الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N° Réf :……

Centre Universitaire
Abd elhafid boussouf Mila
Institut des Sciences et de la Technologie
Département de Mathématiques en Informatique

Mémoire préparé En vue de l'obtention du diplôme de Licence

En : - Filière : Informatique

Simulation d'une grille avec ses services en Gridsim

Préparé par :

- Lala Bouali Ridha
- Dehimi Mohamed El-Ghazali
- Mehal Meriam
- Labraoui Kamilia

Encadrer par : Ali Widad

Année universitaire : 2014/2015

Dédicaces

Avec énorme plaisir, un cœur ouvert et une immense joie, que je dédie notre travail à mes très chers, respectueux et magnifiques parents qui m'ont soutenue tout au long de ma vie

A ma Mère Chahrazad

Et Mon père Yahia

Ma grand-mère Ghania

Ainsi qu'à ma sœur Rima, mes frère Youness et Aymen et le petit Adam

A tous ma famille

A mes amis Mohamed El-Ghazali, Salim, Ahmed, Hassanne, Hmida, Amine.

A Tous Mes Collègues d'étude surtout Bouchouk Hadda qui m'ont aidé moralement et réellement a m'ais études surtouts cette année.

A notre ENCADREUR Mme : Ali Widad.

A toutes les personnes qui m'ont aidé et encouragé tout au long de mes études.

Ridha.

Dédicaces

Je dédie ce mémoire :

A mes très chers parents, pour leur soutien, leurs sacrifices, leurs encouragements qui m'ont permis d'arriver

Au bout de ce travail,

A ma mère «FADILA» qui m'a éclairée mon chemin et qui m'a encouragé et soutenue toute au long de mes Études, je prie dieu de vous protège et vous guérit, a Mon père « SAAD (rabi yachfik a baba nchalah) » qui s'est sacrifié afin que

Rien n'en déroulement de mes études,

A mais chère soeurs : somia, hadjer, khawla, manal, salsabil

A mon encadreur Ali Widad,

Àmes très chers amis : hassan, ridha, salim, ahmed, hmida, chouki, djamal, amine,

A Tous les membres de ma famille, à mes profs et mes collègues de la promotion MI 2011/2012

A tous ceux et celles qui me sont chers.

Mohamed El-Ghazali.

Dédicaces

Au début, je transmets mes remerciements et louange à Dieu qui m'a aidé dans mes études et l'achèvement du mémorandum, qui m'aidé à surmonter tous les obstacles pour y mettre fin et a fait don chaque mentionné suit

Je dédie ce mémoire à mes chers parents ma mère malika et mon père mohammed pour leur patience, leur soutien et leur encouragement.

A mon grand-père elhadj, ma tonte dalila et pour mes frères.

À mes amies et mes camarades.

Sans oublier tous les professeurs que ce soit du primaire, du moyen, du secondaire ou de l'enseignement supérieur. Et mas encadreur Ali Widad

Kamilia.

Dédicaces

Je dédie ce modeste travail à mes parent Said et Sabrina, mon mari Radhoane, mes beaux-parents Ahmed et Samia, ma sœur Fadwa, mes frères Abdlhakim, Abdeldjalil et Ahmed Chawki, mes belles sœur Assia, Kenza et Yasmine mon beaux-frères Mehdi, mes copines Halima, Kawthar, Kenza, Iman, Kami sans oublier mes enseignants est spécialement ma encadreur Mme Ali Widad et tous ceux qui m'ont soutenue et m'ont aidé le long de mon chemin de cursus universitaire.

Je les remercie tous et leurs dis que dieu le tout puissant vous garde pour moi.

Votre fille, femme et sœur Meriem.

Remerciements

A l'issue de ce travail, nous tenons à remercier en premier lieu le dieu de nous avoir donné l'aide afin de réaliser ce modeste travail. Nous remerciement nos chers parent, frères sœurs et à toute la famille pour leu inestimable soutien. On tient à remercier notre encadreuse Mme Ali Widad de nous avoir suivis, orienté et soutenu durant toute cette année et durant la réalisation de notre projet de fin d'étude.

Enfin nous remercions toutes les profs qui enseigner de primaire jusqu'à la dernier année de l'université, et toute les personnes qui de près ou de loin on contribués à l'élaboration de cette étude

Nous souhaitons la réussite à tous les étudiants des sections Mathématiques et Informatique.

Résumé

Cette mémoire présente la grille de calcul comme option, bon marché et abordable, du point de vue de développement des applications, pour profiter le calcul à haute performance en utilisant des modèles environnementaux de simulation. La marche à suivre proposée permet l'agrégation des ressources de différents organismes augmentant le débit des investissements. En outre, des solutions de calcul numérique sont montrées qui réduit la charge élevée de communications inhérente aux systèmes distribués.

Mots clé: Simulation environnementale, grille de calcul, calcul numérique.

Abstract

This memory presents grid computing as a non-expensive and affordable option, from the point of view of application development, to leverage high performance computing using environmental simulation models. The proposed method allows the aggregation of resources provided by different organizations, thus allowing a substantial increase in the return of investments. Numerical computing solutions reducing the high communications load inherent to distributed systems are also shown.

Keywords: Environmental simulation, grid computing, numerical computing.

Table des matières:

Introduction générale

Les Grilles de calcul ont apparu comme des paradigmes populaires pour la génération suivante parallèle et ont distribué le calcul. Ils permettent l'accumulation de ressources distribuées pour résoudre des problèmes à grande échelle de la science, l'ingénierie et le commerce. Dans la Grille, les ressources sont d'habitude géographiquement distribués dans des domaines administratifs multiples, gérés et appartenant aux organisations différentes avec des politiques différentes et connectés par des réseaux de zone large ou Internet. Ceci présente un certain nombre de gestion de ressource et demande prévoyant des défis dans le domaine de sécurité, la ressource et l'hétérogénéité de politique, la tolérance aux pannes, changeant continuellement des conditions de ressource et la politique. La gestion de ressource et les systèmes de planification pour calcul de Grille.

Dans un environnement de Grille, ce n'est pas facile et même impossible d'exécuter l'évaluation de performance de planificateur dans une façon répétable et contrôlable comme des ressources et les utilisateurs sont distribués à travers des organisations multiples avec leurs propres politiques.

Dans cette mémoire nous allons expliquer c'est quoi la grille de calcul et quelles sont ses objective et ses inconvénients ainsi nous allons parler sur un outil de simulation des grilles cet outil s'appelle GridSim pour démontrer la pertinence de cet outil, nous avons simulé le grille de calcul avec ses services.

L'objectif de ce travail est d'être capable de mener une analyse du grille permettre une élévation de leur niveau de performance et de découvrir ce quelle optimal pour certain travaille ou non.

Chapitre I

État de l’art

Introduction

La recherche informatique, lors des dernières décennies, s'est portée sur la façon d'augmenter les performances des ordinateurs afin de répondre à une demande croissante de la part des utilisateurs. Avec les progrès accomplis dans l'architecture des processeurs, pour gagner en puissance de calcul, et avec les progrès en matière de réseaux, de nouvelles perspectives d'évolution s'offrent à la technologie informatique. Deux voies ont été explorées : le meta-computing et les techniques à base de clusters, qui ont conduit à la création des grilles de calcul.

Le terme « The Grid » ou « Grille de calcul », a été introduit pour la première fois aux Etats-Unis dans les années 1990 pour décrire une infrastructure de calcul répartie utilisée dans des projets de recherche scientifique et industrielle. Dans un des premiers documents qui définissent des concepts de grille de calcul « 1 », les scientifiques tentent de fournir une définition détaillée des objectifs, de la forme et de l'architecture des grilles de calcul.

1 Historique et définition

Le mot "Grille" vient de l'anglais "Grid" qui a été choisi par analogie avec le système de distribution d'électricité américain (Electric power grid), ce terme été répandu en 1998 par l'ouvrage de Ian Foster et Carl Kesselman « 12 ». En effet, une grille peut être vue comme un instrument qui fournit de la puissance de calcul et/ou de la capacité de stockage de la même manière que le réseau électrique fournit de la puissance électrique. La vision des inventeurs de ce terme est qu'il sera possible, à terme, de se "brancher" sur une grille informatique pour obtenir de la puissance de calcul et/ou de stockage de données sans savoir ni où ni comment cette puissance est fournie, à l'image de ce qui se passe pour l'électricité.

L'objectif de cette section est de définir ce que nous entendons par "Grille de calcul" La recherche autour des grilles de calcul est très intense, il existe de multiples définitions. Nous allons ici présenter quelques définitions des grilles de calcul.

1.1 Définition 1

Ian Foster est défini la grille de calcul dans « 3 » comme une « infrastructure matérielle et logicielle qui fournit un accès fiable, cohérent, a tout de pénétration élever et bon marche a des capacités de traitement et de calcule »

C'est-à-dire que cette infrastructure fournit un accès tel que l'utilisateur a besoin de s'assurer de recevoir un bon niveau de Performance de la part des composants de la grille , et il est fondamental que la grille fournisse des services stables et Réguliers , même en présence de systèmes hétérogènes avec la disponibilité des services à tout instant, sans se soucier de l'état de l'environnement.

1.2 Définition 2

Une autre définition de Ian Foster basé sur la première définition et l'approfondi par : Une coordination des Resource partagées, une résolution des problèmes d'organisation virtuelle, dynamique et multi-institutionnelle. « 11 »

Cette définition prend en compte le partage entre un ensemble d'individus et d'institutions et qui nécessite un contrôle rigoureux par les fournisseurs et les consommateurs en définissant clairement et précisément qui a le droit d'effectuer ce partage et sous quelles conditions et ce qui doit être partage.

En raison de son caractère relativement vague, d'autres définitions ont été proposées on « 1,2 » Toutes s'accordent toutefois pour énoncer que les grilles de calcul permettent de résoudre des problèmes complexes par la mise en commun de Resource informatiques (disques, processeurs, etc…)

2 Les Caractéristiques des grilles de calcul

Les spécificités des grilles de calcul rendent leur gestion délicate de par :

L'hétérogénéité: la première caractéristique des ressources constituant une grille est sans nul doute leur hétérogénéité. Qu'elles soient matérielles ou bien logicielles, les ressources sont souvent très différentes les unes des autres. Cette hétérogénéité impose des contraintes de portage de code, d'utilisation de langages Multi-platforms et d'utilisation de protocoles de communication standardisés. L'utilisateur doit de plus pouvoir utiliser la grille de manière transparente et homogène quelle que soit l'architecture de sa propre machine et l'architecture de la machine à laquelle il se connecte.

La multiplicité des domaines d'administration: les ressources sont géographiquement distribuées et appartiennent à différentes organisations indépendantes. Elles appartiennent donc à plusieurs domaines d'administration distincts, ayant chacun son propre politique de gestion et de sécurité en termes d'accès au réseau, d'accès aux données, d'authentification ou encore

de confidentialité. Ainsi, les personnes chargées d'administrer la grille n'auront pas forcément de privilèges particuliers sur les machines des diffèrent domaines d'administration. Il est donc indispensable de mettre au point des méthodes d'administration particulières ne nécessitant aucun privilège sur les machines cibles. Il est également indispensable que chaque administrateur local conserve ses propres privilèges sur les machines qui lui appartiennent.

