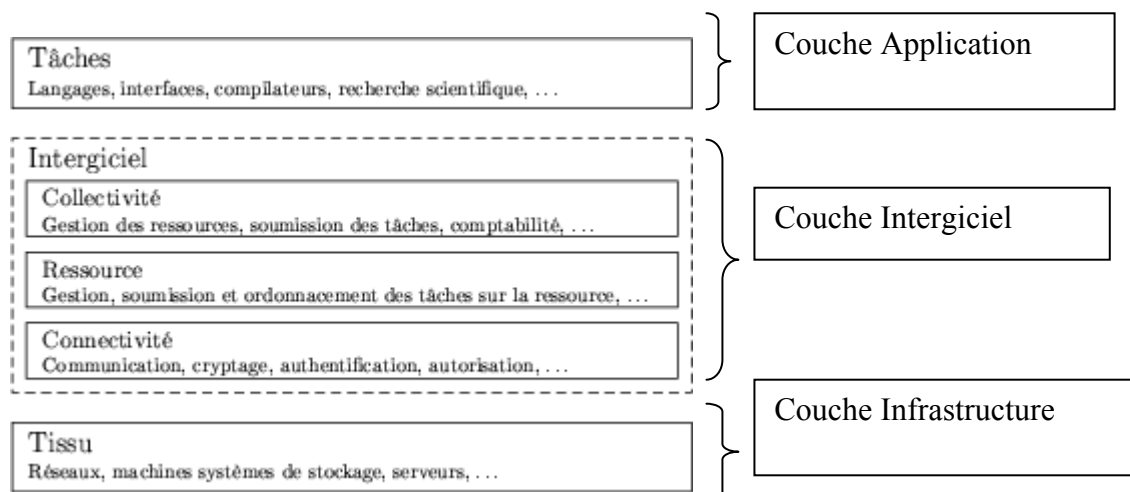


Figure 1. Architecture d'une grille

I.2.5.1 La couche infrastructure

Cette couche concerne les ressources proprement dites interconnectées à travers des réseaux locaux et grande distance. Cette infrastructure matérielle (fabric) comprend notamment des PC, des stations de travail, des grappes de calcul (clusters), des équipements de stockage, des bases de données, des équipements réseaux, des équipements spéciaux, etc. Les ressources d'une grille peuvent varier des plus simples (machines de même architecture, même système d'exploitation, réseau d'interconnexion homogène) au plus compliquées (différentes architectures de machines, plusieurs systèmes d'exploitation, étendue géographique, différentes politiques de sécurité, plusieurs administrateurs, etc.).

La couche infrastructure permet aux utilisateurs de se renseigner sur la structure, l'état, la capacité des ressources mais aussi de se munir de mécanismes de gestion de ces ressources.

I.2.5.2 La couche interlogiciel

La couche interlogiciel permet de gérer les ressources de la grille. Elle se charge de la sécurité, de la soumission de tâches, de la gestion des données et des informations. Donc, elle permet à l'utilisateur d'utiliser ces ressources la grille sans se préoccuper de l'architecture matérielle et logicielle de celles-ci. La couche interlogiciel est composée de trois couches.

I.2.5.2.A La couche connectivité : communication et sécurité

La couche connectivité définit des protocoles de communication et d'authentification pour des transactions entre les ressources de la couche infrastructure. Plus précisément, les protocoles de transmission permettent l'échange des données entre les ressources alors que les protocoles d'authentification fournissent des mécanismes de sécurité cryptographiques pour vérifier l'identité des utilisateurs et des ressources. La communication nécessite par ailleurs le transport, l'adressage et le routage. Par conséquent, la couche connectivité fait appel certaines couches du modèle TCP/IP et plus précisément les couches Internet (IP, ICMP), transport (TCP, UDP).

I.2.5.2.B La Couche Ressource

La couche Ressource utilise les services des deux couches précédentes (fabrique et connectivité) pour collecter des informations sur les caractéristiques des ressources, les surveiller, les contrôler et gérer une éventuelle facturation quant à leur utilisation. Pour ce faire, elle utilise la couche fabrique qui lui fournit les fonctions nécessaires au contrôle et à la collecte d'informations. Les différentes fonctionnalités que nous avons présentées dans la section 3.4.1 constituent donc les opérations qui doivent pouvoir être effectuées à partir de la couche Ressource.

Les protocoles définis dans la couche Ressource ne concernent les ressources que d'un point de vue individuel. Elle ne s'intéressera donc qu'aux caractéristiques intrinsèques des ressources et à la façon dont celles-ci se comportent. Les tâches relatives à leur partage proprement dit incombent à la couche Collectif, présentée dans la section suivante.

I.2.5.2.C La Couche Collectif

Puisque la couche ressource s'occupe des ressources de façon individuelle il est nécessaire de mettre en place une couche de plus haut niveau afin de traiter les ressources de façon collective. C'est le travail de la couche collectif. Les protocoles et services décrits dans cette couche permettent de voir les ressources de façon globale et de décrire les interactions entre elles. Donc, elle se charge de l'ordonnancement et de la co-allocation des ressources lorsqu'un utilisateur fait appel à plusieurs ressources en même temps. Elle se charge aussi de la réplication des données. Enfin, elle se charge de la récupération et de la gestion des alarmes de monitoring.

I.2.5.3 La couche application

Enfin, au sommet du modèle, on retrouve les applications qui seront exécutées sur la grille afin d'y récupérer des données ou effectuer des calculs. Les différentes couches seront alors sollicitées. Les couches Collectif et Ressource seront en charge de la recherche des ressources pouvant héberger l'exécution de l'application. Une fois celles-ci trouvées, la couche Connectivité sera utilisée pour s'authentifier et, enfin, la

couche Fabrique sera en charge de l'accès à proprement parler aux ressources. Parmi ces applications, on peut citer entre autre des projets scientifiques, médicaux, financiers, ingénierie, etc.

I.2.6 Les interlogiciels de grille de calcul

Le moteur des grilles de calcul est le l'interlogiciel. Le rôle de ces interlogiciels est de fédérer un ensemble de machines, sous une seule et même identité. Aujourd'hui, il existe plusieurs interlogiciels pour la mise en œuvre d'une grille de calcul. Parmi ces interlogiciels, on peut citer Legion, Globus, Glite.

I.2.6.1 Legion

L'interlogiciel sert d'interface entre le propre OS de l'utilisateur, et un nombre quasi infini de ressources distribuées partout sur le réseau, et hébergées par les autres utilisateurs de Legion. Chaque utilisateur a donc l'impression de ne "voir" que son propre ordinateur, mais fait en réalité appel à de multiples "ressources" répartis sur le réseau. Legion est un système orienté objet, et la notion de "ressources" partagées est à comprendre au sens large (bibliothèques, codes sources, □ fichiers exécutables, etc.), d'autant que les créateurs du système se flattent du fait qu'il soit capable de gérer plusieurs milliers de milliards de ressources, disponibles sur des plates-formes matérielles et logicielles de toutes natures [17].

I.2.6.2 Globus

Globus est formé d'un ensemble de composants. Son architecture modulaire permet d'apporter les modifications et les améliorations d'une manière rapide et efficace. Globus est devenu le standard 'ipso facto' utilisé dans les projets de grilles de calcul. De nombreuses entreprises ont ainsi adopté Globus pour servir comme base de leurs produits commerciaux pour les grilles de calcul.

Dans le prochain chapitre, nous détaillerons les principales fonctionnalités offertes par Globus à ses utilisateurs en termes de sécurité, de services d'information, de gestion des communications, de gestion des ressources et de traitement des données [17].

I.2.6.3 Glite

Cet interlogiciel est entièrement basé sur les web services. Il ressemble beaucoup à Globus dans sa conception et fournit à peu près les mêmes services (gestionnaires de tâches, de données, de service de sécurité et de déploiement). Cet interlogiciel est issu du projet européen EGEE. Il se veut plus avancé que Globus mais est toujours en phase de développement [18].

I.3 Les limites des grilles de calcul

Les grilles de calcul présentent pas mal de problèmes et de limites parmi lesquels on peut citer :

- la sécurité au niveau de la grille : les protocoles de sécurité mis en jeu au niveau de la grille ne sont pas assez suffisants pour assurer tous les services de sécurité telle que l'intégrité des données échangées,
- la confiance entre les participants : c'est un véritable problème de la grille. En effet, les organismes participants ne veulent pas mettre toutes leurs données à la disposition de tout un chacun par, mesure de prudence,
- la complexité de mise en œuvre des algorithmes distribués : en effet, face au nombre important de ressources devant être utilisées en parallèles il s'avère très difficile de mettre en place des algorithmes pour l'exploitation de toutes ces ressources.

Conclusion

Cette partie a été l'occasion de fournir une vue d'ensemble des buts, objectifs, formes et architecture des grilles de calcul. Dans la suite de notre étude nous verrons comment les services que nous venons de décrire dans les différentes couches de l'architecture de la grille de calcul se retrouvent implémentés dans Globus.

Chapitre II: ETUDE DE GLOBUS ET MISE EN PLACE DE LA GRILLE

II.1 Introduction

L'interlogiciel Globus [3] a été développé depuis les années 1990 pour supporter le développement d'applications distribuées et de service pour la conception et la mise en œuvre d'une grille de calcul. Il est principalement développé aux Etats-Unis au sein l'"Argonne National Laboratory" par le groupe de Ian Foster et c'est un projet qui est toujours en cours.

Il est composé d'un ensemble de composants associés dans un cadre commun permettant d'assurer les services de sécurité, l'accès aux données, la gestion des ressources, le transfert des données, la découverte des ressources, etc. Ces composants offrent un ensemble d'outils utilisés pour développer une infrastructure de grille et d'applications distribuées. Nous présentons ici les principales caractéristiques de Globus version 4 basé principalement sur les web services. En d'autres termes, nous présentons les apports de la version 4 par rapport aux versions antérieures et les différents services qu'ils nous offrent.

Globus est :

- une communauté d'utilisateurs et de développeurs qui collaborent dans le cadre du développement d'un interlogiciel libre et gratuit pour la mise en place d'une grille de calcul et mettant une documentation en place pour les organisations virtuelles [6],
- l'interlogiciel est un ensemble de bibliothèques et de programmes permettant de mettre en place des services de systèmes et des applications distribués. Il fournit une variété de composants offrant des possibilités parmi lesquelles on peut citer :
 - un ensemble de services permettant la gestion des ressources, le transfert de données, la découverte de ressources disponibles, etc ;
 - des outils pour la création de nouveaux services web en java, C, Python ;
 - un puissant standard de sécurité pour l'authentification et l'autorisation ;
 - une documentation détaillée de ces composants et leur mode d'utilisation pour créer des applications.

Ces composants engendrent un ensemble d'outils permettant ainsi l'interopérabilité entre plusieurs domaines d'application.

II.2 Motivation et concept

Globus permet de mettre en place des applications pour fédérer des ressources distribuées comme des ordinateurs, des données, des services, des capteurs etc.

Initialement, le travail était motivé par la demande des organisations virtuelles travaillant dans le domaine scientifique. Mais aujourd'hui, les applications commerciales deviennent de plus en plus importantes poussant les entreprises à vouloir mettre en place des grilles de calcul. Cependant, dans les deux cas il est nécessaire d'accélérer les moyens technologiques afin de réduire les problèmes liés à l'accès et leur traitement. La fédération des ressources est entièrement motivée par le besoin d'accéder à des données qu'on ne peut pas répliquer en local. Ainsi, il est important de mettre en place des services permettant de gérer la découverte des ressources, le lancement d'une application, la gestion des données, le contrôle des utilisateurs etc.

Une bonne implémentation de ces services permet de réduire les coûts liés au développement des méthodes et techniques pour une bonne utilisation des données. En plus si toutes ces implémentations s'effectuent dans un cadre standard cela permettra alors d'augmenter l'interopérabilité entre les organisations.

L'interlogiciel Globus a pour objectif de répondre aux besoins liés à ces implémentations en offrant d'abord un modèle libre et gratuit pour permettre la contribution et aussi l'adaptation et ensuite des implémentations standard pour permettre l'interopérabilité.

Aujourd'hui, l'interlogiciel en est à sa version 4. La principale innovation de cette branche 4.0 est l'intégration des services Web. Nous commencerons donc cette section par une présentation succincte de cette technologie. Nous détaillerons ensuite quelques-uns des principaux services qu'offre Globus.

II.3 Les innovations de la version 4 de Globus

II.3.1 Les services web

Avec sa version 4.0, Globus est rentré dans l'ère des services Web [6]. Cette technologie, en pleine expansion, offre une méthode simple, souple et portable d'échanges d'informations entre des applications au travers d'un réseau.

A l'instar de Corba ou RMI, les services Web permettent l'invocation de méthodes ou fonctions situées sur une autre machine que celle qui effectue l'appel. La principale différence avec ces deux technologies est que les services Web utilisent principalement le protocole HTTP pour les transferts réseaux (ils se révèlent donc être particulièrement adaptés pour être utilisés à travers Internet) et se veulent entièrement indépendants de tout langage et de toute plate-forme. Ainsi, on peut imaginer qu'un client écrit en Python et tournant sous Microsoft Windows interroge un serveur de services Web en Java sous GNU/Linux.

Les services Web sont capables de se décrire eux-mêmes. Ainsi, une fois un service localisé, on peut lui demander de se décrire afin de récupérer les différentes opérations possibles et la manière dont on peut les invoquer. Ils assurent l'interopérabilité entre des systèmes pouvant fonctionner sur des environnements très hétérogènes.

Parmi les inconvénients des services web citons :

- un Service Web est sans état (*stateless*) c'est-à-dire qu'il ne peut pas sauvegarder le résultat d'une invocation à une autre. En effet, nous ne pouvons pas invoquer le même Service Web deux fois en passant comme paramètre pour la deuxième invocation le résultat de la première.
- Une autre limitation est la non persistance, c'est-à-dire si deux clients invoquent le même Service Web et que le premier client décide d'arrêter ce service (arrêter le container du services), le deuxième client sera pénalisé et ne pourra pas profiter de ce Service web.

Pour remédier à ce problème, Globus a développé le Web Service Resource Framework (WSRF). Il introduit la notion de Ressource (à ne pas confondre avec les ressources physiques ou logiques de la grille que nous avons vues jusqu'ici) qui stocke

un contexte d'exécution. Celui-ci, se présente sous la forme d'un ensemble de variables, possède un identifiant unique et est préservé entre les appels.

II.3.2 Les nouveaux composants

Avec sa version 4, on note l'apparition de certains nouveaux composants :

- Myproxy : qui facilite la gestion de clés et certificats appelés "grid credentials" de façon à ce que l'utilisateur n'ait pas à transférer ses clés de cryptographie à chaque fois qu'il change de client.

Avec la version 4 de Globus de nouveaux outils ont été ajoutés aux composants. On peut citer :

- globus-rls-admin pour l'administration des répliquas.
- globus-rls-cli permettant aux clients d'avoir accès aux répliquas.

II.4 Architecture de Globus version 4

Globus est composé d'un ensemble de modules ayant chacun une interface que les services de niveau supérieur pourront utiliser pour invoquer ses mécanismes.

Nous trouvons ainsi des modules pour :

- la localisation et allocation des ressources : permet aux applications d'exprimer leurs besoins en ressources et fournit les mécanismes permettant d'identifier les ressources adéquates,
- la communication : permet aux différentes applications de communiquer entre elles. Différents paradigmes de communication sont fournis (communication par messages, mémoire distribuée, appel de procédures à distance),
- l'information sur les ressources : permet d'obtenir des informations sur l'état et la structure globale du système,
- les mécanismes de sécurité : fournissent les mécanismes d'authentification et d'autorisation des utilisateurs,
- la création et lancement des processus : permet de préparer l'environnement dans lequel un processus s'exécutera et de le lancer,

- l'accès aux données : accès performant et consistant aux données stockées dans des fichiers et des bases de données.

Ainsi on trouve une architecture comme indiquée ci après :

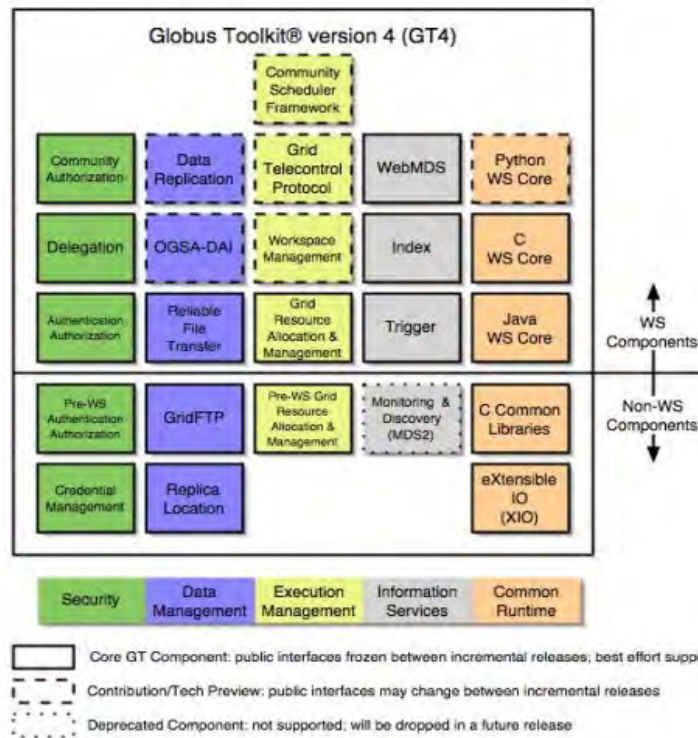


Figure 2. Vue d'ensemble des composants de Globus

Voyons maintenant les quatre grandes classes de services offerts par le toolkit Globus : le service de sécurité, le service d'information, le service de gestion des données et service de gestion des tâches.

II.4.1 Le service de sécurité

II.4.1.1 Concepts

Globus a développé un ensemble de protocoles nommés *Grid Security Infrastructure* (GSI) basés sur une architecture à chiffrement asymétrique (paire clé publique / clé privée) [17]. Ils sont utilisés pour l'authentification, la confidentialité des communications ainsi que pour l'identification. GSI répond à des contraintes telles que l'identification unique (single sign on), la délégation de permissions, l'intégration avec

divers outils de sécurité locaux (tels que Kerberos). Les utilisateurs et machines sont identifiés à l'aide d'un certificat au format X.509.

Ce dernier contient :

- a. des informations sur le propriétaire du certificat : nom, institut, etc.
- b. la clé publique associée au certificat,
- c. l'autorité de confiance qui assure que le propriétaire du certificat est bien celui qu'il dit être,
- d. la signature de l'autorité de confiance en question.

II.4.1.2 Procédure de fonctionnement

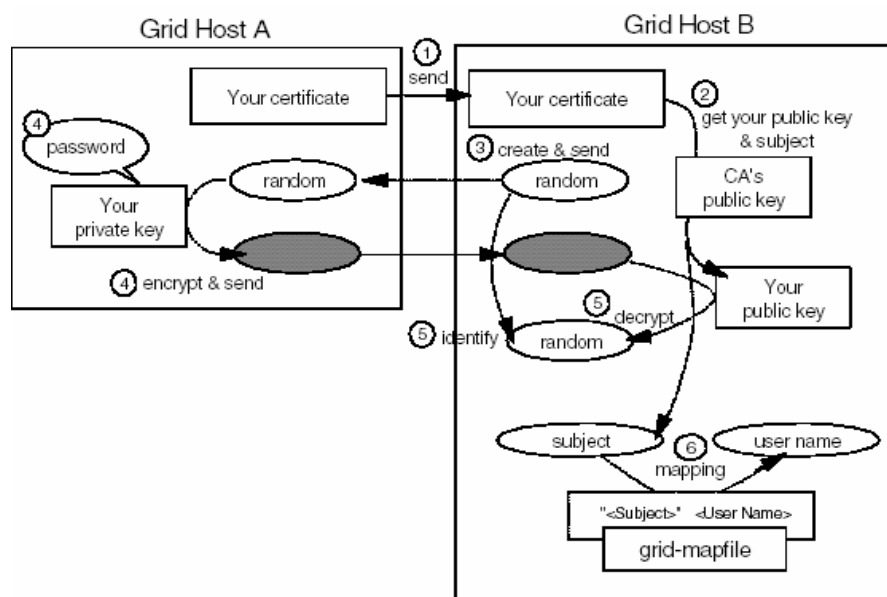


Figure 3. Authentification et autorisation.

La procédure est la suivante [12] :

1. Le noeud A envoie son certificat au noeud B.
2. B s'assure que le certificat est valide et extrait l'identité et la clé publique de A du certificat.
3. B génère un nombre aléatoire et l'envoie à A.
4. Lors de la réception de ce nombre, A le chiffre avec sa clé privée (c'est ici que A pourra demander à l'utilisateur d'entrer le mot de passe) et l'envoie à B.
5. Lors de la réception du nombre chiffré, B le déchiffre avec la clé publique de A et s'assure qu'il est le même. A est alors authentifié par B.

6. La procédure est répétée dans le sens inverse pour que A authentifie B.
7. B transpose le nom de l'utilisateur se trouvant dans le certificat en un nom d'utilisateur local au noeud. Pour cela un fichier appelé grid-map est utilisé. Il contient une entrée du genre "/O=Grid/O=Globus/OU=enst.fr/CN=Un Tel" untel. Cette ligne permet de transposer le DN (« Distinguished Name ») de Monsieur Un Tel en un nom d'utilisateur local (untel). Cette dernière étape constitue la phase d'autorisation.

II.4.2 La gestion des données

On y retrouve des services pour la localisation, le transfert et la gestion de données distribuées. Pour le transfert de données, Globus a développé une extension du bien connu *File Transfert Protocol* (FTP) nommée Gridftp. Ce protocole de transfert pour grilles est conçu sous la forme d'extensions au protocole FTP. Les buts qu'il souhaite fournir sont décrits [4].

La gestion des données distribuées est assurée, entre autres, par le *Repliqua Location Service* (RLS) qui enregistre dans un catalogue les différentes localisations d'un fichier afin d'en retrouver les répliquas. Le RLS peut être distribué sur plusieurs serveurs distants afin d'augmenter sa capacité et, surtout, d'éviter d'avoir un seul serveur centralisé qui mettrait à mal la gestion des données de la grille en cas de dysfonctionnement [4].

II.4.3 La gestion des ressources

II.4.3.1 Concepts

Lorsque l'on désire effectuer une tâche sur un ordinateur, il est nécessaire d'acquérir un accès à ce dernier, de le configurer pour qu'il puisse satisfaire nos besoins, d'y installer un exécutable, de démarrer et surveiller son exécution et, enfin, de récupérer les résultats. Le service *Grid Ressource Allocation and Management* de Globus (GRAM) fournit une interface au travers de services Web pour effectuer ces étapes sur des ordinateurs distants. Cette interface permet au client de spécifier des contraintes telles que les ressources désirées, le programme à exécuter ainsi que ses arguments. . .

GRAM peut s'interfacer avec plusieurs systèmes de gestion de processus tels que LSF, PBS, Condor.

Ainsi, il faut bien comprendre que GRAM n'est pas un Ordonnanceur de job mais bien une couche d'abstraction permettant de communiquer avec une série d'Ordonnanceur.

II.4.3.2 Fonctionnement

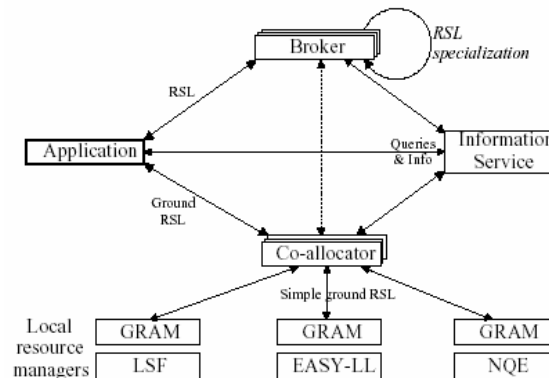


Figure 4. Architecture de gestion de ressources de Globus

Dans cette architecture, une application exprime ses besoins en ressources sous la forme d'une expression RSL de haut niveau. Les courtiers de ressources analysent récursivement la requête et la transforment en un ensemble de RSL bas niveau plus spécifique jusqu'à l'identification de l'ensemble des ressources susceptibles de satisfaire la demande de l'application. A ce stade de l'analyse nous avons une RSL de base (« **ground RSL** ») identifiant un ensemble de GRAM. Cette RSL de base est alors décomposée en RSL individuelles et chacune est envoyée au GRAM adéquat qui transforme la requête en une demande que le système sous-jacent comprend.

II.4.4 Le service d'information

La dernière grande catégorie de fonctionnalités que fournit Globus concerne la surveillance et la découverte des services et ressources. Leur but est de pouvoir récupérer, distribuer et collecter des informations relatives aux différents services et ressources disponibles sur la grille. Ces informations peuvent inclure :

II.5 Les limites de Globus

Les limites du Globus sont liées généralement à la sécurité et à son utilisation.

II.5.1 La sécurité

Globus n'assure pas tous les services de sécurité notamment l'intégrité des données échangées sur la grille. Par conséquent les données peuvent être modifiées sans qu'il y'ait aucune possibilité de faire des vérifications.

II.5.2 L'utilisation de Globus

L'utilisation de Globus n'est pas aisée dans la mesure où tout se fait en ligne de commande. Cependant une interface graphique appelée Cog jglobus est en cours de développement.

II.6 Mise en place de la grille de calcul

Cette couvre l'installation de base de l'interlogiciel Globus dans sa version 4.0.1 dont le but est d'obtenir un groupe de stations qui peuvent servir de nœuds pour une grille de calcul en utilisant un réseau LAN (Local Area Network) et de mettre en relief l'utilisation pratique des services de cet interlogiciel. Un de ces nœuds fera office d'autorité de certification pour la gestion des certificats des nœuds. Pour chaque étape, des tests sont exécutés. Ils sont nécessaires à la validation de l'installation.

II.6.1 Préparation à l'installation de Globus 4.0.1

L'installation de cette plate-forme nécessite l'installation du système d'exploitation (Linux), de Globus 4.0.1, et d'autres logiciels utiles au déploiement et à la réalisation, principalement "Java JDK", "Apache Ant" et "PostgreSQL". Ce logiciel n'est pas une version commerciale, et il reste à l'usage des universitaires et des centres de recherche. Le matériel utilisé est 2 PCs Pentium IV 2 GHz, avec 512 Mo de mémoire et des cartes réseaux Fast Ethernet de 100 Mbps et une machine Pentium III 800 MHz 256 Mo de RAM avec carte réseau Fast Ethernet 100 Mbps .Ces PCs sont interconnectés

via un switch 5 port 100 Mbps. Les média utilisés sont des câbles UTP catégorie 5 à paires torsadés non blindés.