L'aspect dynamique : du fait du grand nombre de ressources considérées, la défaillance d'une ressource (machine ou réseau) est un événement courant qui ne doit pas mettre en péril le fonctionnement de la grille. Le gestionnaire de ressources tout comme les applications doivent tenir compte de cet aspect et être capables de réagir rapidement à la perte ou à l'ajout d'un nœud. De la même manière, l'ajout de fonctionnalités logicielles doit pouvoir se faire, dans la mesure du possible, sans qu'il soit nécessaire d'arrêter ou de réinstaller l'ensemble des nœuds de la grille.

La gestion des ressources: la capacité de mise à l'échelle d'une grille est également une caractéristique à prendre en compte. En effet, une grille pourra être constituée d'une dizaine de ressources, tout comme de plusieurs milliers de ressources. Ce problème de dimensionnement pose de nouvelles contraintes sur les applications et les algorithmes de gestion des ressources. L'observation des ressource notamment, ne doit pas induire une surconsommation de ressources trop importante et ce quel que soit le nombre de nœuds de la grille.

Passage à l'échelle: Le passage à l'échelle (Scalability) est un aspect critique de toute application distribuée. Il représente sa capacité à croître avec le nombre d'utilisateurs et de services requis.

3 Les objectifs des grilles de calcul

L'objectif d'une grille de calcul est de concevoir une architecture informatique permettant de mettre à disposition des utilisateurs toutes les Resource dont ils ont besoin au moyen d'une interface simplifiée.

La complexité du réseau et des logiciels de gestion du système doit être invisible pour l'utilisateur. Il doit accéder aux ressources de façon transparente.

Ian Foster (le concepteur de Globus, logiciel qui sera traité plus tard) propose trois critères pour définir une grille de calcul :

Le premier critère est celui lié à la gestion des Resource du système : une grille de calcul est un système qui gère des Resource qui ne sont pas sujettes à un contrôle centralisé.

Chaque nœud d'une grille gère ses Resource de façon autonome selon des spécifications (stratégies de sécurité, facturation à l'utilisation, gestion des utilisateurs, etc…) qui lui sont assignées par le concepteur de la grille.

Un mécanisme permettant d'interroger chaque nœud doit être conçu pour connaître son état, sa charge de travail, ses caractéristiques systèmes et ses modes de fonctionnement, ainsi que ses caractéristiques matérielles.

Le deuxième critère est celui lié à l'utilisation de protocoles : une grille est un système implémentant des protocoles standardisés.

Une grille de calcul s'appuie sur une infrastructure réseau, généralement sous TCP/IP. En outre, les outils de son interface doivent être conçus ou adaptés afin de gérer la sécurité de la grille.

Le système doit également mettre en œuvre des mécanismes pour la découverte et le référencement des Resource, pour permettre de les localiser et les mettre à disposition.

Il s'agit donc de la mise en œuvre d'un annuaire (ou système d'information) pour référencer les Resource et pouvoir fournir des informations sur leur état ou leur disponibilité aux utilisateurs ou aux applications qui en auraient besoin.

Enfin, une grille de calcul est un système devant offrir une grande qualité de service, elle doit mettre à disposition des ressources fiables, elle doit garantir des temps de réponse acceptables, elle doit garantir la sécurité des accès aux données et la cohérence de celles-ci.

Dans cette perspective, une grille met en œuvre des mécanismes de duplication des données pour la tolérance de pannes, et le système doit veiller à ce que ces données dupliquées restent à jour de façon à en maintenir la cohérence.

4 Les applications des grilles de calcul

Nous présentons dans ce qui suit cinq principales classes d'applications utilisant le calcul de grille :

- **Supercalculateur réparti (distribue supercomputing):**une grille de calcul peut agréger un grand nombre de Resource fournissant ainsi, une puissance de calcul nécessaire a de nombreuses applications telles que la météorologie, l'aéronautique, etc…

- **Calcul haut débit (high-throughput computing):**la grille de calcul est utilisée pour ordonnancer un grand nombre de taches indépendantes, en exploitant les cycles processeur non utilisent.
- **Calcul sur demande (on-demand computing)**: ce type d'applications utilise les capacités de la grille pour satisfaire des requêtes à court terme qui ne peuvent être satisfaites par les Resource locales. Le défi principal dans ce type d'application réside dans la nature dynamique des requêtes et le grand nombre d'utilisateurs et de Resource.
- **Calcul sur données intensives (Data intensive computing) :** dans ce type d'applications, une grande quantité de données est traitée, stockée et générée. Le défi dans ce type d'application est celui de l'ordonnancement et la configuration d'un flux de données complexe et volumineux.
- **Calcul collaboratif (Collaborative computing)**: les applications collaboratives concernent principalement les interactions entre les humains dans des environnements de simulation en temps réel, ce qui constitue le principal défi dans ces applications.

Le tableau suivant donne un récapitulatif des principales applications des grilles de calcul avec leurs caractéristiques et quelques exemples :

Catégorie	Exemples	Caractéristiques
Supercalculateur reparti	Aéronautique météorologie,…	Problèmes nécessitant de la mémoire, CPU, etc…
Calcul haut débit	Problèmes cryptographiques,…	Utilisation des Resource oisives pour augmenter le débit agrège.
Calcul sur demande	Instrumentation médicale, détection des nuages,...	Intégration des Resource distantes avec le calcul locale, souvent pour un temps limite.
Calcul sur données intensives	Données physiques, Prévisions climatiques,…	Synthèse d'informations nouvelles provenant de diverses sources de données.

Synthèse d'informations nouvelles provenant de diverses sources de données.	Travail collaboratif, éducation, exploration de données,…	Communication ou travail collaboratif entre de multiples participants.

Tableau 1: Principales catégories d'applications sur les grilles de calcul

5 L'architecture de la grille

L'architecture des grilles de calcul est décrite en termes de couches, chacune offrant des fonctions spécifiques :

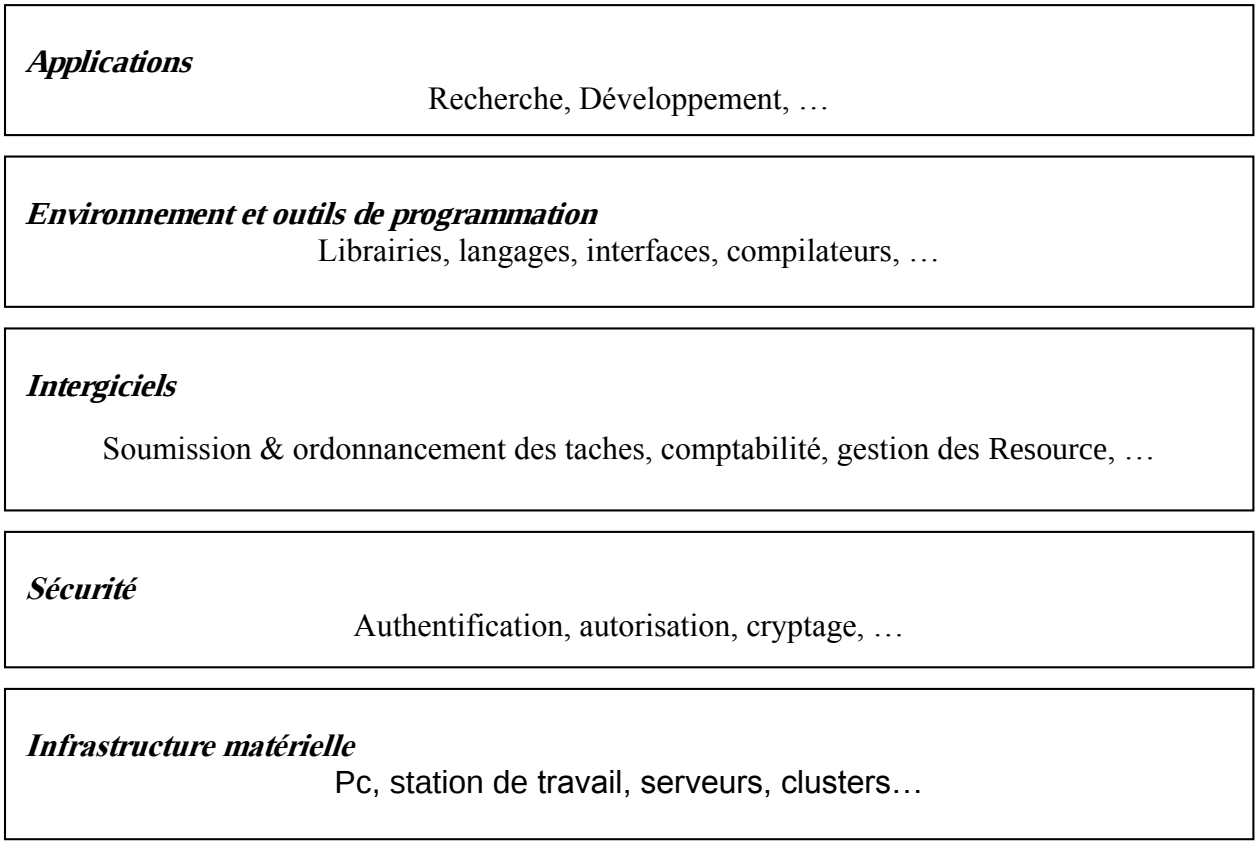

Figure 1 : L'architecture de la grille « 4 »

La couche la plus basse correspond à l'infrastructure matérielle. Elle comprend les Resource interconnectées à travers les réseaux LAN ou WAN. Cette infrastructure matérielle comprend notamment des PCs, des stations de travail, des grappes de calcul (clusters), des équipements de stockage, des bases de données, des équipements spéciaux …

La seconde couche fournit les mécanismes de sécurité nécessaires à la protection des Resource, dans les grilles de calculs, la sécurité est un souci beaucoup plus important que dans les environnements d'informatique répartie traditionnels.

Le nombre important de Resource et leur étendue géographique constituent un facteur important, ainsi il est fondamental de bien identifier les Resource à protéger et les degrés de protection nécessaires à appliquer.

Parmi les mesures de protection on trouve ceux normalement employés dans les réseaux (pare-feu, détection d'intrusion …) ainsi que ceux spécifiques aux environnements répartis et aux grilles (authentification, autorisation, single sign-on …).

La troisième couche fournit les intergiciels (middleware) nécessaires à la gestion des Resource, la coordination de l'accès, l'ordonnancement des tâches …

La quatrième couche regroupe tous les outils de développement permettant de concevoir des applications optimales pour la grille. On y trouve plus particulièrement des compilateurs, des librairies, des outils de conception d'applications parallèles ainsi que des interfaces de programmation ou API (découverte et réservation de Resource, mécanismes de sécurité, stockage …).

Enfin, **la dernière couche** regroupe les applications proprement dites qui sont de nature variée : projets scientifiques, médicaux, financiers, ingénierie …

6 Les déférentes topologies de grille

V. Berstis énumère les grilles d'un point de vue topologique en trois types par ordre croissant d'étendue géographique et de complexité : intragrilles (intragrids), extragrilles (extragrids) et intergrilles (intergrids).

- Intragrille (en analogie avec Intranet) : la plus simple des grilles est l'intragrille, composée d'un ensemble relativement limitée de Resource et de services et appartenant à une organisation unique. Les principales caractéristiques d'une telle grille sont l'interconnexion à travers un réseau performant et haut débit, un domaine de sécurité unique et maîtrisé par les administrateurs de l'organisation et un ensemble relativement statique et homogène de Resource.
- Extragrille (en analogie avec Extranet) : une extragrille étend le modèle en regroupant plusieurs intragrilles. Les principales caractéristiques d'une telle grille sont la présence d'un réseau d'interconnexion hétérogène haut et bas débit (LAN/WAN), de plusieurs domaines de sécurité distincts, et d'un ensemble plus ou moins dynamique de Resource. Un exemple d'utilisation est lors d'alliances et d'échanges « Business -to- Business » (B2B) entre entreprises partenaires.

- Intergrille (en analogie avec Internet) : une intergrille consiste à agréger les grilles de multiples organisations, en une seule grille. Les principales caractéristiques d'une telle grille sont la présence d'un réseau d'interconnexion très hétérogène haut et bas débit (LAN / WAN), de plusieurs domaines de sécurité distincts et ayant parfois des politiques de sécurité différentes et même contradictoires, et d'un ensemble très dynamique de Resource.

7 Les intergicieles des grilles de calcul

7.1 Les Fonctions des intergiciels

Voici les principales fonctions assurées par ces intergiciels :

- Ordonnancement (Scheduling) :

L'ordonnanceur doit connaître à tout moment la charge de chaque machine du Grid afin de pouvoir attribuer aux Resource les plus sous exploitées les taches de calculs.

Il supervise aussi du début à la fin le déroulement d'un job. Il peut le soumettre à nouveau s'il est brusquement interrompu et le terminer prématurément s'il se trouve dans une boucle infinie d'exécution.

- Réservation :

Cette fonction permet de réserver des Resource à l'avance dans le but de garantir une certaine qualité de service et de respecter certaines échéances.

- Services d'information et d'annuaire (Information and Directory services) :

Une grille est un environnement où le nombre et la charge des Resource sont dynamiques. Il est alors nécessaire de fournir des mécanismes permettant d'enregistrer et de découvrir ces Resource tout en identifiant leurs statuts.

- Service de nom (Naming Service) :

Comme dans tout système réparti, une grille devra permettre de référencer ses Resource d'une façon standard et uniforme, une grille de calculs devra fonctionner avec un large spectre de technologies matérielles ou logicielles.

Tout comme les utilisateurs du WEB ne se soucient pas de savoir si les serveurs WEB tournent sur une machine x86 ou Sparc ou bien si le système d'exploitation est Unix ou Windows, les utilisateurs d'une grille ne veulent pas se préoccuper des détails matériels et

logiciels de l'infrastructure, un environnement de grille idéal fournira ainsi un accès transparent aux Resource en cachant toutes les différences « physiques ».

7.2 La boîte à outils Globus

La Globus Alliance est une « Organisation Virtuelle » qui a pour but d'aider le monde scientifique, la recherche, les entreprises à créer des applications basées sur le Grid Computing.

Elle développe et essaie de standardiser la technologie qu'est le Grid Computing au moyen de son middleware vedette le « Globus Toolkit ».

Comme son nom l'indique c'est une boîte à outils (modules, bibliothèques) de type middleware développée en C et en Java, qui permet de créer une architecture de grille et de développer des applications basées sur la technologie du Grid Computing.

Le Globus Toolkit est à la base de nombreux projets dans le monde scientifique et des entreprises

Globus fournit les fonctionnalités et les services de base nécessaires pour la construction de grilles de calcul. Ainsi, nous trouvons des services et des mécanismes tels que la sécurité, la localisation et la gestion des Resource, la communication… « 10 »

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

La table suivante expose ces différents services « 5 »

Service	**Nom**	**Description**
Gestion de Resource	GRAM	Allocation des Resource et des processus
Communication	Nexus	Service de communication unicast et multicast
Sécurité	GSI	Authentification et autorisation
Information	MDS	Information sur la structure et l'état de la grille

Tableau 2 : Principaux services offerts par Globus.

8 Les Services de la grille de calcul

8.1 Service d'information grille

La grille est un environnement dynamique où la disponibilité et le type des Resource varient constamment.

Il est donc nécessaire de connaître avec précision la liste ainsi que l'état des machines connectées à la grille. Le service d'information grille (GIS) est l'entité qui gère ces informations en temps réel « 6 ». Il collecte des informations telles que :

- la liste des Resource de calcul et de stockage disponibles
- la liste des utilisateurs de la grille
- les services disponibles
- l'état de chaque ressource (taux d'occupation processeur, mémoire et disque)
- l'état du réseau d'interconnexion au moyen d'outils comme NWS (Network Weather Service) « 7» Service d'ordonnancement de tâches

Les applications pour grille sont découpées en tâches indépendantes, de façon à paralléliser les calculs pour exploiter au mieux toutes les Resource disponibles. Le service d'ordonnancement des tâches est chargé de sélectionner les Resource (nœuds de calcul) de la grille, susceptibles de les exécuter. L'objectif principal de cet ordonnanceur est de trouver la meilleure configuration pour que le temps d'exécution d'une tâche soit minimal.

Les principaux critères de sélection des Resource sont : le type et la vitesse des processeurs, la mémoire disponible, la taille de la mémoire virtuelle, l'emplacement géographique ainsi que la charge courante de la machine. Les outils de monitoring du GIS sont alors très sollicités pour connaître l'état des nœuds de la grille au moment de l'ordonnancement.

L'ordonnanceur tient également compte des droits d'accès dont dispose l'utilisateur qui s'adresse à lui. En effet, il se peut qu'il ne soit autorisé à accéder qu'à un sous-ensemble des Resource de la grille.

Pour pouvoir travailler, l'ordonnanceur a besoin de connaître certains détails relatifs aux tâches qu'il doit affecter à ces Resource. Pour cela, il existe des méthodes qui permettent de décrire les tâches ainsi que les Resource de la grille. Les détails pouvant ainsi caractérisé une tâche sont nombreux : le système d'exploitation ou bien l'architecture de la machine sur laquelle la tâche doit s’exécuter « 8 »

- la quantité de mémoire et l'espace disque nécessaire
- la puissance de calcul requise
- la priorité de la tâche

8.2 Service de gestion des données « 9 »

Le service de gestion des données se divise en deux catégories : l'accès aux données et la gestion des métadonnées, cette distinction permet de différencier le stockage des données utiles de celui des métadonnées.

Dans certains systèmes il est plus avantageux de regrouper les deux types de données (par exemple dans les bases de données distribuées).

Au contraire, dans d'autres systèmes, il peut être plus performant de séparer ces deux types de données, quitte à les stocker sur des Resource différentes.

- Le service d'accès aux données fournit des mécanismes pour gérer la localisation et le transfert des données. Il assure également la sécurité au niveau du dispositif de stockage, ainsi que lors des transferts. Il est capable de détecter et parfois de corriger les erreurs grâce à la mise en place de codes correcteurs. En fin, ce service est souvent pourvu d'autres fonctionnalités comme par exemple, la réservation des Resource de stockage.
- Le service de métadonnées quant à lui, assure la gestion des informations qui caractérisent les données utiles de la grille. Très souvent, la gestion des métadonnées existe à plusieurs niveaux dans les intergiciels. Les métadonnées les plus connues sont celles concernant les fichiers (droits utilisateurs, nom du propriétaire, date de dernière modification, taille, nombre de blocs utilisés, type de fichier, etc...). Ceci dit, il en existe d'autres comme par exemple les métadonnées liées à la réplication, qui assurent la correspondance entre toutes les instances d'un fichier répliqué et leurs emplacements physiques respectifs sur la grille (i.e. la ressource de stockage utilisée).

8.3 Service de réplication des données

La réplication est aujourd'hui largement utilisée dans les grilles. Elle consiste à créer plusieurs copies d'un même fichier sur des Resource de stockage différentes de la grille.

Cette technique permet d'augmenter la disponibilité des données, la charge étant alors répartie sur les différents nœuds possédant une réplique, le service de réplication met en œuvre une politique de cohérence particulière, il doit également maintenir un catalogue des répliques,

décidé quand créer et supprimer une copie, il s'appuie en général sur le système de monitoring des Resource du GIS, pour prendre ces décisions, par exemple, lorsqu'une donnée est partagée par plusieurs tâches ou applications, la ressource de stockage ainsi que le réseau, peuvent très vite saturer.

Le gestionnaire des répliques peut alors décider de créer une nouvelle copie sur un autre noud, afin de répartir la charge et permettre aux applications de s'exécuter plus vite, les grilles informatiques applications en question ne modifient pas la donnée (ex. serveur de vidéo à la demande).

Dans le cas contraire, il est moins avantageux de maintenir la cohérence entre les deux copies, plutôt que de n'avoir qu'une seule copie, toutefois, la réplication sert également à éviter de perdre des données sensibles lorsque les nœuds tombent en panne.

Il faut donc trouver le meilleur compromis en fonction du type de donnée et des applications. Tout cela fait partie des problématiques liées à la réplication et font l'objet d'études approfondies

8.4 Service de gestion de la concurrence

Dans un environnement distribué tel que la grille, la concurrence d'accès aux données est très forte. Pour une application utilisant MPI1, ce service n'est pas nécessaire car c'est l'application elle-même qui gère la concurrence d'accès à ses données, par contre, si l'application respecte le standard Posix2, c'est le système de fichiers qui se doit de garantir la cohérence des données.

Dans un environnement grille, celle-ci est gérée par une entité : le service de gestion de la concurrence. Il distribue des verrous sur des objets (métadonnées, données, entrée de cache, etc...).

Ce service est étudié en détail dans ce manuscrit. Dans les chapitres suivants, nous en présenterons l'état de l'art. Nous montrerons également les nouveautés que nous avons apportées par rapport aux modèles existants.

9 Principe de la simulation

9.1 Définition

Le mot simulation a donc deux significations. Le premier sens, « Faire paraître comme réelle une chose qui ne l'est pas », rejoint la notion d'illusion. L'exemple le plus fréquemment utilisé

est celui d'un joueur de football simulant une blessure pour obtenir une faute. Le deuxième sens se rapproche quant à lui de la notion d'imitation. C'est à dire de faire comme quelque chose mais sans vraiment le faire. Ce deuxième sens est celui que l'on rencontre le plus en informatique. Par exemple lors de la gravure d'un CD, il est possible de faire une simulation. Cette simulation consiste à tout faire comme si l'on allait graver le CD, mais en réalité le CD n'est pas graver. Cette technique perm et d'être qu'aucun facteur extérieur à l'action de graver elle-même ne pourra nuire à la gravure.

A moins d'être en marketing et vouloir des résultats montrant ce que l'on veut et non la réalité, c'est le deuxième sens qui s'applique en termes de simulation de grille. En effet, en simulant une application (souvent un algorithme) qui s'exécute sur un architecture répartie, il permet d'observer son exécution sans disposer réellement de l'architecture.

9.2 Intérêt de la simulation

Le premier avantage de la simulation, comme énoncé précédemment, est qu'il ne nécessite pas

l'infrastructure sur laquelle doit tourner l'application. Par exemple, un laboratoire veut acquérir une grille de calcul. Il travaille en parallèle sur un algorithme distribué pour cette grille. Grâce à la simulation, il pourra tester son algorithme même s'il ne possède pas de grille opérationnelle.