Les versions utilisées dans l'installation :

- Fedora Core 3.
- Apache Ant 1.7.0.
- GT4.0.1.
- JDK-1-5-12.
- PostgreSQL-8.2.5.

II.6.1.1 Configuration du réseau

Lors de l'installation du système Linux, nous avons dû attribuer un nom et une adresse IP à chaque nœud de la grille pour servir par la suite à la configuration de Globus 4.0.1.

Il est important de noter que les noms des différents hôtes doivent avoir toute la forme suivante :

"nom_machine.nom_domaine" qui est une exigence de Globus.

Dans notre cas, notre grille est configurée de la manière suivante :

Nom de la machine	Adresse IP	Description
Poste1.ucad.sn	192.168.0.1	Client /Serveur
Poste2.ucad.sn	192.168.0.2	Client /Serveur
Poste3.ucad.sn	192.168.0.3	Client /Serveur

Tableau 2. Tableau des adresses des nœuds.

II.6.1.2 Installation du système Linux

Il faut suivre la procédure d'installation normale de 'Fedora Core3' en faisant attention à quelques points:

- Type d'installation Poste de Travail (WorkStation).
- Sécurité : pas de pare-feu (Pour ne pas gêner le fonctionnement de Globus).

- Mot de passe 'globus'.
- Personnalisation du jeu de paquetages s'il y a des pilotes particuliers pour la machine. Il faut ajouter les paquetages du noyau.
- Date et Heure doivent être réglées. Ce point est très important, car si les dates systèmes ne sont pas synchronisées, des problèmes auront lieu au niveau de la validité des certificats.

II.6.1.3 Création des comptes utilisateurs

Nous avons besoin de certains utilisateurs : un utilisateur "user", comme utilisateur simple, un utilisateur "root" comme administrateur "globus" qui sera obligatoire sur chaque noeud de la grille et servira à l'installation de l'interlogiciel Globus 4.0.1.

II.6.1.4 Création des répertoires d'installation

Pour l'installation de Globus et ses outils nécessaires, nous devons créer deux répertoires sur chaque nœud de la grille sous le répertoire '/usr/local' ; un pour les outils et un pour Globus, il faut ensuite ajouter les variables d'environnements correspondantes au fichier '/etc/profile' tel que la variable d'environnement '\$GLOBUS_LOCATION' afin de faciliter le traitement ci après. L'utilisateur est libre de choisir le répertoire d'installation à condition que ce répertoire appartienne à l'utilisateur 'globus'.

II.6.1.5 Installation des outils nécessaires pour Globus 4.0.1

Après l'installation de chaque outil, nous devons rajouter les modifications nécessaires aux fichiers '/etc/profile'.

- Installation de Apache Ant : c'est un exécuteur de tâches permettant de compiler et déployer un programme Java. Il est nécessaire lors du déploiement des services de grille.
- Installation de Java JDK : c'est un kit de développement nécessaire lors de la compilation du code de Globus 4.0.1 dont une grande partie est écrite en Java.

- Installation de PostgreSQL : c'est un SGBDR (système de gestion de base de données relationnelles) qui fonctionne sur des systèmes de type UNIX. C'est un logiciel libre qui possède de nombreuses caractéristiques faisant de lui un SGBDR robuste et puissant digne des SGBD.

II.6.2 Installation de l'interlogiciel Globus 4.0.1

Le code source de Globus est de 82Mo, son installation nécessite 312 Mo, et son déploiement 20 Mo. Le déploiement consiste à garder, suivant la configuration de la machine (système d'exploitation et type de processeur), les fichiers qui seront réellement utiles pour faire du calcul distribué.

II.6.2.1 Installation de l'Autorité de Certification : SimpleCA

SimpleCA est un paquetage qui fournit une autorité simplifiée de certification afin de publier des qualifications aux utilisateurs et aux services Globus 4.0.1, il englobe les fonctionnalités de OpenSSL CA. Pour l'installer, il faut lancer le script 'setup-simple-ca' qui demande d'entrer le mot de passe général du certificat. C'est ce dernier qui sera utilisé pour signer tous les certificats dépendant de cette Autorité de certification et il est le plus important.

Ce script se charge de créer la structure des répertoires qui contiennent tous les certificats de l'Autorité de Certification et ses clés publiques et privées (/home/globus/.globus/simpleCA).

Ensuite il faut installer le service GSI qui représente l'infrastructure du service de sécurité pour Globus Toolkit.

II.6.2.2 Configuration des services de grille de calcul

II.6.2.2.A Configuration du service Gridftp

Gridftp est un protocole à rendement élevé, sécurisé, fiable, permettant le transfert transparent de données. Il est optimisé pour les réseaux étendus. Le protocole Gridftp est construit au dessus du protocole FTP standard auquel sont ajoutées d'autres fonctions. Il utilise le service GSI pour l'identification des utilisateurs.

Ce protocole est déjà installé et il ne nous reste plus qu'à le configurer. Il faut donc éditer le fichier '/etc/services' et rajouter le nom de service, le port et le protocole [7].

II.6.2.2.B Lancement des web containers

Pour administrer les web containers, Java WS englobe deux actions : une pour le lancement (globus-start-container) et une pour l'arrêt (globus-stop-container).

II.6.2.2.C Configuration du service RFT (Reliable File Transfer)

Ce service fournit des interfaces pour contrôler les transferts entre serveurs Gridftp [7]. Il s'agit d'une réadaptation de l'utilitaire 'globus-url-copy'. Il nécessite le lancement du serveur 'postmaster' du SGBD 'Postgresql' en plus du service Gridftp. Sa configuration consiste à utiliser le processus d'authentification par lequel le serveur de bases de données établit l'identité du client et détermine si l'application cliente (ou l'utilisateur sous le nom de laquelle elle tourne) est autorisée à se connecter sous le nom d'utilisateur demandé. L'authentification du client est contrôlée par le fichier 'pg_hba.conf' situé dans le répertoire data, sous le chemin '/var/lib/pgsql/data/pg_hba.conf'. Un fichier 'pg_hba.conf' par défaut est installé lorsque le répertoire data est initialisé par la commande 'initdb'.

II.6.2.2.D Configuration du service GRAM

Comme il permet de lancer des programmes sur des ressources distantes, sans tenir compte de l'hétérogénéité locale (système d'exploitation, ordonnanceurs), la configuration du service GRAM consiste en l'édition du fichier 'gram_fs_map_config.xml' et l'ajout de deux lignes d'adaptateur local dans le fichier '/etc/sudoers' qui sont utiles lors de l'exécution des jobs sur différents hôtes et pour que le programme 'globusgridmap-and-execute' s'exécute à travers les utilisateurs qui se trouvent dans le fichier 'grid-mapfile'.

II.6.2.2.E Soumission des Jobs

Un job est une tâche à exécuter dans la grille et écrite selon un langage de programmation (RSL dans le cas de Globus) dans le but d'avoir des résultats. Il est considéré comme une unité simple de travail qui est typiquement soumis pour l'exécution sur la grille. Il nécessite des données, produit des sorties, et des conditions d'exécution afin d'accomplir sa tâche. Un job simple peut lancer un ou plusieurs processus sur un nœud indiqué. Il peut exécuter des calculs complexes sur des grandes quantités de données comme il peut être relativement simple. Les utilisateurs désirant utiliser la grille, doivent, après avoir récupéré un certificat, initier un proxy. Lors d'une requête de type " job ", plusieurs éléments de Globus entrent en jeu, comme GRAM et GSI (voir annexe).

II.7 Mise en place du site de Dakar dans le cadre du projet HP-UNESCO

Le site de Dakar a été mis en place pour participer à la grille mondiale du projet EGEE. C'est un projet mettant en partenariat HP, UNESCO et l'Université Cheikh Anta Diop de Dakar. La grille a été mise en place au niveau du Centre de Calcul Informatique de l'UCAD. Aujourd'hui, le site de Dakar fait partie de la grille mondiale EGEE et il constitue une organisation virtuelle.

L'interlogiciel Glite ressemble beaucoup à Globus dans son fonctionnement puisqu'il reprend tous les services de celui-ci. Leur différence entre eux se trouve au niveau du type d'architecture qu'ils peuvent fournir. En effet avec Globus nous avons une architecture Peer-to-Peer. Cela signifie que tous les services qu'il offre (sécurité, information, soumission etc.) sont implémentés au niveau de chaque nœud. Dans le cas de Glite nous avons une architecture orientée service. Chaque nœud de la grille implémente un service spécifique. Ainsi nous avons :

- un serveur de stockage : chargé de stocker toutes les données du site,
- un serveur de soumission : chargé de soumettre les jobs à l'unité d'exécution,
- un serveur d'exécution : chargé d'exécuter les jobs qui lui sont remis par le serveur de soumission,
- une interface utilisateur : qui permet de surveiller les jobs soumis.

Il faut remarquer que l'installation du système d'exploitation et de l'interlogiciel Glite a été prise en charge à distance par un serveur appelé quattor. Donc, le serveur quattor installe tous les composants nécessaires et suffisants pour mettre en place le système d'exploitation et Glite.

II.7.1 Préparation à l'installation de l'interlogiciel

L'installation nécessite la mise en place du système d'exploitation linux scientifique (version 4.6 dans notre cas) .La version de Glite utilisée est la version 3.1.

II.7.2 Configuration du réseau

Pour la configuration réseau nous avons utilisé une plage d'adresse attribuée par la Direction Informatique (DI). Ces machines sont interconnectées via un Switch 24 ports 1Gigabits.

Nom machine	Description
quattosrv.cci.ucad.sn	Serveur quattor
gridse.cci.ucad.sn	Serveur de stockage
gridce.cci.ucad.sn	Serveur de soumission
gridwn.cci.ucad.sn	Serveur d'exécution
gridwn2.cci.ucad.sn	Serveur d'exécution
gridui.cci.ucad.sn	Interface utilisateur

Tableau 3. Tableau des nœuds

II.7.3 Installation de Glite

II.7.3.1.A Configuration du serveur quattor

Il contient tous les fichiers de configuration. Les seuls templates à modifier sont dans `cfg/sites/cii/...` et `cfg/clusters/cii-3.1`. Ces templates permettent entre autres de:

- définir les profils des machines : `cfg/clusters/cii-3.1/profiles`,
- définir la liste des machines : `cfg/clusters/cii-3.1/site/pro_ste_database`,
- définir les paramètres réseau du site: `cfg/clusters/cii-3.1/pro_site_global`,

- ajouter de nouveaux utilisateurs.

Sur le serveur quattor est aussi installé DNS, DHCP(Ne répond qu'à des requêtes pour les machines gérées par Quattor).

II.7.3.1.B Configuration de la machine Computing Element

La configuration se trouve principalement dans 2 templates :

- sites/cci/site/glite/config : paramètres Glite généraux.
- sites/cci/site/glite/maui : paramètres du batch scheduler (MAUI).

Les Principales commandes Torque/MAUI :

- Liste des jobs : showq (MAUI).
- Listes des WNs (-n), queues (-c)... : diagnose (MAUI).
- Information détaillée sur un job (MAUI) : checkjob jobid.
- Information détaillée sur un job (Torque) : qstat -f jobid.

II.7.3.1.C Installation du Storage Element

Le fichier de configuration principale est dans : cfg/sites/cci/site/dpm/config.tpl

La principale commande de gestion : dpm-qryconf qui affiche la configuration, utilisation

II.7.3.1.D Installation du Worker Node

Il est entièrement configuré par quattor. Le fichier de configuration principale est dans: cfg/sites/cci/site/dpm/config.tpl. Les paramètres Glite : sites/cci/site/glite/config.

Conclusion

Ce chapitre a été l'occasion de se familiariser avec Globus dans sa version 4. Effet, nous avons fait une vue d'ensemble de ses différents composants et services offerts notamment la sécurité, la gestion des données, la gestion des ressources etc. Nous avons aussi fait un état sur les limites de l'interlogiciel.

Dans ce qui suit nous parlerons des systèmes de transfert dans les grilles pour essayer de dégager la problématique de notre recherche.

Chapitre III: SYSTEMES DE TRANSFERT DE DONNEES ET PROBLEMATIQUE DE RECHERCHE

III.1 INTRODUCTION

Pour gérer des données, il existe plusieurs grandes familles de techniques [20]. Nous allons définir quelques systèmes :

- les systèmes de gestion de base de données : ces systèmes permettent bien évidemment de gérer les données stockées dans des bases de données. SQL Server est un SGBD (système de gestion de fichier),
- les systèmes de transfert de données: ces systèmes sont plutôt utilisés dans l'optique de mettre en place un système de pré-chargement (*download*) puis de rapatriement (*upload*) des données. Les systèmes dépendent alors d'un système de stockage qui utilise alors généralement un système de fichiers local,
- les systèmes de fichiers : C'est le moyen le plus naturel pour accéder à des données puisque celles-ci sont attachées au système d'exploitation. NTFS (*New Technology File System*) est un système de gestion de fichier.

Dans la suite du mémoire, nous étudierons en détail les systèmes de transfert de fichier. Nous insisterons sur le protocole Gridftp qui est utilisé dans les grilles de calcul afin de mieux comprendre son mode de fonctionnement. De ce fait, nous pourrons dégager notre problématique de recherche.

III.2 Les systèmes de transfert de fichiers

L'une des considérations les plus importantes dans un transfert de fichiers est la représentation des données. Par représentation des données nous entendons la structure et le mode de transfert des données. Pour chaque représentation la fin du fichier est signalée par un *EOF* (End Of File) en fin de transfert.

III.2.1 La structure des fichiers

La structure définit la manière avec laquelle sont représentées les données. Trois structures de fichiers sont connues : la structure fichier, la structure enregistrement et la structure page.