Le deuxième avantage de la simulation est de ne pas occuper les ressources présentes. Par exemple, le laboratoire possède déjà une grille mais ne dispose pas des ressources de sa grille car elle est déjà occupée par des calculs. Ainsi, on peut simuler l'exécution d'une application sans utiliser les ressources nécessaires à une vraie expérience. « 14 »

La simulation permet également une meilleur maîtrise des ressources (réseau, charge CPU, charge mémoire), en effet un tiers ne peut pas intervenir dans le système et occuper des ressources. De même il ne peut y avoir de perturbations dans le système. Cela permet donc de mieux mesurer l'impact de tel ou tel algorithme sur l'application indépendamment des perturbations éventuelles d'un système réel.

Un point important dans la simulation est la possibilité de reproduire l'expérience à l'identique et en ayant strictement les mêmes résultats. Cela permet de prouver l'efficacité d'un algorithme OU d'en trouver les failles. On est également sûr que l'expérience finira et donc qu'on aura des résultats à analyser

Enfin le dernier intérêt est la vitesse d'exécution des expériences. En effet comme les traitements sont fictifs, ils ne prennent aucun temps et on peut donc réaliser beaucoup plus de tests en un temps donné.

9.3 Taxonomie des outils de simulations « 15 »

Les outils de simulation sont nombreux et il est difficile d'en faire une présentation détaillée. Pour cela, la nécessité d'avoir une taxonomie qui permet d'uniformiser les terminologies pour une meilleure description est indispensable. Ainsi, Anthony Sulistio, Chee Shin Yeo et Rajkumar Buyyaont proposé une taxonomie largement adoptée.

Taxonomie des utilisateurs : l'outil de simulation peut être utilisé comme un simulateur ou comme un émulateur ; un simulateur est un outil qui représente un système réel par contre l'émulateur est un outil qui agit comme un système réel.

Taxonomie de simulation: en général, une simulation comporte trois propriétés :

- Présence du temps: indique si la simulation d'un système prend en compte le facteur temps. Une simulation statique ne considère pas le temps en tant qu'élément de simulation, contrairement à une simulation dynamique.
- Valeur de base: spécifie les valeurs que peut prendre une entité simulée. Une simulation discrète à des valeurs d'entités appartenant à un intervalle ainsi tandis qu'une simulation continue propose des valeurs d'entités appartenant à un intervalle infini.
- Comportement: la simulation peut se dérouler d'une manière déterministe (sans événements aléatoires) ; Ainsi la répétition de la même simulation rendra toujours les mêmes résultats contrairement à une simulation probabiliste (avec événements aléatoires) ; la répétition de la même simulation rend souvent des résultats différents.

Taxonomie de conception: Cela consiste à classer les outils de simulations par catégories basées sur les composants et les dispositifs nécessaires à la simulation:

- Moteur de simulation : la simulation peut être exécutée en mode séquentiel ou en mode parallèle ; Une simulation séquentielle est exécutée en utilisant un seul processeur, alors qu'une simulation parallèle ou distribuée est exécutée en utilisant plusieurs processeurs.
- Environnement de conception : détermine comment l'utilisateur utilise l'outil pour concevoir des modèles de simulation. Un langage fournit un ensemble de constructions

définies pour concevoir des modèles de simulation, alors qu'une bibliothèque fournit un ensemble de routines pour être utilisé avec un langage de programmation.

- Interface utilisateur: détermine comment l'utilisateur agit avec l'outil de simulation. Une interface de conception visuelle permet à l'utilisateur de créer un modèle de simulation beaucoup plus facile et plus rapide. Alors qu'une interface de conception non-visuel exige à l'utilisateur d'écrire des codes de programme ce qui exige plus de temps et d'effort.
- Supports système: fournit les dispositifs utiles et prêts à employer qui aident l'utilisateur à construire un modèle de simulation précis.

9.4 Quelques outils de simulation de grille

Dans la littérature beaucoup d'outils standards et spécifiques à l'application ont été établis parmi lesquels nous pouvons citer :

- **Bricks :** Il a été proposé et conçu pour des études de comparaisons d'algorithmes d'ordonnancement.
- **OptorSim :** conçu pour l'étude d'algorithmes d'ordonnancement traitant spécifiquement de la réplication ou de la migration de données,
- **GridSim ou SimGrid :** des outils de modélisation de ressources et réseaux d'une grille de calcul pour tester des algorithmes d'ordonnancement distribués.
- **MicroGrid :** permet aux développeurs d'exécuter les applications dans une grille virtuelle. Plus précisément, il a été conçu pour émuler Globus.

10 GRIDSIM

10.1 Définition

Gridsim est développé par Australien Melbourne University Raj Kumar Buyya Leadership Développement. Gridsim est un simulateur qui étudie l'ordonnancement dans les grilles. Il fournit un service complet pour créer différentes classes de ressources hétérogènes, applications d'utilisateurs. Le Resource peut être à un processeur ou à multiprocesseur simple avec de la mémoire partagée ou distribuée, en d'autres termes, la politique de Gridsim est flexible et extensible pour incorporer de nouveaux composants dans son infrastructure. Gridsim contrôle plusieurs abstraction (appelées les entités) : utilisateurs, brokers, ressources, services d'information des grilles (GIS), entrées/sorties. « 14 »

10.2 Les objectifs

L'objectif principal de projet GridSim est d'étudier les techniques d'allocation des ressources efficaces fondées sur l'économie de calcul par simulation. Pour simuler des millions de ressources et des milliers d'utilisateurs avec des exigences variées et étude évolutivité des systèmes, des algorithmes, l'efficacité des politiques et de la satisfaction des utilisateurs affectation des ressources. Gridsim également intéressé à explorer comment considérablement l'économie locale et le positionnement global (par exemple, le fuseau horaire) d'un rôle de jeu de ressources particulier à obtenir des emplois dans diverses situations et prix offre / demande. « 17 »

Les caractéristiques principales de la boîte à outils GridSim comprennent ce qui suit:

- la modélisation et la simulation des ressources hétérogènes, les utilisateurs, les Brokers et les modèles d'application.
- Les ressources peuvent être modélisés sous mode espace ou temps partagé.
- Il fournit des primitives pour la création de tâches d'application, l'affectation des tâches aux ressources, ainsi que leur gestion.

Conclusion

A travers ce chapitre, nous avons présenté la grille de calcul, selon sa définition et sa caractéristique et services, engendre par la représentation de l'architecture générale de la conception de la grille de calcul et sa simulation nous avons voire que la mise en œuvre de l'infrastructure de grille nous a permis de réellement fédérer des Resource de calcul, en offrant un service d'exécution des Jobs unifié, et que la grille de calcul est nécessaire pour la facilite de plusieurs applications importants à la recherche scientifique avec le moindre risque de sécurités.

En plus de ca la simulation de grille de calcul nous permit d'ignorer les problèmes de la conception réelle par la gestion de l'environnement de simulation.

Chapitre II

Conception

Introduction

Ce chapitre expose notre simulation de grille en utilisant le simulateur GridSim. Une courte description d'environnement de grille de calcul est présentée dans ce chapitre, ainsi que les diverses entités dans cet environnement et l'interaction entre eux. Le modèle de notre système, nos motivations, et les algorithmes de notre simulation sont discutés. Ce chapitre se termine par une présentation de notre structure de simulation.

1 Vue d'ensemble de l'environnement de Grille

Les principes entités dans une grille sont : L'Utilisateur, Le Broker, Service d'information et les ressources.

Les interactions entre ces entités sont illustrées à la Figure 1. Pour chaque utilisateur, il y aura un Broker. Les utilisateurs de la grille vont soumettre leur programme au Broker, à son tour il a trouvé les ressources appropriées et affecter le programme à une ressource.

Ici, une ressource peut être une ressource de calcul ou une ressource de stockage ou une ressource réseau (Cluster) ou tout périphérique capable de participer dans la grille.

Les ressources devraient mettre en œuvre au moins un mécanisme d'enquête qui permet la découverte de leur structure, l'état et la capacité. « 18 »

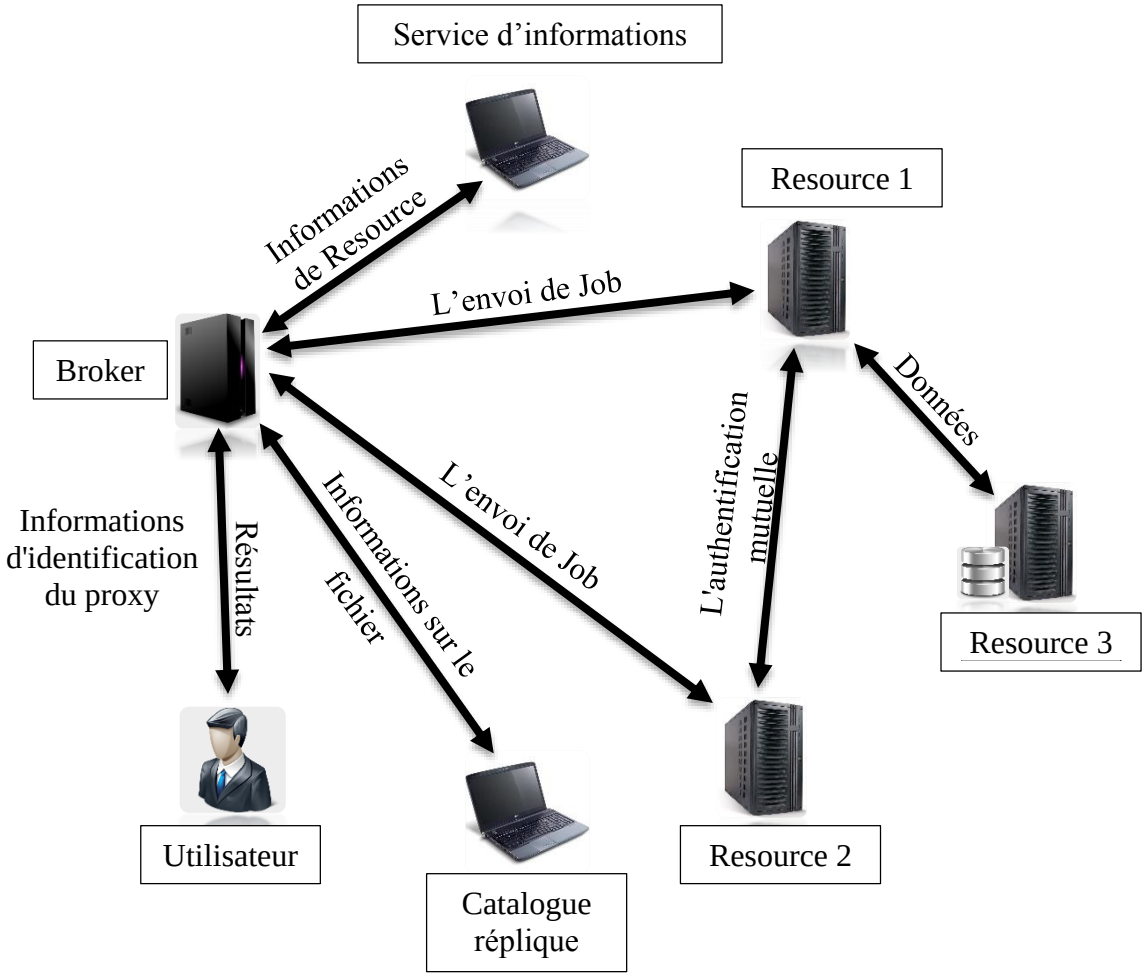

Figure 1 : Les interactions entre les entités de la grille de calcul

Le Broker contacter un Service d'Information pour trouver une ressource appropriée.