III.2.1.1 La structure fichier

Dans cette structure le fichier est considéré comme une séquence continue d'octets contigus, sans structure sous-jacente induite. Il n'y a en fait aucune structure sous-jacente et le fichier doit être considéré comme une suite continue de caractères.

III.2.1.2 La structure page

Dans laquelle le fichier peut être considéré comme une suite de pages indépendantes indexées. Les fichiers de ce type sont aussi connus comme des fichiers à accès aléatoire. Chaque fin de page est marquée par un ***Last Page***.

III.2.1.3 La structure enregistrement

Ici un fichier peut être vu comme une séquence d'enregistrements. Le fichier est alors reconnu comme une suite ordonnée d'enregistrements successifs. Chaque enregistrement est suivi par un marqueur de fin d'enregistrement appelé ***EOR*** (End Of Record).

Le tableau ci-dessous résume les différentes architectures.

Nom de la structure	Description
Fichier	Données envoyées sous forme d'octet contigus + EOF
Page	Données envoyées sous forme de page + <i>Last Page</i>
Enregistrement	Données envoyées sous forme d'enregistrement + EOR

Tableau 4. Tableau récapitulatif des structures

III.2.2 LES MODES DE TRANSFERT DE FICHIERS

Le mode de transmission définit la façon dont les données sont transmises. Trois modes sont reconnus : le mode flux, le mode bloc et le mode compressé [13].

III.2.2.1 Le mode flux

Dans ce cas les données sont transmises comme un flux d'octets. Il n'y a aucune restriction sur la représentation des données utilisées. Des structures enregistrement sont autorisées. Dans un fichier d'enregistrements, les séquences *EOR* et *EOF* seront toutes deux marquées par un code de contrôle. Si la structure est de type "fichier", la séquence EOF sera implicitement marquée par la fermeture du canal. Tous les octets transmis sont donc des octets de données.

III.2.2.2 Le mode bloc

Le fichier est transmis comme une suite de blocs de données précédées d'un ou plusieurs octets d'en-tête. L'en-tête contient un champ de comptage de blocs, et un code de description. Le champ de comptage indique la longueur totale du bloc de données en octets, et indique donc le début du bloc suivant (il n'y a pas de bits de bourrage). Le code de description indique : le dernier bloc du fichier (*EOF*), le dernier bloc de l'enregistrement (*EOR*), le marqueur de reprise ou de données suspectes (c'est-à-dire qu'il est possible que les données transférées soient erronées et non fiables).

III.2.2.3 Le mode compressé

Dans le mode compressé les données sont envoyées sous la forme d'octets représentant le nombre d'occurrences des caractères et leur position. Trois classes d'informations doivent être envoyées : des données "littérales", envoyées comme des chaînes d'octets et représentant les occurrences; des données compressées, consistant en des octets "répliqués" et représentant la position des différentes occurrences; et des informations de contrôle, émis selon des séquences d'échappement à deux octets. Si

$n > 0$ octets (jusqu'à 127 représentant les occurrences) littéraux sont émis, ces n octets doivent être précédés d'un octet dont le bit de poids fort est nul, les 7 autres bits contenant ce nombre n .

Le tableau ci-dessous résume les protocoles de transfert et leur mode de représentation.

Nom du mode de transfert	Description
Flux	Données envoyées sous forme de flux d'octets
Bloc	Données envoyées sous forme de bloc de données
Compressé	Données envoyées sous forme littérale et répliquée

Tableau 5. Tableau récapitulatif des modes de transfert

III.2.3 Quelques protocoles de transfert de fichier

III.2.3.1 Le protocole http

Le protocole HTTP (Hyper Text Transfer Protocol) est un protocole de type client/serveur conçu pour transférer les données des sites web. Une connexion TCP est d'abord initiée par le client afin de pouvoir transmettre une requête, puis la réponse du serveur revient par le chemin inverse. La version 1.1 permet, entre autres, de réutiliser la connexion ouverte afin d'envoyer plusieurs requêtes de manière séquentielle sans avoir à payer à nouveau le coût d'établissement d'une nouvelle connexion. Un autre aspect intéressant est le support de l'extension *byte-range* qui permet de récupérer une partie d'un fichier (pour terminer un téléchargement ou bien pour récupérer des morceaux d'un fichier répliqué sur plusieurs sites) [20]. Il supporte la structure page et le mode de transfert flux.

III.2.3.2 Le protocole FTP

Le protocole FTP (*File Transfer Protocol*) [20] est un protocole développé, comme son nom l'indique, pour effectuer des transferts de fichiers, et ce, entre un client et un

serveur. FTP s'appuie sur le protocole TCP et lors d'un transfert de fichier, deux connexions sont créées. La première est une connexion de contrôle, créée à l'initiative du client vers le port du service FTP (usuellement port 21). La seconde (port 20 par défaut) peut être créée, soit à l'initiative du serveur (mode actif). Ce qui peut poser des problèmes avec certains systèmes pare-feu. Soit à l'initiative du client (mode passif). Entre plusieurs transferts consécutifs de fichiers, la connexion de contrôle est conservée. FTP permet aussi de gérer des transferts à l'initiative d'un client tiers, c'est-à-dire entre deux serveurs en utilisant des requêtes permettant de préciser la destination (requêtes **PORT** et **PASSIVE**). Avec les adresses de deux machines différentes, les transferts et les ports qui vont bien, le transfert peut donc être effectué. Cette opération peut devoir être autorisée explicitement sur les serveurs qui doivent accepter une connexion venant d'une machine différente de celle qui a demandé contrôle.

Dans sa version de base, le protocole est considéré comme peu sûr car d'une part le canal de contrôle est dérivé du protocole telnet et d'autre part, le « vol de connexion » est relativement aisé si le serveur n'effectue pas de vérifications suffisamment poussées sur les sources et les destinations du transfert. De plus, cette mauvaise réputation est entachée par le fait que de nombreux trous de sécurités critiques permettant d'obtenir aisément les privilèges d'administrateur aient été découverts dans de nombreuses implantations de ce protocole [14].

Le protocole FTP supporte les modes de transfert flux, bloc et compressé. Il supporte les structures fichier, enregistrement et page. Dans le mode bloc l'en-tête de consiste en trois octets. Sur ces 24 bits d'information d'en-tête, les 16 bits de poids faible représentent le compte d'octets, les 8 bits de poids fort donnent le code de description selon les définitions ci-dessous.

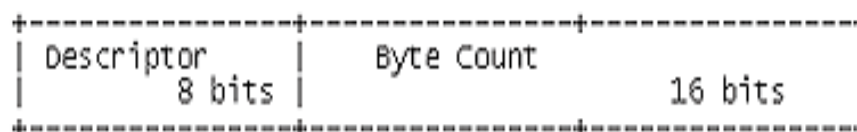


Figure 6. Entête de bloc FTP

Le descripteur est formé de bits indicateurs. Quatre codes sont actuellement reconnus, dont le nombre représente la valeur décimale du masque :

- 128 : le bloc est le dernier d'un enregistrement (**EOR** en fin de bloc),
- 64 : le bloc est le dernier de la transmission (**EOF** en fin de bloc),
- 32 : Erreurs probables dans les données de ce bloc,
- 16 : Le bloc de données est le premier d'une reprise.

III.2.3.3 Le protocole Gridftp

Ce protocole est une extension du protocole FTP conçu pour les grilles de calcul [4]. Il fournit un ensemble de fonctionnalités permettant d'utiliser les différents systèmes de stockage au cours d'utilisation. Le choix d'étendre le protocole FTP s'explique par le fait que c'est le protocole le plus utilisé pour le transfert de données dans l'Internet, et en plus des autres protocoles il reste le plus proche pour satisfaire les besoins de la grille.

Gridftp comprend les fonctionnalités suivantes :

- le service de sécurité GSI,
- le contrôle des données par *third-party* : un client peut initier un transfert de fichiers entre deux serveurs,
- le transfert de données en mode stripé : la distribution d'un fichier sur plusieurs hôtes,
- le transfert de données parallèle : envoi d'un fichier à travers plusieurs ports de données,
- la négociation de la taille des buffers TCP,
- la reprise après panne,
- l'instrumentation de code.

Les extensions consistent en des commandes et un nouveau mode de transfert. Parmi les commandes nous pouvons citer la commande SPAS, SPOR, ERET, ESTO, SBUF.

III.2.3.3.A La commande de Striped Passive (SPAS)

Cette commande est analogue à la commande **PASV** (mode passif de FTP). Cette commande demande au serveur de se mettre à l'écoute d'un port de données et d'attendre un ou plusieurs canaux de données plutôt que de prendre l'initiative d'en

établir une sur réception d'une commande de transfert. La réponse à cette commande précise les adresses et les ports sur lesquels le serveur s'est mis en écoute.

III.2.3.3.B La commande Striped Dates Port (SPOR)

Cette commande est analogue à la commande **POR** (indique le port de données dans FTP) doit être utilisée pour permettre le *third-party* transfert.

III.2.3.3.C La commande Extended Retrieve (ERET)

Analogue à la commande **RETR** (commande pour transmettre les données dans FTP) mais il autorise les modifications des données à manipuler avant d'être transmise.

III.2.3.3.D La commande Extended Retrieve STORE (ESTO)

Analogue à la commande **STORE** (commande pour stocker les données dans FTP) mais il autorise les modifications des données à manipuler avant d'être transmise.

III.2.3.3.E La commande Set Buffer Size (SBUF)

Cette extension ajoute la possibilité au client de définir la taille du buffer TCP c'est-à-dire celle des buffers d'émission et de réception TCP pour les connexions de données.

III.2.3.3.F Le mode Extended Block Mode

Les mode de transfert parallèle et strippé décrits ci dessus exigent un mode de transfert spécifique pour la délivrance des données de façon séquentielle et partielle à travers les ports de connexion. Le mode **E-Block** étend le mode bloc du protocole FTP pour permettre le transfert de larges volumes de données dépassant les limites de FTP.

La structure de l'entête du bloc **E-Block** est :

Descripteur	Champ de comptage	Offset
8 bits	64 bits	64 bits

Figure 7. Entête du bloc E-Block de Gridftp

Le descripteur est formé de bits indicateurs. Six codes sont actuellement reconnus:

- 128 : le bloc est le dernier d'un enregistrement (**EOR** bloc),
- 64 : le bloc est le dernier de la transmission (**EOF** bloc),
- 32 : erreur probable dans les données de ce bloc,
- 16 : le bloc de données est le premier d'une reprise,
- 8 : le bloc est **End of Data Count** (EODC pour cette connexion),
- 4 : l'émetteur va fermer le canal de données.

Certains mécanismes sont ajoutés au canal de données du mode bloc afin de gérer correctement la fin du fichier en présence d'un nombre inconnu de flux.

- Quand il n'y a plus de données à envoyer dans un canal de données, l'émetteur marque le dernier bloc ou envoie un bloc vide. Il va marquer le dernier bloc avec le code 8 pour signifier que c'est le dernier du canal des données.
- Après réception du message **EODC** le canal de donnée peut être utilisé pour d'autres transferts. Pour signifier que le canal de données est fermé après le dernier bloc un code 4 est envoyé dans le dernier message (bloc) envoyé.
- L'émetteur communique un **EOF** à tous les serveurs concernant le transfert.

En plus des champs décrits ci-dessous l'entête du bloc **EODC** comporte un champ appelé **EOD count excepted**. Ce champ de 64 bits représente le nombre de blocs envoyés dans un canal. Le numéro est utilisé par les récepteurs pour déterminer qu'il a reçu toutes les données. Lorsque le nombre de messages **EOD** (fin d'un bloc de données) reçus et le nombre **EOD count excepted** sont égaux, alors la fin du fichier est bien reçue.

Le tableau ci-dessous résume les protocoles de transfert et leur mode de représentation.

Système de transfert	Modes de transfert utilisés	Structures utilisées
HTTP	Flux	Fichier
FTP	Bloc, flux et compressé	Fichier, enregistrement, page

Gridftp	Bloc, flux	Fichier, enregistrement
---------	------------	-------------------------

Tableau 6. Tableau de synthèse des systèmes de transfert

III.3 Performances dans les systèmes de transfert

III.3.1 Les techniques de mesure de performance dans les systèmes de transfert

Il existe trois approches pour évaluer les performances d'un système de transfert. Ces approches sont : le modèle analytique, la simulation et la mesure [21].

- Le modèle analytique : met en œuvre des modèles calculables (files d'attente, réseaux de Pétri, ...) sous la forme de formules mathématiques.
- La simulation : met en œuvre des modèles conceptuels du système de transfert sous test (représentation conceptuel du système) qui nécessitent leur développement dans un outil de simulation.
- La mesure : permet de quantifier les critères de performance en les mesurant directement sur un système de transfert réel. Dans cette catégorie, nous distinguons deux variantes d'approches de mesure : l'instrumentation du code source de l'application en insérant des codes de mesure (*in-side*), ou l'utilisation de moniteurs externes à l'application pour quantifier les mesures au cours de cycles d'exécution de l'application (*out-side*).

III.3.2 Les métriques de performance dans les systèmes de transfert

Une métrique dans un système de transfert est une unité de mesures permettant d'évaluer de façon quantitative la qualité du transfert des données. Les métriques permettent de faire des mesures sur des paramètres de performances comme le temps. Dans un système de transfert les performances sont généralement liées la machine, au réseau et au code de l'application. Ainsi, les métriques qui leur sont reliées sont nombreuses.