Le Service d'Information maintient les informations complètes de toutes les ressources (Comme sa capacité, l'état, les informations de contact ... Etc.) disponibles dans la grille.

Le Broker sélectionne une ressource et soumettre à lui le Job.

Après l'authentification, la ressource s'exécute le programme d'utilisateur.

Au moment d'exécution du programme d'utilisateur, il peut à son tour besoin d'une autre ressource pour terminer le travail (Comme : il besoin d'accédé aux données d'une autre machine ou-il besoin de faite vite l'exécution de job).

Si l'utilisateur possède les informations d'identification pour accéder à cette nouvelle Resource, puis la machine distante (Actuellement la machine en utilise) va être authentifié au nom de l'utilisateur (single sign-on) et qu'il va utiliser cette nouvelle Resource pour terminer le travail.

Catalogue de réplication pour localiser les données.

Le système de gestion de réplication contrôle l'emplacement et le temps des créations des copies de fichiers, et fournit des informations sur l'endroit où se trouvent les fichiers.

En d'autres termes, Le système de gestion de réplication maintient un mappage entre les noms logiques des fichiers et les collections à un ou plusieurs emplacements physiques.

La Figure 2 montre une architecture de systèmes de gestion de flux de travail pour la grille de calcul.

En général, une spécification de flux de travail est créée par un utilisateur à l'aide des outils de modélisation de flux de travail, ou généré automatiquement à l'aide de services d'information tels que MDS « 19 » et VDS « 20 » avant le moment de l'exécution.

Au moment de l'exécution, un moteur de contrôle de processus gère l'exécution de Job/tâche en utilisant middleware gille.

Il y a trois composants principaux dans un moteur de contrôle de processus : l'ordonnancement de flux de travail, transfert de données et la gestion de défaut.

L'ordonnancement de flux de travail découvre des ressources et alloue les tâche aux ressources appropriées pour répondre aux besoins des utilisateurs, tandis que le mouvement de données gère le transfert de données entre la sélection des ressources et la gestion des défauts fournit des mécanismes pour la manipulation de l'échec lors de l'exécution.

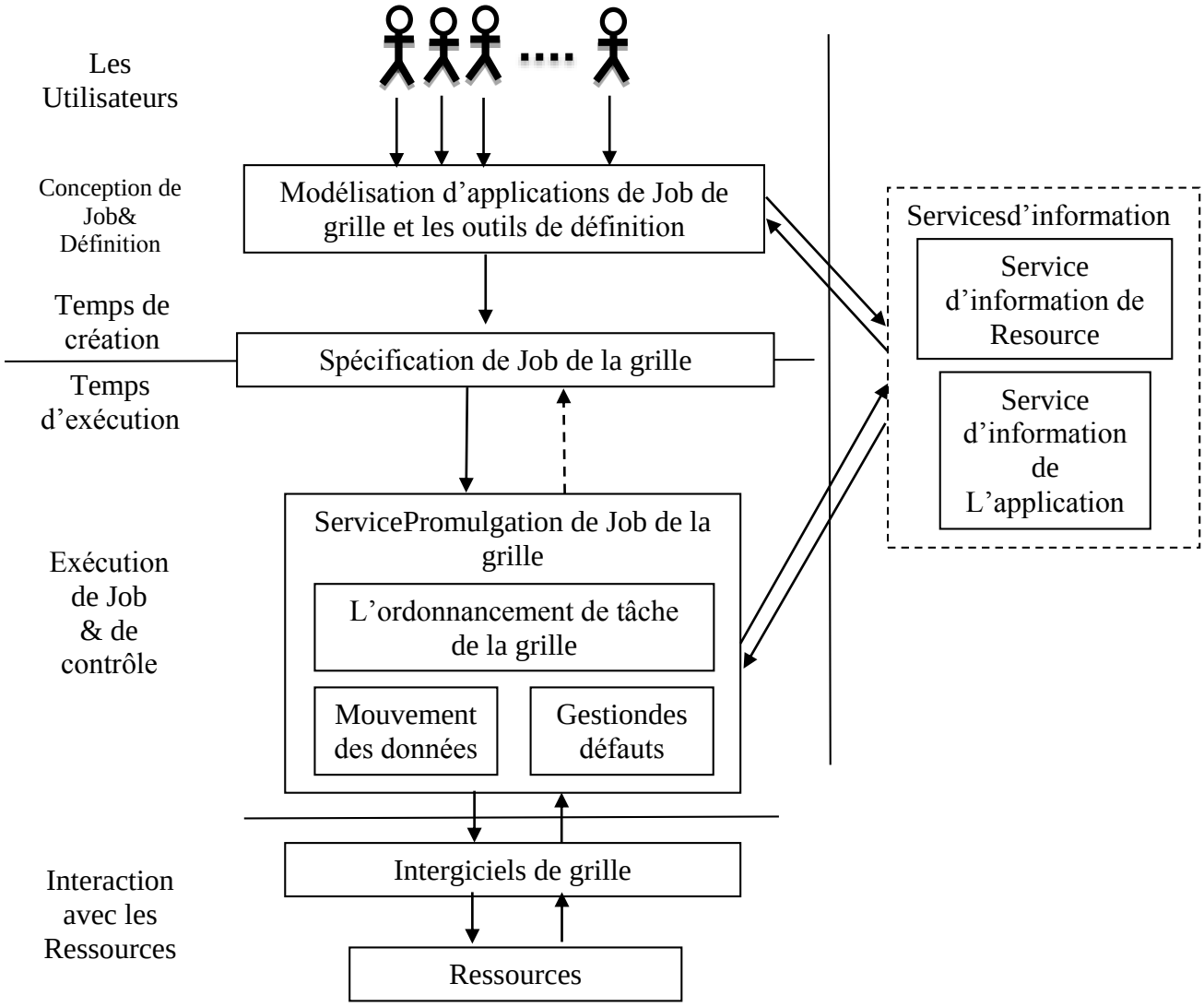

Figure 2 : Système de gestion de Job dans la grille

2 le simulateur Gridsim

2.1 L'architecture de Gridsim

Gridsim est une Boite à outils pour l'évaluation de l'ordonnancement des applications sur une grille. « 21 »

GridSim est basé sur SimJava « 22 », un usage général d'événements discrets package de simulation mise en œuvre en Java.

Nous avons conçu GridSim comme une multicouche architecture d'extensibilité. Ceci permet aux nouveaux composants ou des couches à être ajouté et en-intégrés dans GridSim facilement.

En outre, le GridSim architecture de couche de l'entreprise enregistre le modèle d'environnement de la grille de calcul.

L'ensemble d'architecture de Gridsim peut être illustré à la Figure 3.

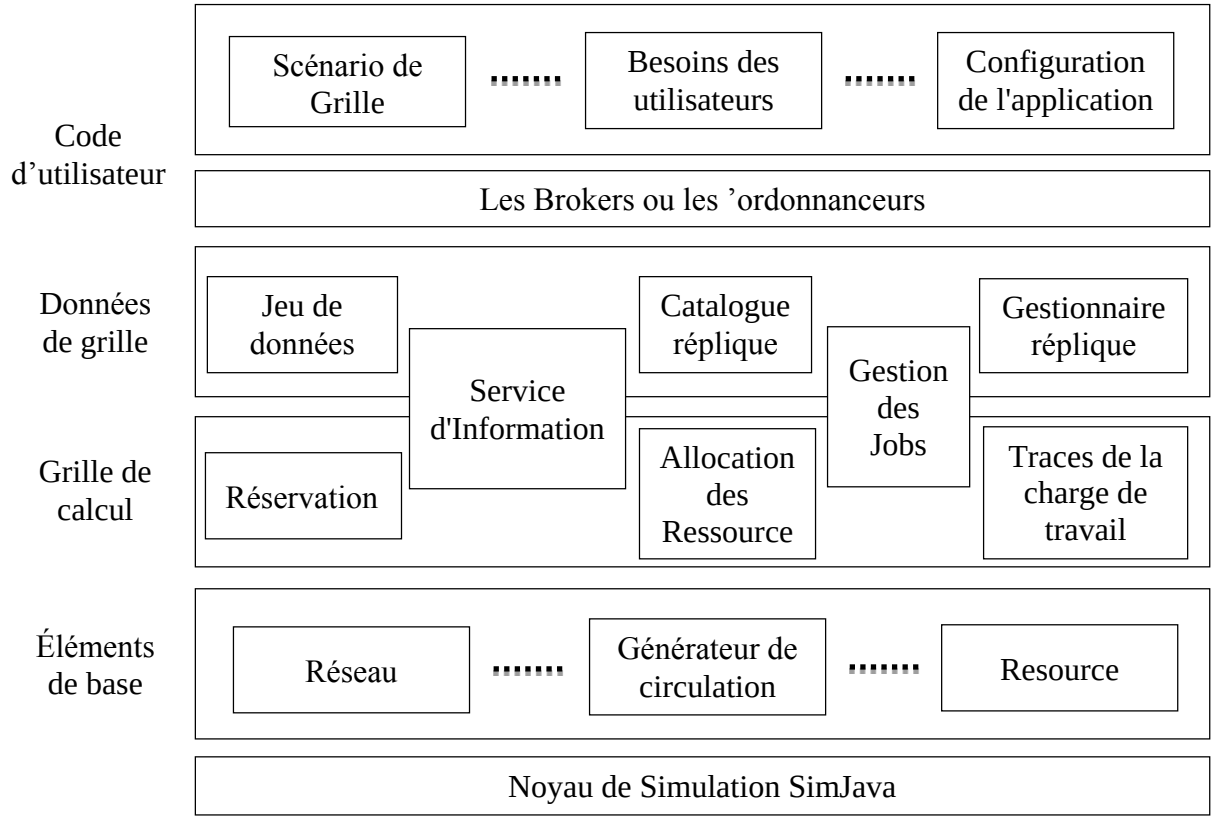

Figure 3 : Architecture de GridSim « 23 »

La première couche en bas de la Figure 3 est gérée par SimJava pour la manipulation de l'interaction ou les événements entre les composants de GridSim.

La deuxième couche est traitée avec les composants de l'infrastructure, comme le réseau et les Ressources matérielles.

Les troisièmes et quatrièmes couches sont concernées avec la modélisation et la simulation des grilles de calcul et grilles de données respectivement.

GridSim composée de Service d'information des grilles (GIS) et des Jobs, la description sont étendues dans la troisième couche d'intégrer de nouvelles exigences d'exécution de données grilles.

Les cinquièmes et sixièmes couches sont attribuées aux utilisateurs qui souhaitent écrire leur propre code dans GridSim.

2.2 Simulation à l'aide de Gridsim

La création d'une expérience dans GridSim exige toujours les étapes suivantes:

1. Créez une ou plusieurs entités de Resource à déférent configurations et capabilités.

2. Créez une ou plusieurs entités d'utilisateur à déférent besoins (applications et qualité de service des besoins).

3. Modélisez les applications par crée un nombre de Gridlets et définir toutes les paramètres associée avec les Jobs, les Gridlets à besoins de grouper dépend de modèle de l'application.