III.3.2.1 Les métriques machines

Certaines caractéristiques de l'ordinateur sont des métriques de performance d'un système de transfert. Parmi ces métriques nous pouvons citer :

- la mémoire allouée : indique la mémoire totale allouée. C'est une métrique de performance très utile dans le cas de transfert mémoire à mémoire,
- l'accès disque : indique le temps d'accès au disque rigide. C'est une métrique très utile dans le cas des transferts disque à disque.

III.3.2.2 Les métriques réseaux

Les performances d'un système de transfert sont aussi à un ensemble de métriques reliées au réseau. Entre autres métriques réseaux nous pouvons citer :

- l'aller-retour : indique le nombre d'aller-retour requis pour qu'un bloc d'informations soit transmis d'un point à l'autre,
- le débit : fournit la vitesse de transmission des données lors des transferts,
- le temps de réponse : fournit le temps requis par une information pour être transmise et recevoir une réponse,
- la latence : indique le délai de transmission entre deux points du réseau,
- la taille du bloc : indique la taille du bloc de données transmis dans le cas d'un mode de transmission bloc,
- la taille des buffers : indique la taille des buffers d'émission et de réception utilisés lors du transfert,
- le nombre de canaux : indique le nombre de canaux utilisés lors du transfert.

III.3.2.3 Les métriques codes

Un système de transfert est une application écrite dans un langage de programmation choisi. Par conséquent, le design du code joue un rôle dans les performances. Parmi les métriques qui lui sont liées nous pouvons citer :

- Le nombre de processus : qui fournit le nombre limites de processus concurrents de l'application,
- Usagers concurrents : fournit le nombre d'accès usagers concurrents par seconde.

Le tableau suivant récapitule les métriques de transfert :

Métrique	type	Unité de mesure	Description
Temps de réponse	Réseau	Milliseconde	fournit le temps requis par une information pour être transmise et recevoir une réponse
Débit	Réseau	Mbps	Fournit la vitesse de transmission des données
Mémoire allouée	Machine	Ko, Mo	la mémoire totale (physique et virtuelle) allouée
Accès disque	Machine	Nombre/sec.	le nombre d'accès au disque rigide par seconde
Taille de paquet	Réseau	Pourcentage	la proportion par paquet de l'en-tête de protocole les données transmises
Latence	Réseau	Millisecondes	le délai de transmission entre deux points de réseau
Aller-retour	Réseau	Nombre	le nombre d'aller-retour requis pour qu'un bloc d'informations soit transmis d'un point à l'autre
Taille des buffers	Réseau	KO, Mo	Taille des buffers d'émission et de réception TCP
Nombre de canaux	Réseau	Nombre	Nombre de canaux de données utilisés lors du transfert
Taille moyenne de paquets	Réseau	Ko, Mo	la taille moyenne des blocs de données transmis
Usagers	Code	Nombre	Fournit le nombre d'accès usagers

concurrents			concurrents par seconde
Nombre de processus	Code	Nombre	Fournit le nombre de processus concurrents de l'application

Tableau 7. Tableau de métriques

III.4 Problématique de recherche

III.4.1 Constat

Dans l'environnement des grilles l'accès aux données est aussi important que l'accès aux ressources. Les applications scientifiques nécessitent le transfert de larges volumes de données (téraoctets ou pétaoctets). L'accès à de larges volumes de données (gigaoctets ou téraoctets) par plusieurs organisations géographiquement distribuées pour des besoins d'analyse et de visualisation constitue l'un des défis du protocole Gridftp. Ainsi, par ses extensions il vise à augmenter les performances lors des transferts des données. Cette augmentation de performance passe par une bonne gestion des ressources, de fichiers et de certains paramètres.

La gestion des ressources telles que la bande passante, les canaux et les ports utilisés peut jouer un rôle très important dans le transfert Gridftp. Cependant, dans certains cas l'utilisation de ces ressources peut conduire à de mauvaises performances. Ainsi, il est important de faire des études de performance afin de comprendre le fonctionnement du protocole et dégager les règles de gestion des ressources suivant la configuration de la grille.

Un autre facteur très important dans le transfert est aussi la gestion des fichiers devant être transférés. Suivant le mode de transfert des fichiers (bloc ou flux) mais aussi la façon de faire les transferts (un seul fichier ou plusieurs à la fois) on peut obtenir des bonnes performances. Cependant, il est important de connaître au préalable les possibilités offertes par le protocole en matière de gestion des fichiers pour réaliser les transferts dans les meilleurs délais possibles. D'où la nécessité d'étude permettant de dégager les règles de gestion des fichiers.

Le protocole offre la possibilité de paramétrer la taille de buffers TCP, la taille des blocs. Cependant, il ne définit pas de règles permettant de choisir les tailles optimales suivant la configuration de la grille. Donc, il serait intéressant de faire des études afin de dégager quelques règles permettant de bien choisir ces paramètres.

De manière générale il a été constaté que les extensions du protocole Gridftp peuvent être de véritables facteurs de performances lors des transferts. Cependant, elles doivent être bien choisies sinon on risque de ne pas obtenir les performances souhaitées. Donc, il est nécessaire de procéder à des études de performances afin de dégager des règles pour leur utilisation.

III.4.2 Apport de notre mémoire

Notre objectif est d'étudier les performances du protocole Gridftp utilisées dans les grilles de calcul. Dans un premier temps, nous nous proposons d'étudier et de réaliser des exécutions du service Gridftp entre un client et un serveur Globus, pour mettre en lumière les apports de ce service par rapport au service équivalent FTP, utilisé dans les environnements systèmes et réseaux IP standards. Cette étude consiste en une évaluation du temps de réponse qui correspond plus précisément au temps de transfert des fichiers.

Il est évident que toutes les métriques citées ci-dessus peuvent être objet de métrique dans cette de performance. Mais le choix porté sur le temps de transfert s'explique par le fait le dans les environnements de grille de calcul le temps traitement des données est le critère le mieux indiqué pour évaluer les performances. Ailleurs, notre objectif est de pouvoir dégager des règles d'utilisation du protocole. Donc, dans un second temps notre étude va se focaliser sur les impacts des extensions sur le temps de transfert. A savoir :

- l'impact du parallélisme sur le transfert,
- l'impact de la taille des buffers TCP sur le transfert,
- l'impact de la taille des blocs sur le transfert.

Gridftp étant d'abord un système de transfert de fichiers et ensuite une extension du

bien connu protocole FTP il est évident qu'il soit confronté à un certain nombre de problèmes liés à la gestion des fichiers, des ressources et de la sécurité. Donc, nous terminerons notre étude par une mise en évidence de quelques problèmes liés au protocole et une proposition d'amélioration de celui-ci.

Conclusion

Dans cette partie nous avons fait une étude des systèmes de transfert et dégager ainsi la problématique de notre recherche. Donc, dans la suite de notre étude on se focalisera sur une étude des performances du protocole Gridftp et de dégager dans la mesure du possible des pistes d'amélioration.

Chapitre IV: ETUDE DE PERFORMANCES ET PROPOSITION

IV.1 Les performances du Gridftp

IV.1.1 Les conditions des tests de performances

Les études de performances se sont déroulées dans les conditions suivantes :

- utilisation des machines poste1.ucad.sn et poste2.ucad.sn de la grille mise en place avec Globus,
- ces deux machines sont aussi configurées comme étant des serveurs FTP (version wu-ftp car c'est la version qui a permis d'implémenter Gridftp),
- utilisation de fichiers de taille variant de 100Mo à 450Mo (logiciel Split File *Pro v 2.2* pour découper les fichiers),
- le temps de transfert est obtenu grâce à l'option de performance `-vb`. Avec cette option nous avons toutes les 5 secondes la taille de fichier transféré, le débit instantané, le débit moyen et le temps de transfert.

IV.1.2 Gridftp vs ftp

Il s'agit de faire une étude comparative entre Gridftp et ftp.

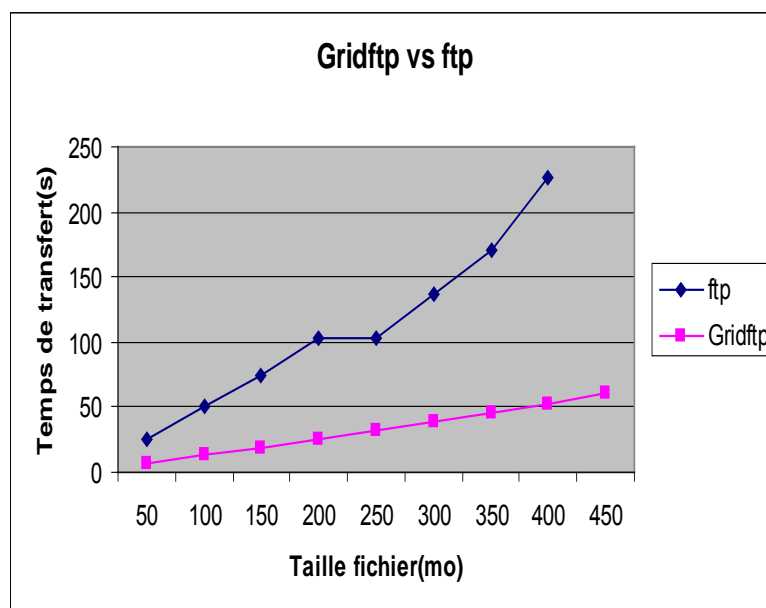


Figure 8. Etude comparative de Gridftp et de ftp

Interprétation

Les protocoles Gridftp et FTP utilisent le mode de transfert bloc dans le cas de ce test. Avec le protocole Gridftp les données sont envoyées par blocs de taille égale à 64Ko (dans le cas précis de ce test). De même, le protocole FTP utilise une taille de bloc égale à 64Ko. Ailleurs, les protocoles utilisent tous les deux le protocole TCP pour le transport des données. La taille du buffer d'émission et de réception est de 104Ko dans le cas du Gridftp et de 64Ko dans le cas de FTP.

Donc, le nombre de paquets transmis par seconde par TCP dans le cas du Gridftp est supérieur à celui transmis dans le cas de ftp. Par conséquent, le temps de transfert des données Gridftp est plus petit que celui du transfert des données ftp.

IV.1.3 Impact du parallélisme sur le transfert

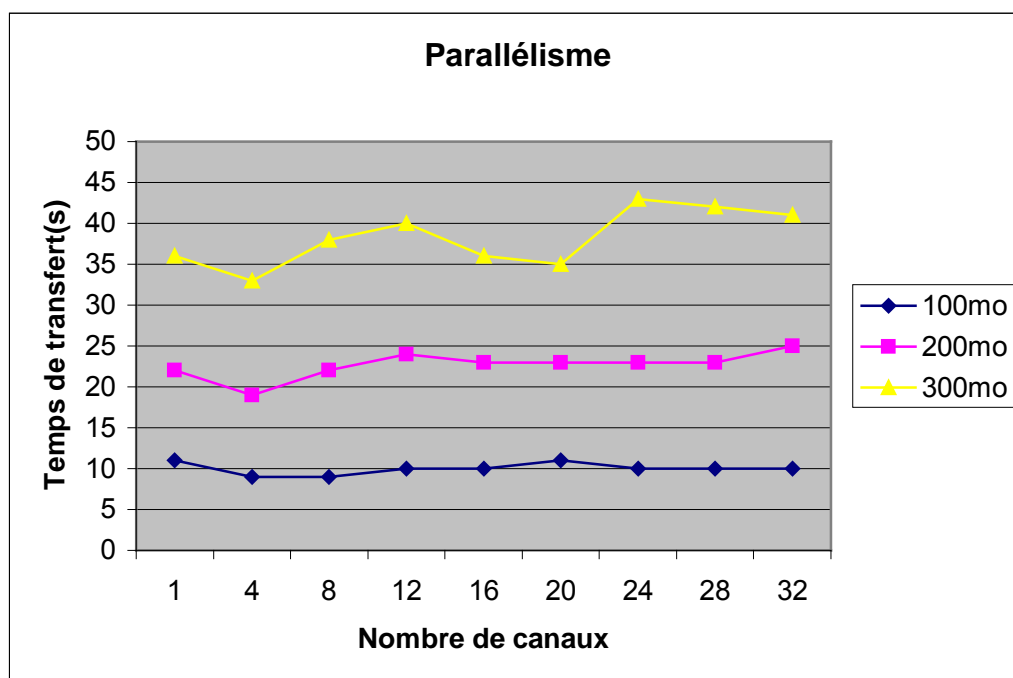


Figure 9. Le transfert parallèle

Interprétation



On voit bien que les meilleures performances sont obtenues pour un nombre de canaux compris entre 4 et 8. Ceci s'explique par le fait qu'entre 4 et 8 canaux il n'y a pas beaucoup d'échecs d'émission et que le débit dans les canaux ouverts est suffisant pour acheminer les données.

Par contre, au delà de 8 canaux on note une dégradation des performances et aussi des bouleversements. Il faut noter que la bande passante théorique utilisée pour le transfert est divisée par le nombre de canaux ouverts. Par conséquent, le débit utilisé devient plus faible favorisant. En le temps de retransmission des paquets TCP est fortement lié au **RTT** (Round Trip Time est le temps par un paquet pour faire un aller-retour) qui est petit (**RTT** inférieur à 1ms). Par conséquent, le **RTO** (Retransmission Time Out est le temps de retransmission des paquets TCP) est aussi petit d'où les retransmissions évoquées un peu plus haut.

En conclusion le nombre de canaux ouvert dépend fortement du **RTT**. Il est donc préférable d'utiliser un nombre de canaux petits dans le cas d'un faible RTT sinon on risque de faire un gaspillage de ressources (ports, bande passante).

IV.1.4 Impact de la taille des blocs

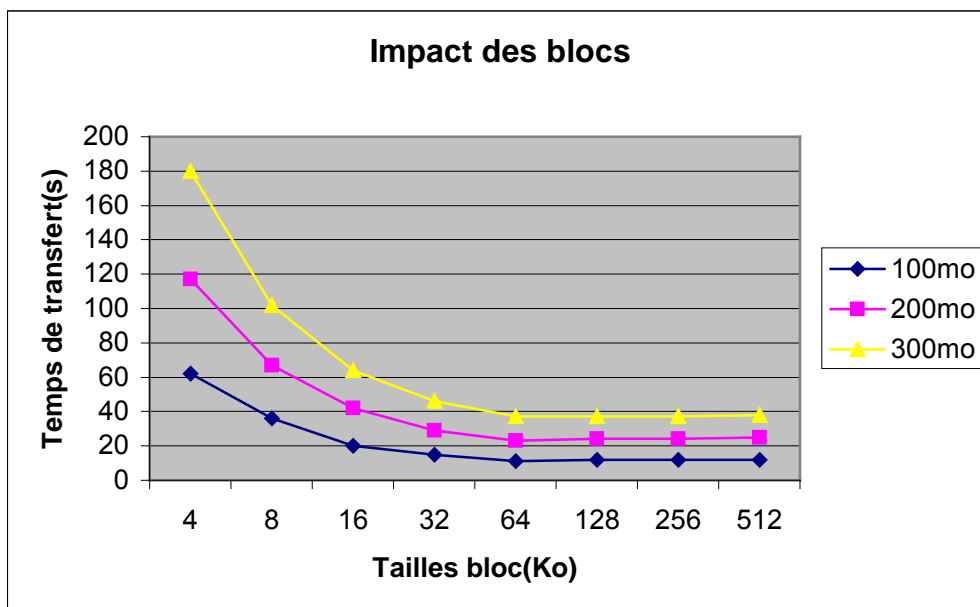


Figure 10. Paramétrage des blocs

Interprétation

Pour une taille de bloc inférieure à 64Ko le temps de transfert est décroissant. En effet, plus la taille des blocs augmente plus le temps mis par Gridftp pour mettre les données dans le buffer de réception TCP de réception est petit. Au delà de 64Ko le temps de transfert est constant. En effet 64Ko représente la taille des blocs par défaut adopté par Gridftp. Cela signifie que au delà de 64Ko les blocs de données sont découpés en blocs de 64Ko même si on paramètre les tailles des blocs.

Pourquoi Gridftp découpe les blocs de données supérieures a 64Ko ?

En réalité, au delà de 64Ko les blocs de données sont envoyés sous forme d'enregistrement de taille égale à 64Ko. Ceci est un bug du protocole Gridftp. Il est à l'image de Globus donc en phase de développement et par conséquent le paramétrage des blocs Gridftp n'est pas encore effectif dans son développement.

Pour un RTT très petit le paramétrage de la taille des blocs n'a pas beaucoup d'effet sur les performances du transfert des fichiers avec Gridftp.

IV.1.5 Impact de la taille des buffers TCP

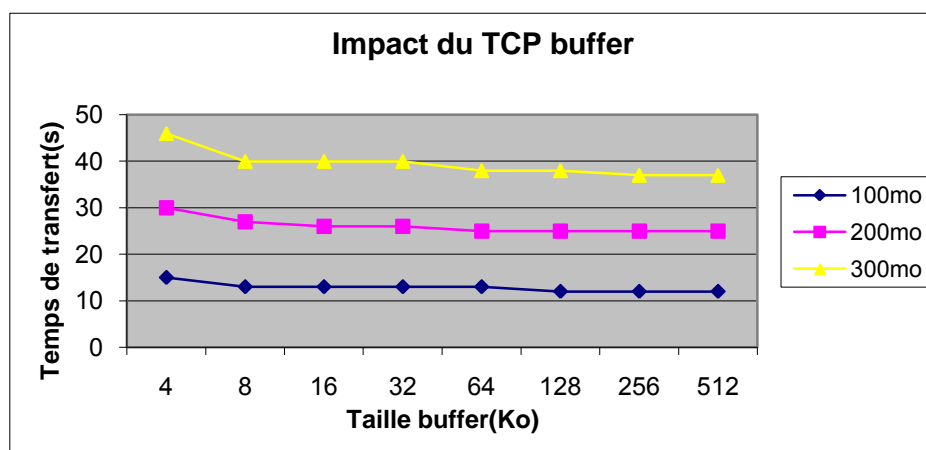


Figure 11. Paramétrage de la taille des buffers

Interprétation

Plus la taille des buffers est grande alors plus le nombre de paquets transmis par seconde par le protocole TCP est grand et plus le temps mis pour transférer les fichiers est petit. Au-delà d'une certaine taille (104 Ko), le temps de transfert reste constant. En effet, il a été trouvé que la meilleure taille de buffer est obtenue en appliquant la formule : Taille buffer= (Bande passante×RTT×1000) ÷8. Cependant, cette taille doit être supérieure ou égale à la taille du buffer par défaut (104Ko) sinon le système va considérer la taille par défaut. Dans le cas de notre test, la bande passante maximale =100 Mbps, RTT=0,317s d'où la taille de bloc optimale est 3963 octets. Donc, le système va considérer la taille par défaut d'où au delà de 104Ko le temps de transfert n'évolue pas.

Le paramétrage des buffers TCP n'a pas beaucoup d'influence sur les performances du Gridftp dans le cas où le RTT est faible.

IV.1.6 Impact de la combinaison sur les performances

IV.1.6.1 Parallélisme et blocs

Dans ce test nous avons fait varier la taille des blocs en fixant le nombre de canaux égal à 4.

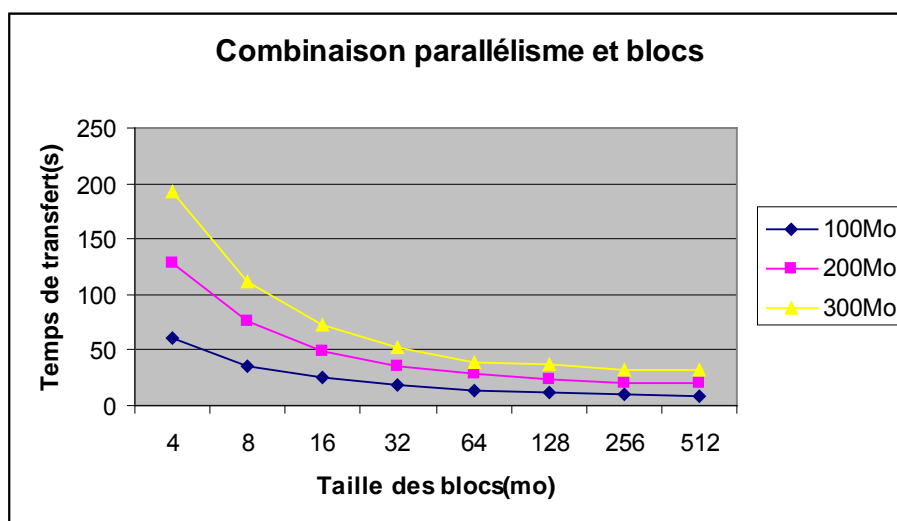


Figure 12. Paramétrage de la taille des blocs pour 4 canaux

Interprétation

On remarque que l'allure de la courbe est la même que celle du paragraphe 6.3.2. Cependant, on peut noter une meilleure performance par rapport au 6.3.2. En effet, pour des transferts des fichiers de 100 Mo, 200Mo et 300Mo on obtient respectivement 11s, 23s et 38s au paragraphe 6.3.2 contre 9s, 20s et 32s.

Donc, on peut dire qu'ici les performances notées sont liées en grande partie au parallélisme.

IV.1.6.2 Combinaison du parallélisme et TCP buffer

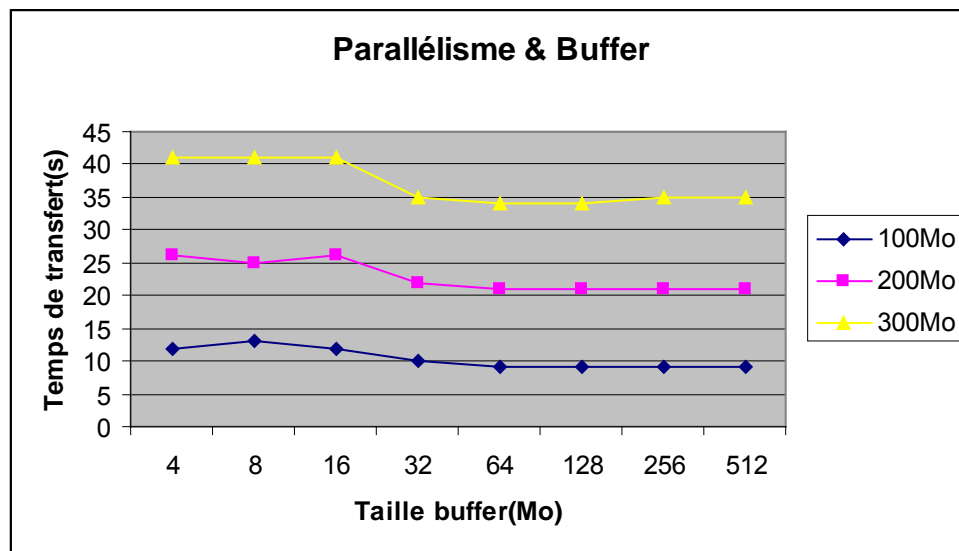


Figure 13. Paramétrage de la taille des buffers pour 4 canaux

Interprétation

On remarque pratiquement la même allure qu'au paragraphe 6.3.4. Cependant, il faut noter que les temps de transfert pour les tailles de buffer compris entre 4Ko et 32Ko ne diffèrent pas beaucoup. Ceci est du fait de l'utilisation des canaux permettant de faire les transferts plus vite qu'au paragraphe 6.3.4. En effet, au paragraphe 6.3.4 les meilleurs temps de performance pour les fichiers de 100Mo, 200Mo et 300Mo sont respectivement 12s, 25s et 37s contre 9s, 21s et 35s dans le cas du test ici. Ici encore, on peut dire les performances sont fortement liées au parallélisme.

A travers cette étude, on peut remarquer les extensions jouent des rôles très importants dans le temps transfert. Cependant, les performances diffèrent d'une extension à une autre. En effet, on constate que :

- le parallélisme est l'extension qui influence le plus le temps de transfert. Cependant, le nombre de canaux dépend fortement du RTT. Il est donc recommandé de choisir un nombre de canaux petit si le RTT est petit et d'en choisir un nombre assez important s'il (RTT) est assez grand.
- le paramétrage des buffers peut aussi influencer le temps de transfert s'il est bien paramétré. Ce paramétrage fait aussi appel au RTT. Si celui est faible, il est préférable de ne pas paramétrer le buffer. S'il est important, il est primordial de bien calculer la taille optimale avant de le paramétrer,
- la taille des blocs peut aussi être un facteur de performance s'il est aussi bien paramétré. Mais au stade actuel, la taille des blocs ne peut dépasser 64Ko du fait que le protocole est toujours en phase de développement.

IV.2 Proposition d'optimisation du temps de transfert des fichiers

Dans la partie précédente, nous avons fait une étude de performance du Gridftp et nous avons pu dégager les apports en performances de différentes extensions. Cependant, nous avons noté que ces performances peuvent être dégradantes si celles-ci ne sont pas bien choisies. Dans cette partie, nous nous proposons donc d'apporter des solutions pour améliorer les performances du protocole.

IV.2.1 Les problèmes du Gridftp

Parmi ces limites du protocole Gridftp, nous pouvons citer : le paramétrage des blocs, l'intégrité des données et le transfert séquentiel des fichiers.

IV.2.1.1 Le paramétrage des blocs

Le protocole Gridftp ne permet pas d'envoyer des blocs de taille supérieure à 64Ko. Donc, au-delà de 64Ko, tout bloc est envoyé sous forme de plusieurs blocs de 64Ko.

IV.2.1.2 L'intégrité des données

L'intégrité des données n'est pas assurée au niveau du canal de données. Ainsi, les données peuvent être victimes de modification sans aucune possibilité de faire des vérifications.

IV.2.1.3 Le transfert séquentiel

Avec le protocole Gridftp, il n'est pas possible de transférer plusieurs blocs de plusieurs fichiers sur un même canal de données de façon séquentielle.

Dans la suite de notre étude, nous nous focaliserons sur le troisième point cité c'est-à-dire le transfert séquentiel afin d'apporter des améliorations sur le Gridftp.

Le tableau suivant résume les problèmes du protocole Gridftp.

Problème	Description
Paramétrage bloc	Impossible d'avoir des blocs de taille supérieure a 64Ko
Intégrité de données	Pas de mécanisme de gestion de l'intégrité des données
Transfert séquentiel	Impossibilité des transferts de plusieurs fichiers

Tableau 8. Problèmes du Gridftp

IV.2.2 Identification du problème par test

Dans cette section, nous nous proposons d'identifier le fait que Gridftp ne puisse pas transférer plusieurs blocs de plusieurs fichiers de façon séquentielle sur un ou plusieurs canaux de données. Ainsi, les tests se sont déroulés dans certaines conditions.