Modélisation des ressources

Dans GridSim nous pouvons créer des CPU (également appelés éléments de traitement (PE)) avec différents MIPS (millions d'instructions par seconde) ou SPEC-like notations.

Ensuite, un ou plusieurs PEs peut être mis ensemble pour créer une machine, telle une ou plusieurs machines peuvent être mises ensemble pour créer un Ressources.

Cette ressource peut être à un seul processeur, mémoire partagée multiprocesseurs (SMP), ou une mémoire distribuée cluster d'ordinateurs.

Ces ressources peuvent être gérées par les systèmes à temps -partagé ou l'espace partagé selon le type de la ressource.

Généralement, un seul PE ou SMP type de Resource est géré par un système d'exploitation à temps partagé utilisant la politique de planification round-robin et un cluster comme un Resource est géré par l'espace partagé Q-planificateurs utilisant différentes stratégies de planification comme premier arrivé-servi premier (FIFO), le Job court-servi premier (SJF), et ainsi de suite.

Modélisation d'Application

GridSim ne définit aucune modèle d'application explicitement spécifique. Il est au les développeurs (Des planificateurs et des Brokers) pour les définit.

Dans GridSim, chaque tâche indépendante peut être hétérogène en termes de temps de traitement et la taille des fichiers d'entrée.

Ces tâches/Jobs peuvent être créés et leurs exigences peuvent être définies via Gridlet objets.

Le GridSim prend en charge une large taille des protocoles de gestion de Gridlet et de services qui permet de mapper ou attribuer un Gridlet à un Resource pour l'exécution et manager à la même par le biais de son cycle de vie.

Modélisation d'utilisateur

Chaque utilisateur peut être modélisée avec différentes caractéristiques/exigences telles que:

- Types de Job créé par ex: heure d'exécution de la tâche, le nombre de réplications paramétrique, Etc…
- Politique d'ordonnancement par ex: le coût, le temps, ou la minimisation des deux.
- Taux d'activité par ex: combien souvent il crée un nouveaux Job.
- Fuseau horaire.

3 Diagramme de classes de notre simulation en GridSim

3.1 Structure du système proposé

La structure de notre système est un ensemble de : User, broker, GridResource, Gridlet, La Figure 4 illustre la relation entre les différentes classes.

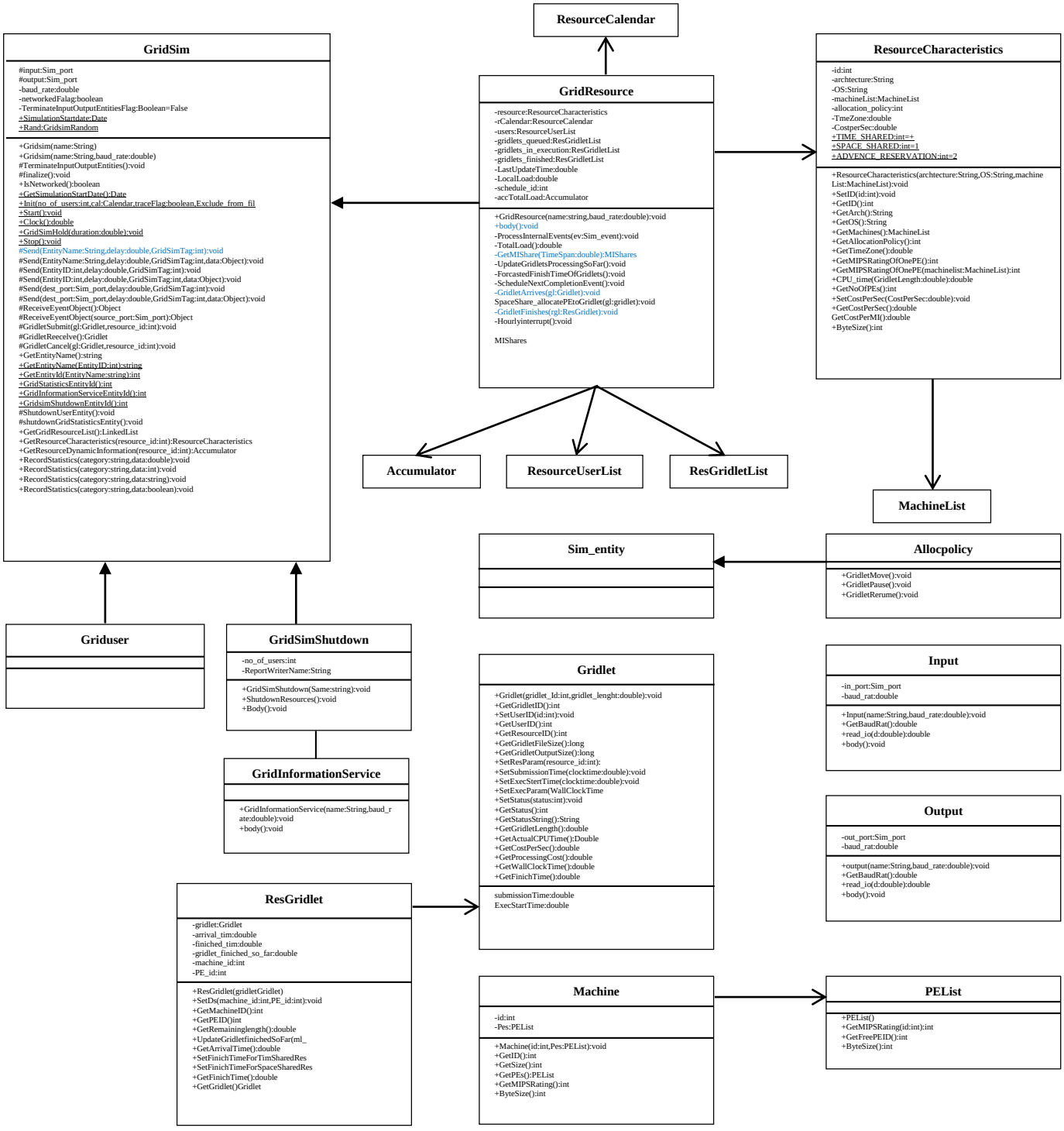

Figure 4: Diagramme de classe de notre simulation

Class gridsim.GridUser : permet de générer les utilisateurs. Un utilisateur de GridSim est responsable d'envoi des Jobs à un ou plusieurs ressources. Il contient un ou plusieurs Gridlets (Jobs) à traiter (Figure 5). Chaque utilisateur différé du reste des utilisateurs par :

- Types des Jobs créé, par exemple le temps d'exécution du Job, nombre de réplication. etc...
- Stratégie d'optimisation, par exemple minimisation du coût de transfert, du temps d'exécution, ou les deux.

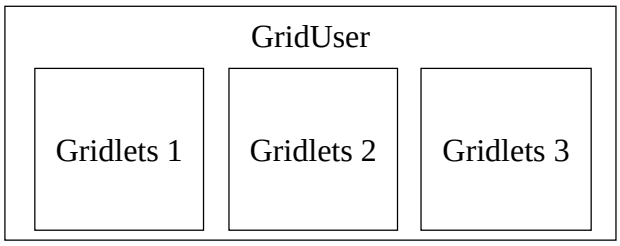

Figure 5 : Classe Griduser

Classe gridsim.Input : cette classe étend la classe eduni.simjava.Sim_entity. Cette classe définit un orifice à travers lequel une entité de simulation reçoit des données du réseau simulé. Il maintient une file d'attente d'événement de sérialiser l'en flux de données et délivre à son entité mère. Entrées simultanées peuvent être modélisés à l'aide de multiples instances de cette classe.

Classe gridsim.Output : cette classe est très similaire à la classe gridsim.Input et il définit un port par lequel une entité de simulation envoie des données sur le réseau simulé. Il maintient une file d'attente d'événement pour sérialiser les données-out-débit et fournit à l'entité de destination. Les sorties simultanées peuvent être modélisés en utilisant plusieurs instances de cette classe.

Class Broker : en GridSim Chaque utilisateur est relié à un Broker. Chaque Broker essaye d'optimiser la politique de son utilisateur et donc, on s'attend à ce que les Brokers fassent une concurrence en accédant aux ressources.

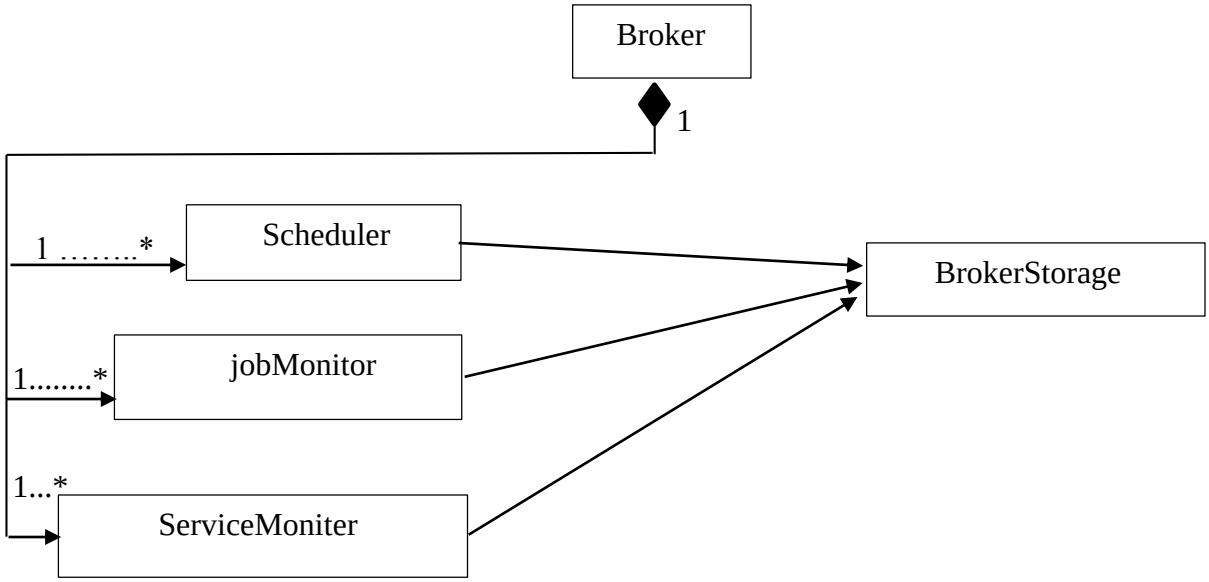

Figure 6: Class Broker

Classe gridsim.GridResource : elle étend la classe GridSim et gains de communication et de capacité d'entité simultanée. Une instance de cette classe simule une ressource avec les propriétés définies dans un objet de la classe gridsim.ResourceCharacteristics. Le processus de création d'une ressource est comme suit : tout d'abord créer des objets PE avec une convenable MIPS/SPEC notation, en suite les assembler pour créer une machine, enfin, en groupent un ou plusieurs objets de machine pour former une ressource. Une ressource ayant une seule machine avec une ou plusieurs PE est gérée comme un système de temps partagé à l'aide de l'algorithme d'ordonnancement de round robin. Une ressource avec plusieurs machines est traitée comme un cluster à la mémoire distribuée est géré comme un système d'espace partagé à l'aide du Politique de planification « premier arrivé premier servi ». (Figure 7).