IV.2.2.1 Les conditions des tests

Les tests ont été réalisés dans les conditions telles que le transfert se fait dans un seul et unique canal de données (puis 4 canaux). Les mesures suivantes ont été prises :

- dans le fichier de configuration du Gridftp nous avons fixé la rangée de port devant être utilisés à un seul canal (puis 4),

- nous avons ensuite lancé le client avec l'option `-dn 50000` pour signifier que celui-ci se connecte au serveur via le port de données 50000,
- lors du test nous choisissons toujours l'option `-p 1` pour dire que le transfert se fait en mode bloc à travers un unique canal de données.

Nous avons utilisé des fichiers de 50Mo, 100Mo, 150Mo, 200Mo, 250Mo, 300Mo, 350Mo, 400Mo. Dans un premier test, nous transférons le fichier de 50Mo et au cours du transfert nous essayons de transférer les autres fichiers puis nous notons les résultats obtenus. Dans un second test, nous transférons le fichier de 100Mo et au cours du transfert nous essayons de transférer les autres fichiers et puis noter les résultats.

IV.2.2.2 Résultats obtenus

Les résultats obtenus sont les suivants :

IV.2.2.2.A Premier résultat

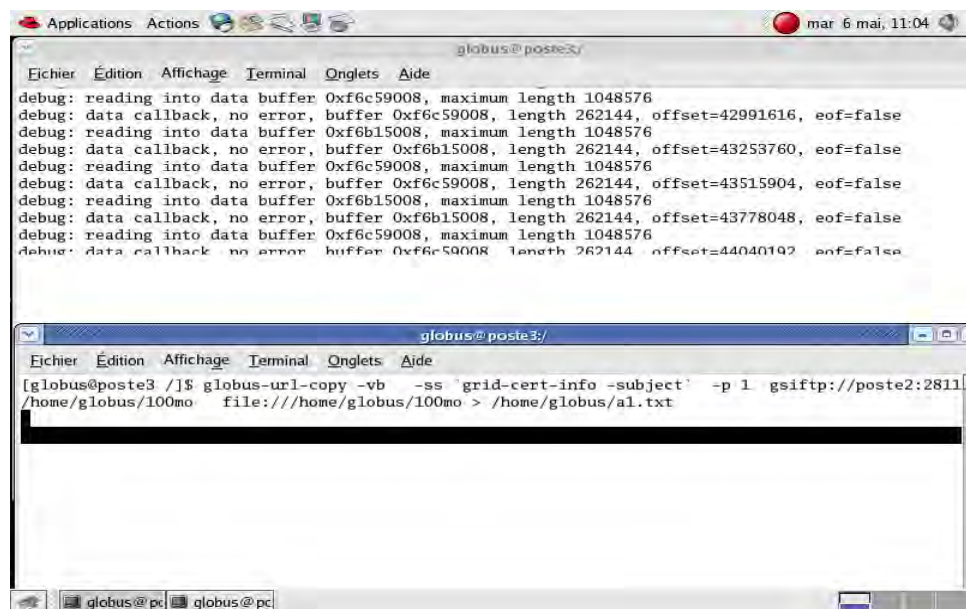


Figure 14. Fenêtre de transfert

IV.2.2.2.B Deuxième résultat

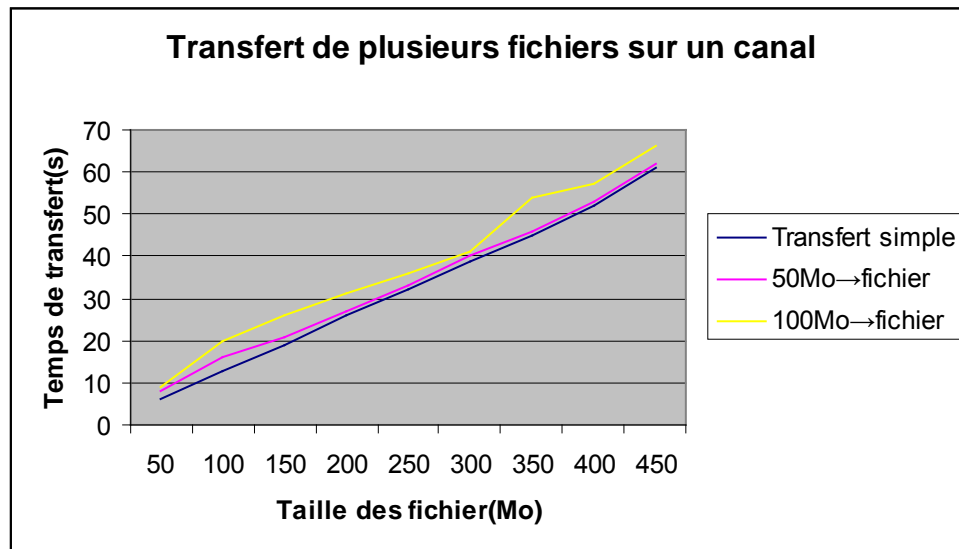


Figure 15. Transfert à travers un seul canal

IV.2.2.2.C Troisième résultat

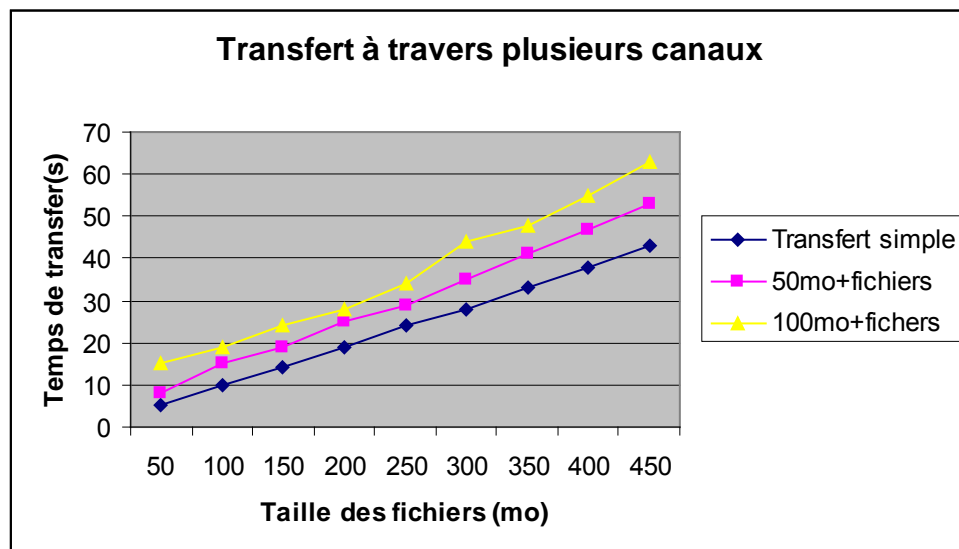


Figure 16. Transfert à travers 4 canaux

A travers ces tests, nous venons de montrer que le protocole Gridftp ne peut pas transférer plusieurs blocs de plusieurs fichiers sur un même canal de données de façon séquentielle. En effet, soient deux fichiers fichier1 et fichier2. Si le fichier fichier1 est entrain d'être transféré sur un canal de données bien défini alors Gridftp ne pourra envoyer le fichier fichier2 que lorsqu'il aura fini d'envoyer le fichier fichier1. Cela signifie que le fichier2 doit attendre un certain temps avant d'être transféré. Cette impossibilité de transfert séquentiel est plus importante lorsqu'on utilise plusieurs canaux de transfert où une partie du transfert peut finir sur des canaux bien avant sur d'autres canaux. Dans ce cas, il sera impossible d'utiliser les autres canaux tant que le transfert du fichier courant n'est pas terminé (envoi du EOF du fichier courant). Cette situation d'attente adoptée par Gridftp s'explique par le fait qu'il n'implémente pas des méthodes de différenciation des fichiers devant être transférés. A cela s'ajoute le fait que le protocole n'implémente pas de méthodes permettant de faire de l'allocation dynamique des ressources. Plus précisément, quand un fichier est entrain d'être transféré avec les ressources qui lui sont allouées et qu'un autre arrive une situation de concurrence se crée entre les deux fichiers pour l'appropriation des ressources.

Ce temps d'attente diminue les performances du protocole dans le cas où nous avons plusieurs fichiers à transférer. Il serait plus intéressant de pouvoir transférer des fichiers à chaque fois qu'on le désire. En outre, nous sommes en environnement de grilles et le traitement des données doit se faire de façon très rapide pour être utilisés (données météorologiques).

Dans la suite, nous nous focaliserons sur les moyens à adapter afin d'augmenter les performances du protocole. Plus précisément, nous travaillerons dans le sens d'éliminer le temps d'attente des fichiers dans le cas d'un transfert de plusieurs fichiers.

IV.2.3 Description de la proposition

IV.2.3.1 Objectif

L'objectif de notre étude est d'optimiser le temps de transfert des fichiers. Par conséquent, nous comptons mettre en place un algorithme qui puisse fournir un moyen d'éliminer le temps d'attente dans le cas de transfert de plusieurs fichiers différents. Donc, l'algorithme doit nous fournir les moyens suivants :

- un moyen d'éliminer la situation de concurrence créée lorsqu'on veut transférer plusieurs fichiers,
- un moyen de différencier les fichiers lors de leur transfert,
- un moyen d'utiliser tous les canaux alloués lors du transfert des fichiers,
- un moyen d'automatiser ce transfert sans aucune intervention manuelle.

IV.2.3.2 Notre démarche

La démarche que nous comptons adopter dans cette étude repose sur les principaux points suivants : exploitation de la structure enregistrement, identification des enregistrements et automatisation des transferts.

IV.2.3.2.A Exploitation de la structure enregistrement

Gridftp peut envoyer les données sous forme d'enregistrements suivis de EOR (End Of Record). Hors le délai d'attente du Gridftp est dû au fait que Gridftp envoie les fichiers les uns à la suite des autres en faisant une distinction entre eux. Donc, un moyen d'éviter ce délai est de faire en sorte que Gridftp considère les fichiers comme des enregistrements. Le choix porté sur la structure enregistrement s'explique par le fait que dans le protocole Gridftp cette structure est la mieux adaptée pour envoyer les données à travers plusieurs canaux. Donc, dans notre proposition nous allons considérer que tout fichier est un enregistrement suivi d'un EOR.

IV.2.3.2.B Identification des enregistrements

Le deuxième point à considérer est l'identification de ces enregistrements. De ce fait, on pourra reconnaître les fichiers lors de leur transfert mais aussi à leur réception. Ceci éviterait ainsi d'avoir des fichiers altérés ou illisibles à l'arrivée. L'idée serait donc de déclarer des identifiants d'enregistrement.

IV.2.3.2.C Automatisation des transferts

Une fois les identifiants des blocs définis il serait donc intéressant de voir comment les automatiser lors des transferts. En d'autres termes, il s'agit d'allouer les identifiants de blocs aux fichiers de façon automatique. Ceci évitera une gestion manuelle des identifiants de bloc.

IV.2.3.3 Les paramètres

Les paramètres à tenir en considération dans notre proposition sont les suivantes : une file, un buffer, un compteur et un délai d'attente, un serveur de stockage.

IV.2.3.3.A Une file

Nous considérons que tous les fichiers candidats sont mis dans une file avant d'être acheminés vers les canaux. La file permettra de stocker les données sous forme d'enregistrement. Gridftp peut acheminer des données allant des centaines de Mégabits à des centaines de péta bits suivant les caractéristiques des réseaux. Dans le cas de notre mémoire, nous supposons que la taille maximale des données est de 2,5 gigabits correspondant à la taille maximale de la file. Il faut noter que 2,5 gigabits est la taille maximale des données utilisées dans le cadre de nos tests de performance. Ainsi, pour rester dans le cadre de notre environnement de test nous travaillons toujours avec ces mêmes fichiers.

IV.2.3.3.B Un GASS(Globus Access to Secondary Storage)

Le GASS un serveur cache de Globus qui permet de stocker de façon temporaire les données devant être traitées et transférées. Nous supposons que dans notre proposition que le serveur GASS à une taille fixe égale à la taille de la file. Ce choix sur la taille du GASS s'explique par le fait que nous voulons toujours rester dans notre environnement de test.

IV.2.3.3.C Un buffer

Chaque fichier doit être stocké avec un EOF lors de sa réception par le système de stockage récepteur. Donc, le buffer permettra de stocker les EOF (blocs nuls) de chaque fichier (enregistrement) transféré. Il faut remarquer que même si le fichier est envoyé sous forme d'enregistrement avec un EOR à l'arrivée il doit être stocké avec un EOF. La taille maximale du buffer est égale à la taille d'un buffer d'émission multiplié par le nombre de buffer (nombre de canaux).Le buffer sera déclarée au niveau du GASS.

Taille buffer=nombre de canaux * taille d'un buffer d'émission.

Ainsi, lors de l'envoi des blocs nuls Gridftp pourra alors partager les blocs entre les différents buffers d'émission de façon très équitable (sans déficit ni excès).

IV.2.3.3.D Un compteur

L'idée du compteur est de contenir les valeurs des identifiants. Ce compteur sera incrémenté au fur et à mesure que les fichiers candidats arrivent et sa valeur sera affectée aux identifiants. Donc, à chaque fois qu'il y'a un fichier candidat Gridftp va lire la valeur du compteur et lui affecter l'identifiant de l'enregistrement avant de l'envoyer à TCP.

IV.2.3.3.E Un temporisateur

Nous considérons un temporisateur pour l'envoi des blocs nuls à partir du buffer. On suppose que ce temporisateur est le temps qu'aurait mis l'application pour remplir le

buffer. Le temporiseur pour l'envoi des blocs nuls est donc égal au temps mis pour mettre un bloc nul dans le buffer multiplié par le nombre de blocs pouvant être dans le buffer.

$$\text{Temporiseur} = \text{temps de bufferisation} * [E ((\text{taille buffer} / \text{taille bloc})) + 1].$$

E est la fonction partie entière.

IV.2.3.4 Schéma de l'algorithme

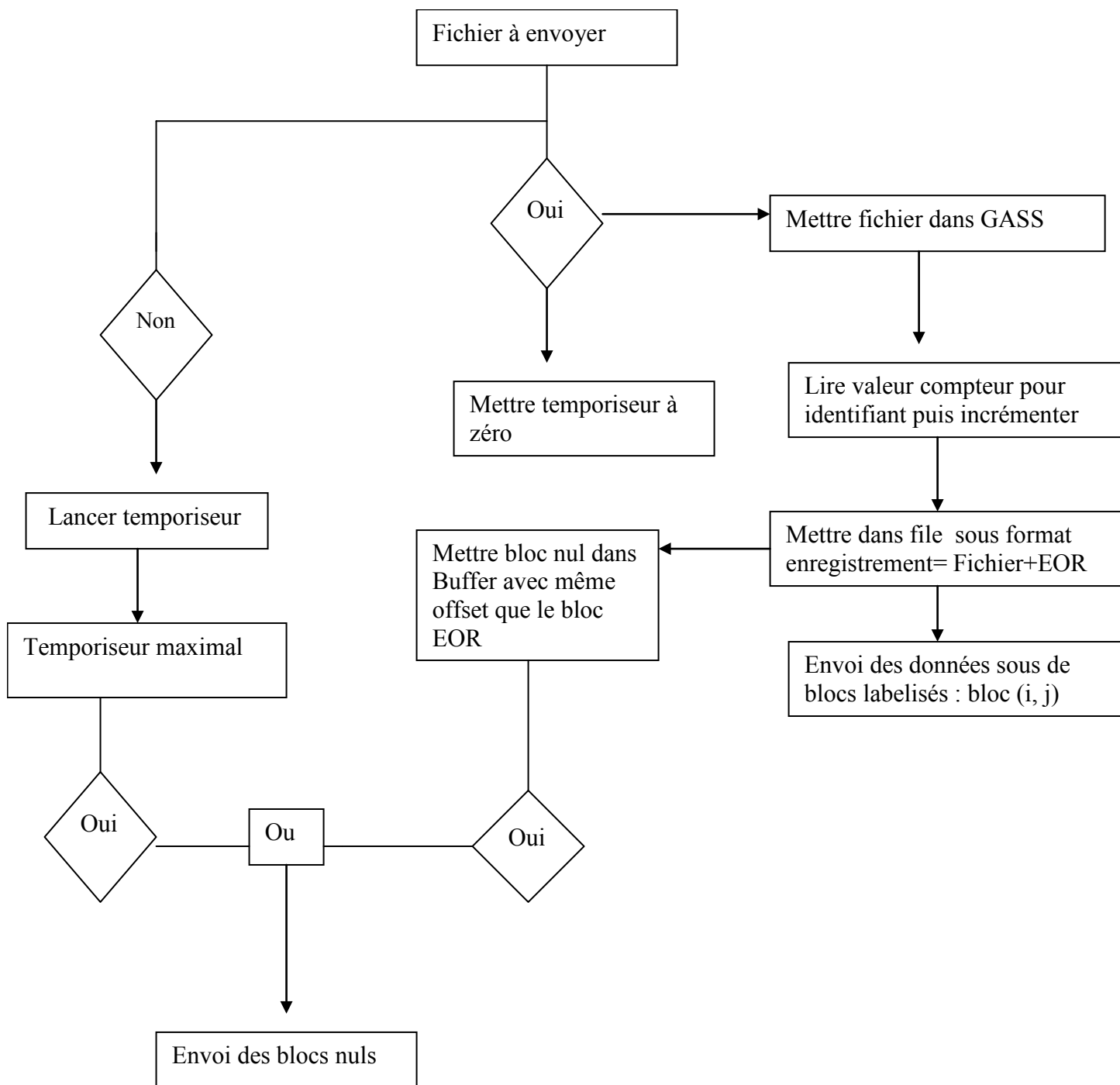


Figure 17. Déroulement de l'algorithme

IV.2.3.4.A Algorithme

Tant que (fichiers_candidat=vrai) alors

 Mettre temporiseur à zéro
 Mettre fichier dans GASS
 Lire valeur compteur pour identifiant puis incrémenter

 Pour $i \leftarrow 1$ à nombre_de_blocs faire

 Identifiant \leftarrow compteur
 Ouvrir bloc " i " en modification
 Ecrire bloc " i " (identifiant)
 Fermer bloc " i "
 Mettre bloc " i " dans la file
 Envoi bloc " i " dans buffer_reception " $j \leftarrow 1$ "

 Si ($j = \text{Nombre_canaux}$)
 $j \leftarrow 1$
 Sinon $j \leftarrow j + 1$
 Finsi

 Fin Pour

Mettre bloc nul dans buffer

 Compteur \leftarrow compteur + 1

Fin tant que

 Lancer temporiseur

Si ((Buffer_plein = Vrai) || (Temporiseur = max)) alors

 Pour $i \leftarrow 1$ à Nombre_blocs_nuls
 Mettre bloc_nul " i " dans buffer " $j \leftarrow 1$ "

 Si ($j = \text{Nombre_canaux}$)
 $j \leftarrow 1$

 Sinon $j \leftarrow j + 1$

 Finsi

FinPour

Fin

IV.2.3.4.B Commentaires

Notre algorithme que nous venons de mettre en place permet de régler les problèmes constatés dans le fonctionnement du protocole. Ces problèmes étaient principalement liés en l'absence de méthodes pour faire la différenciation des fichiers et à la gestion des ressources allouées. Ainsi, grâce au mécanisme de labellisation des blocs sous format bloc (i, j) introduit dans l'algorithme nous avons pu résoudre le problème de la différenciation des fichiers au niveau des canaux. En outre, à la réception des fichiers, le système de stockage sera en mesure de bien identifier tous les fichiers pour les mettre dans les différents emplacements prévus. Aussi, grâce à la mise en place de la file la concurrence sur les ressources allouées est résolue. En effet, grâce à la file tous les fichiers sont envoyés sous la forme d'enregistrement d'un seul et unique fichier. Donc, les ressources sont allouées à cet unique fichier qui est en réalité la concaténation de plusieurs fichiers. Par conséquent, notre algorithme optimise le temps de transfert des fichiers en supprimant le délai d'attente des fichiers qui était liés principalement en l'absence de méthodes de différenciation et de gestion des ressources.

Cependant, l'algorithme soulève quelque part des problèmes principalement liés à l'occupation de l'espace disque. En effet, le buffer et la file doivent occuper un certain espace disque. Donc, pour le fonctionnement de l'algorithme il faut un système de stockage assez grand qui puisse tenir en compte le buffer mais aussi la file.

Conclusion et perspectives

A travers les quatre chapitres de ce mémoire, nous avons pu mettre en évidence l'intérêt des grilles de calcul dans le monde scientifique. Ainsi, à travers une mise en place de la grille de calcul avec Globus et une étude de performance du protocole Gridftp nous avons pu montrer que les grilles de calcul reste aujourd'hui une technologie nécessaire pour le développement de la recherche dans un monde où les besoins de calcul deviennent de plus en plus grands. En effet, les technologies vont encore évoluer, mais les premières générations de grilles permettent déjà de résoudre

des problématiques, jusque là insolubles, de prise en compte dans des délais et des coûts raisonnables de traitements coopératifs ou intensifs sur des données massives distribuées. Cela, pour dire que des travaux doivent encore être menés le sens de l'optimisation de la technologie. Ainsi, nous avons pu mettre en place un algorithme allant dans le sens de l'optimisation du protocole Gridftp en permettant d'effectuer des transferts dans les meilleurs délais.

Comme perspectives, nous allons mettre en place une implémentation de notre algorithme afin qu'elle soit une contribution au développement de Globus. Pour cela un ensemble de procédures est à suivre.

- Soumission de l'algorithme à la communauté Globus.
- Etude et validation de l'algorithme par la communauté Globus.
- Mise en place d'une équipe (composée de chercheurs de la communauté et notre équipe de recherche) pour l'implémentation.
- La mise en disposition du code expérimentale pour l'insertion du code de l'algorithme.

Comme autre perspective, nous nous proposons de mettre en place un système de gestion des priorités des fichiers à transférer. En effet, nous sommes dans les environnements de grille de calcul et suivant les besoins des organisations virtuelles des fichiers peuvent être plus urgents à transférer que d'autres.

REFERENCES

- [1] La recherche – Les défis des supercalculateurs, Mai 2007, numéro 408.
- [2] I. Foster, C Kesselman, S Tuecke, "The anatomy of the Grid: Enabling Scalable Virtual Organisations", International Journal of Supercomputer Applications, March, 2001.
- [3] I. Foster," Globus Toolkit Version 4: Software for Service-Oriented Systems», IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, July 2006.
- [4] W.Allock (editor),"GridFTP Protocol Specification", Globus Grid Forum Recommendation GFD R.0201, March 2003.
- [5] W. Allock, J Bresnahan, R Kettimuthu, M. Link, C. Dumittrescu, I. Raicu, I. Foster "The Globus Striped GridFTP Framework and Server ", Proceedings of Supercomputing 2005 (SC05), November 2005.
- [6] Alliance Globus (infos, téléchargement, documentation en ligne) :
- [7] W.Allock, "GFD 20: GridFTP: Protocol Extensions to FTP for the Grid", April 2003.
- [8] Igor Mandrichenko,"GFD 21: GridFTP Protocol Improvements", July 11, 2003.
- [9] Ralph Niederberger, William Allcock, "GFD 83: Firewall Issues overview", August 16, 2006.
- [10] Ian Foster, Dan Fraser, John Bresnahan, "GridFTP Pipelining",TERAGRID 2007 CONFERENCE, MADISON.
- [11] Romaric David, "paper 46 : Une grille de calcul pour la recherche", 12 Octobre 2003.
- [12] Toni SOUEID, "Thèse: Grilles de calcul : Etat de l’art", Juin 2003.
- [13] J. Postel, J. Reynolds" RFC 959: FILE TRANSFER PROTOCOL (FTP) ", October 1985.
- [14] M. Horowitz, "RFC 2228 - FTP Security Extensions", October 1997.
- [15] TOP500 Team, "TOP500 Report ", July 2007.

- [16] "RFC793 - Transmission Control Protocol «, September 1981.
- [17] Parvin Asadzadeh, Rajkumar Buyya¹, Chun Ling Kei, Deepa Nayar, and Srikumar Venugopal, "Global Grids and Software Toolkits: A Study of Four Grid Middleware Technologies".
- [18] Xavier Jeannin (CNRS UREC), " Glite : Vue d'ensemble de l'architecture logicielle", JTR'2006, 16-18 octobre 2006, Lyon.
- [19] Philippe MARQUET, Pierre BOULET " Outils pour le calcul scientifique à haute Performance», Mai 2001.
- [20] Pierre Lombard, "NFSP : Une Solution de Stockage Distribué pour Architectures Grande Échelle",10 décembre 2003.
- [21] Abdelkader Lahmadi, Laurent Andrey, Olivier Festor " Évaluation de performance des architectures de gestion de réseaux: état de l'art et perspectives", Rapport de Recherche, Juin 2005.

Liste des figures

FIGURE 1.	ARCHITECTURE D'UNE GRILLE.....	20
FIGURE 2.	VUE D'ENSEMBLE DES COMPOSANTS DE GLOBUS.....	30
FIGURE 3.	AUTHENTIFICATION ET AUTORISATION.	31
FIGURE 4.	ARCHITECTURE DE GESTION DE RESSOURCES DE GLOBUS..	33
FIGURE 5.	MODELE CONCEPTUEL DE MDS.....	34
FIGURE 6.	ENTETE DE BLOC FTP.....	48
FIGURE 7.	ENTETE DU BLOC E-BLOCK DE GRIDFTP.....	50
FIGURE 8.	ETUDE COMPARATIVE DE GRIDFTP ET DE FTP.....	59
FIGURE 9.	LE TRANSFERT PARALLELE.....	60
FIGURE 10.	PARAMETRAGE DES BLOCS.....	61
FIGURE 11.	PARAMETRAGE DE LA TAILLE DES BUFFERS.....	62
FIGURE 12.	PARAMETRAGE DE LA TAILLE DES BLOCS POUR 4 CANAUX..	63
FIGURE 13.	PARAMETRAGE DE LA TAILLE DES BUFFERS POUR 4 CANAUX	64
FIGURE 14.	FENETRE DE TRANSFERT.....	67
FIGURE 15.	TRANSFERT A TRAVERS UN SEUL CANAL.....	68
FIGURE 16.	TRANSFERT A TRAVERS 4 CANAUX.....	68
FIGURE 17.	DEROULEMENT DE L'ALGORITHME.....	74

Liste des tableaux

TABLEAU 1.	PARALLELISME VS DISTRIBUTION.....	12
TABLEAU 2.	TABLEAU DES ADRESSES DES NŒUDS.	36
TABLEAU 3.	TABLEAU DES NŒUDS.....	41
TABLEAU 4.	TABLEAU RECAPITULATIF DES STRUCTURES.....	45
TABLEAU 5.	TABLEAU RECAPITULATIF DES MODES DE TRANSFERT.....	47
TABLEAU 6.	TABLEAU DE SYNTHESE DES SYSTEMES DE TRANSFERT.....	52
TABLEAU 7.	TABLEAU DE METRIQUES.....	55
TABLEAU 8.	PROBLEMES DU GRIDFTP.....	66