Chaque ressource a un nombre de processeur, une vitesse des processus et une politique d'ordonnancement des processus internes.

Chaque Resource peut différer du reste des ressources par :

- Coût de traitement.
- Nombre de processeurs.
- Vitesse de traitement.
- Fuseau horaire.
- La vitesse des ressources et le temps d'exécution du Job sont exprimés en MIPS (Million d'instructions par seconde).

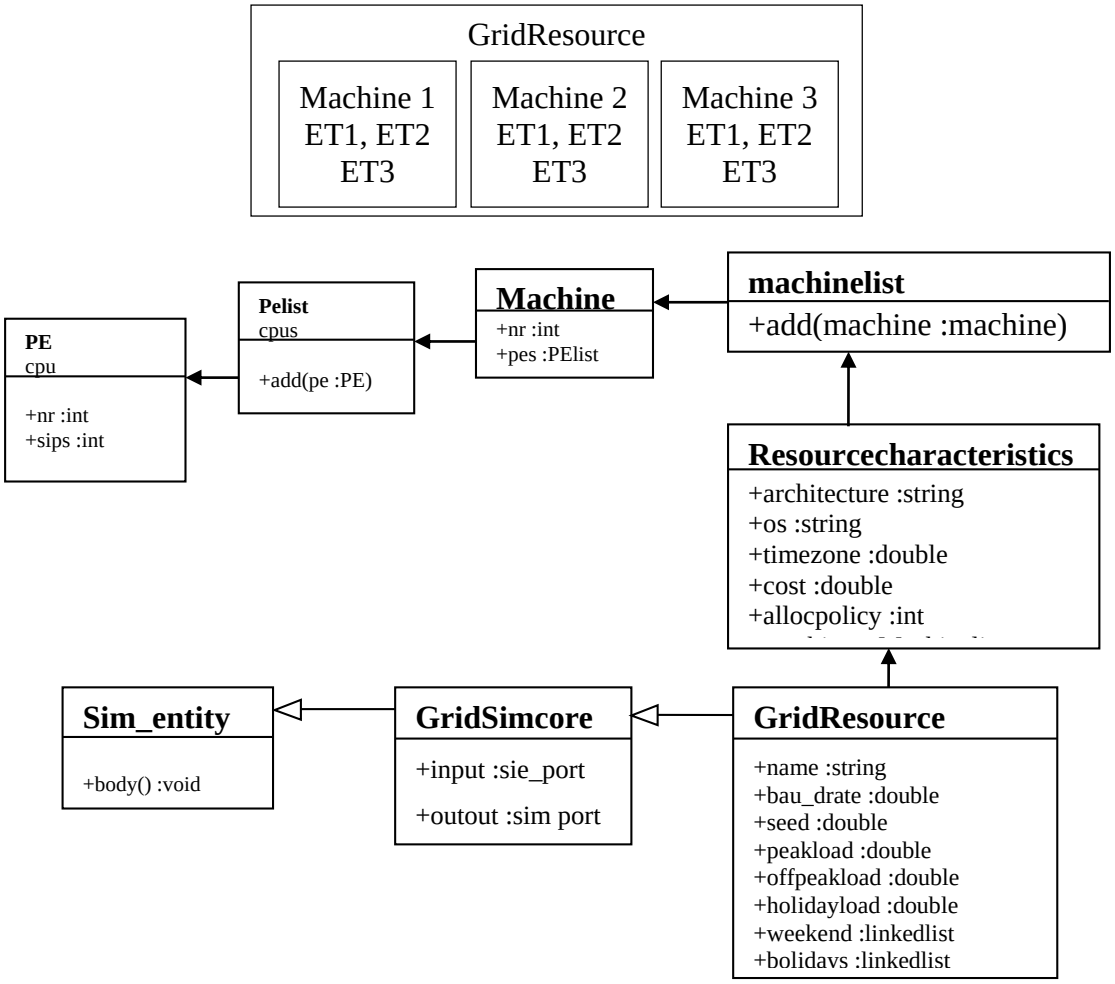

Figure 7 : La Classe GridResource

Classe gridsim.ResourceCharacteristics : elle représente les propriétés statistiques d'une ressource telle que l'architecture des ressources, SE, la politique de gestion (temps ou espace partagé), le coût, et le fuseau horaire à laquelle la ressource est située le long de la configuration des ressources.

Classe gridsim.ResourceCalendar : cette classe implémente un mécanisme pour prendre en charge la modélisation de chargement local sur les ressources de la grille qui peut varier selon le fuseau horaire, le temps, week-ends et les jours fériés.

Classe gridsim.Gridlet : un job est un ensemble de tâches atomiques qui doivent être exécutées sur un ensemble des ressources. Gridlet contient des informations de tâches/Job ou de données (Figure 8). Ces paramètres de base aident à déterminer le temps d'exécution en MI, le temps nécessaire pour transporter des fichiers d'entrée et de sortie entre les utilisateurs et les ressources distantes, et en retournant le Gridlets traitées à l'expéditeur avec les résultats.

Les données associées à chaque Job sont :

- Taille exprimée en MI (Million d'instructions).
- Temps d'exécution cumulé.
- Temps d'attente cumulé.
- Date d'arrivée.
- Date début d'exécution.

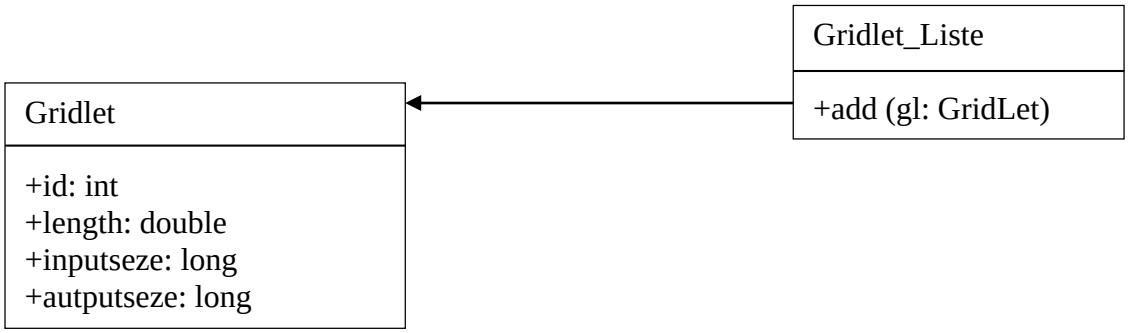

Figure 8 : La Classe Gridlet

Classe gridsim.ResGridlet : elle représente un Gridlet soumis à la ressource pour le traitement. Il contient objet Gridlet avec son heure d'arrivée et l'ID de la machine et PE qui lui sont allouées. Il agit comme un espace réservé pour maintenir le montant de partage de ressources allouées à diverses reprises pour simuler l'ordonnancement en temps partagé en utilisant les événements internes.

Classe gridsim.GridInformationService : une entité GridSim qui fournit des services d'enregistrement des ressources de grille, l'indexation et la découverte de services. Les ressources de la grille enregistrent leur volonté de traiter Gridlets en vous inscrivant eux-mêmes avec cette entité.

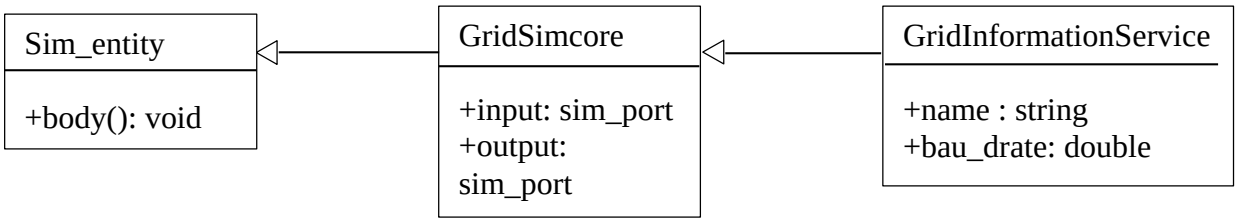

Figure 9 : La Classe GridInformationService

Classe gridsim.AllocPolicy : est une classe abstraite qui gère la politique d'allocation de GridResource interne. Nouveaux algorithmes d'ordonnancement peuvent être ajoutés dans une entité GridResource en étendant cette classe et mettre en œuvre les méthodes abstraites.

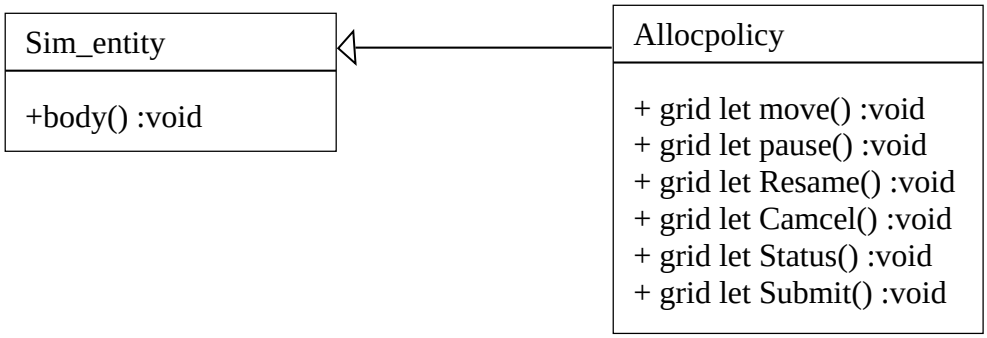

Figure 10 : La Classe Allocpolicy

4 Diagramme de séquence de notre simulation en GriSim

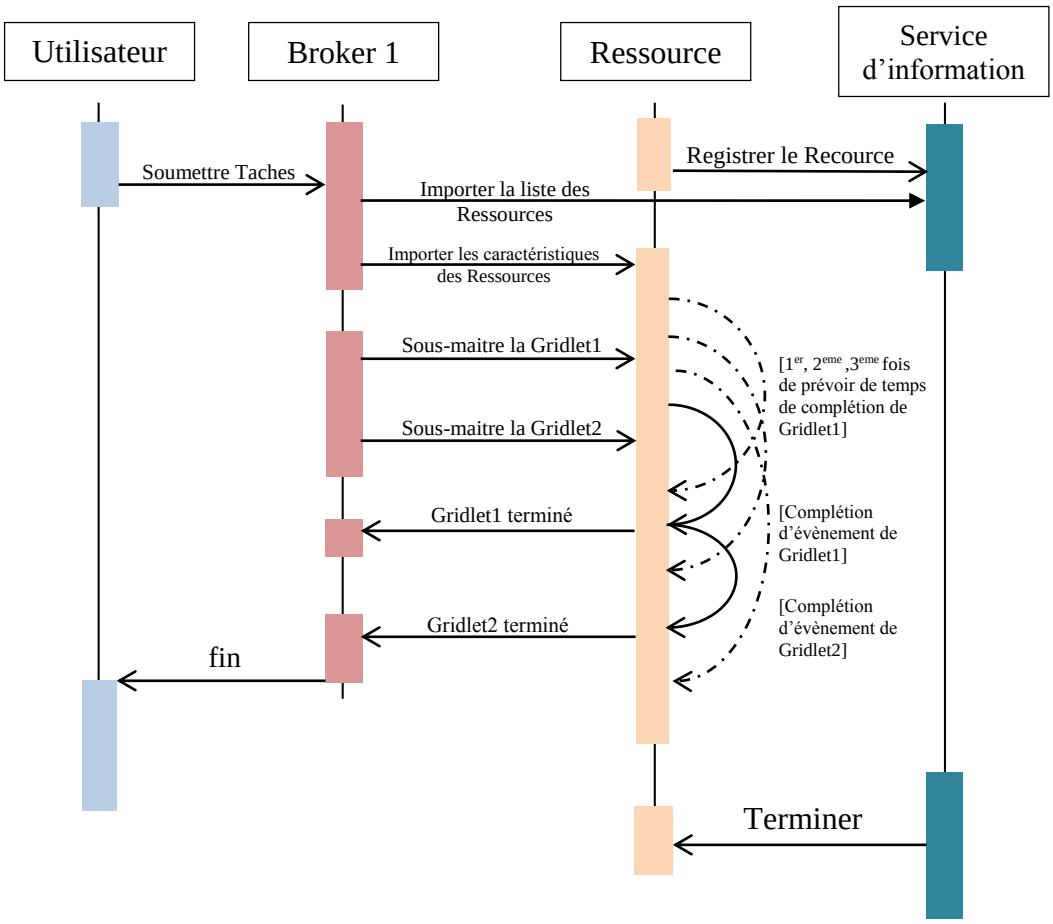

Figure 11 : Diagramme de séquence de Simulation

La Figure 11 montre les différentes interactions entre les entités de GridSim pendant la vie de la simulation, c'est un exemple de diagramme de séquence dans lequel le temps augmente de haut en bas, initialement, l'utilisateur soumet des Gridlets aux broker qui affecte chaque tâche soumise par l'un des utilisateurs à une ressource selon l'une des stratégies d'ordonnancement apprêt d'obtenue les caractéristique des ressources.

Les tâches sont insérées aux files d'attente de chaque ressource. Quand la ressource est prête à traiter la tâche elle la dépile et l'exécute. A chaque période de temps la ressource évalue son état courant, et envoie son information de charge au Service d'information.

5 Nos algorithmes de simulation

Pour simuler une grille de calcul il faut passer plusieurs étapes en appliquant un algorithme de simulation suivant :

NBR : Norme de ressources ;

Initialiser le package GridSim ;

Crée une ou plusieurs entités régionaux de GIS ;

Pour i aillent de 1 à NBR faire

Créer un objet de Machine List pour stocker un ou plusieurs Machines ;

Créer une machine avec son ID, le nombre de PEs et MIPSRating par PE ;

Créer un objet Ressource Caractéristiques qui stocke les propriétés d'une ressource (architecture, OS, la liste des machines, la politique d'allocation (time- ou space-shared) le fuseau horaire et son prix (G$/PE unité de temps)) ;

Créer un objet GridResource ;

Allouer cette ressource à une entité régionale GIS aléatoire ;

Fin pour

Crée une ou plusieurs entités utilisateurs de la grille et Obtient un ID pour chaque entité et crée une liste de Gridlets ou Tâches pour chaque utilisateur ;

Allouer cette ressource à une entité régionale GIS aléatoire ;

Construit la topologie du réseau pour toutes les entités ;

Démarre la simulation ;

En peut déduire un diagramme d'activité de simulation comme ce luit a la Figure 12.

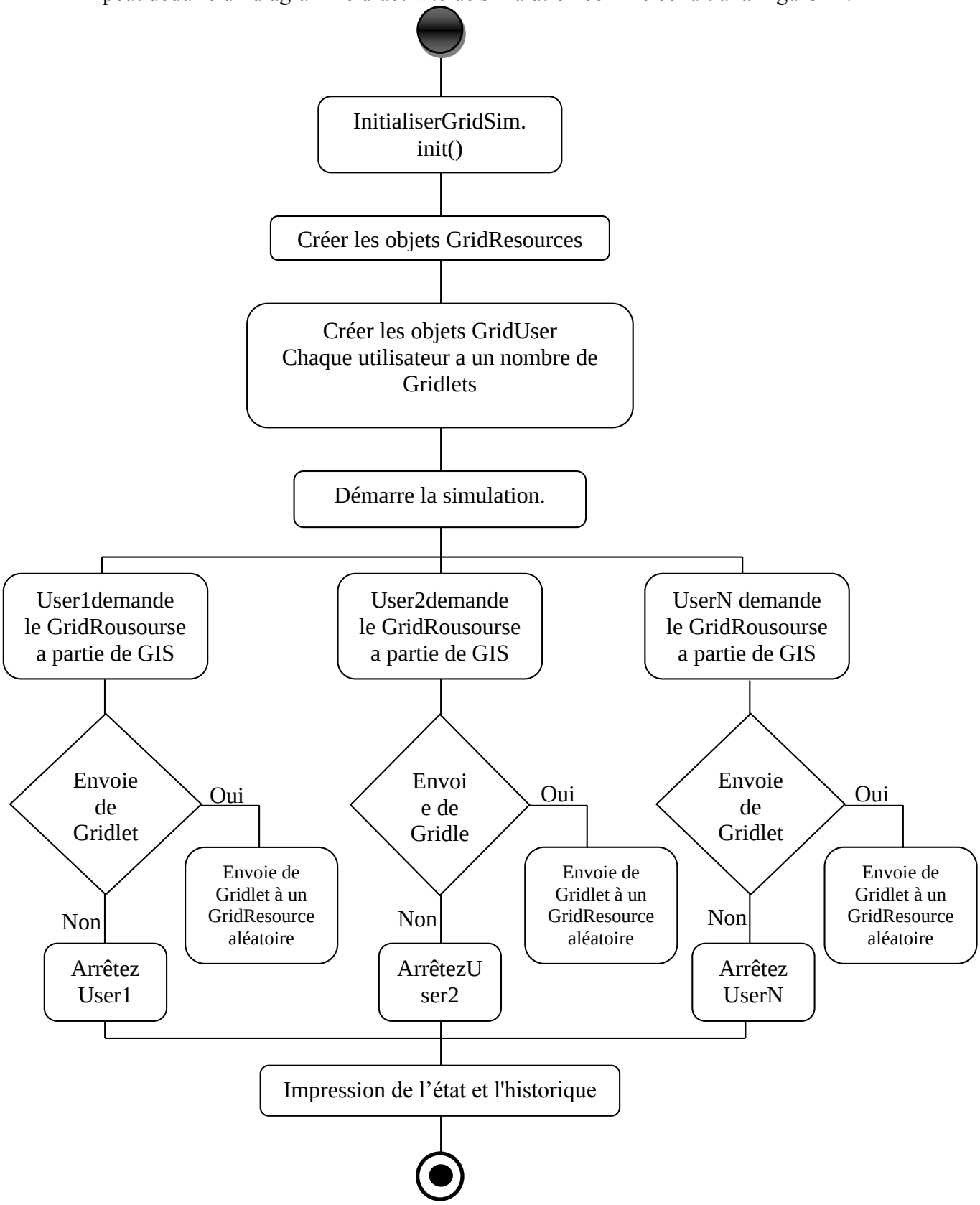

Figure 12 : Diagramme d'activité de la simulation

La simulation est toujours passée sur le cycle de vie selon les étapes suivantes :

- L’initialisation de GridSim.
- La création des entités des ressources.
- La création des entités des utilisateurs.
- Chaque utilisateur crée son Gridlets.
- Démarrage de la simulation.
- Chaque utilisateur recevoir les caractéristiques des ressources.
- Chaque utilisateur envoi ces Gridlet au une ressource aléatoire sinon il exit de la simulation.
- Quand tous les Gridlets d’un utilisateur finis il exit de la simulation.
- Quand tous les utilisateurs exit, GridSim affiché le résultat de simulation.
- En fin étendre tous les entités de simulation.

6 les algorithmes d’ordonnancement

6.1 Simulation de L’ordonnancement des ressources en temps partagé « 24 »

Un algorithme complet pour la simulation de L’ordonnancement et de l'exécution à temps partagé est illustré à la Figure 13. Si un événement nouvellement arrivés se trouve être un événement interne dont le numéro de tag est le même que l'événement le plus récemment prévu, il est reconnu comme un achèvement des événements des taches. Selon le nombre de Gridlets dans l'exécution et le nombre de PEs dans une ressource, GridSim alloue une quantité appropriée de la part de PE à tous les Gridlets pour la durée de l'événement en utilisant l'algorithme de la Figure 14. Il convient de noter que Gridlets partageant le même PE serait obtenir un montant égal de parts de PE. Le Gridlet terminé est renvoyé à son auteur (Broker ou Utilisateur) et retiré de l'ensemble de l'exécution. Horaires GridSim une nouvelle de l'événement interne à livrer dans les meilleurs délais d'achèvement prévue des Gridlets restants.

```
Algorithm: Time-Shared Grid Resource Event Handler ()
1. Wait for an event
2. If the external and Gridlet arrival event, then:
BEGIN /* a new job has arrived */
a. Allocate PE Share for Gridlets Processed so far
b. Add arrived Gridlet to Execution_Set
c. Forecast completion time of all Gridlets in Execution_Set
d. Schedule an event to be delivered at the smallest completion time
END
3. If event is internal and its tag value is the same as the recently scheduled internal event
tag,
BEGIN /* a job finish event */
a. Allocate PE Share of all Gridlets processed so far
b. Update finished Gridlet's PE and Wall clock time parameters and send it back to the
broker
c. Remove finished Gridlet from the Execution_Set and add to Finished_Set
d. Forecast completion time of all Gridlets in Execution_Set
e. Schedule an event to be delivered at the smallest completion time
END
4. Repeat the above steps until the end of simulation event is received
```

Figure 13 : Algorithme d'ordonnancement à temps partagé.

```
Algorithm: PE_Share_Allocation(Duration)
1. Identify total MI per PE for the duration and the number of PE that process one
extra Gridlet
TotalMIperPE = MIPSRatingOfOnePE()*Duration
MinNoOfGridletsPerPE = NoOfGridletsInExec / NoOfPEs
NoOfPEsRunningOneExtraGridlet = NoOfGridletsInExec % NoOfPEs
2. Identify maximum and minimum MI share that Gridlet get in the Duration
If(NoOfGridletsInExec<= NoOfPEs), then:
MaxSharePerGridlet = MinSharePerGridlet = TotalMIperPE
MaxShareNoOfGridlets = NoOfGridletsInExec
Else /* NoOfGridletsInExec>NoOfPEs */
MaxSharePerGridlet = TotalMIperPE/ MinNoOfGridletsPerPE
MinSharePerGridlet = TotalMIperPE/(MinNoOfGridletsPerPE+1)
    MaxShareNoOfGridlets = (NoOfPEs - NoOfPEsRunningOneExtraGridlet)*
                                MinNoOfGridletsPerPE
```

Figure 14 : Algorithme pour PE partage d'allocation à Gridlet en temps partagé

6.2 Simulation de L'ordonnancement des ressources en espace partagé

Le simulateur GridSim utilise des événements internes pour simuler l'exécution et la répartition des PEs à Job de Gridlet. Quand un Job arrive, le système de partage d'espace commence l'exécution immédiatement s'il y a un PE libre disponible, sinon, il est mis en attente.

Durant l'affectation de Gridlet, le temps de d'exécution de Job est déterminé et l'événement est prévu pour la livraison à la fin de temps d'exécution. Chaque fois un Job de Gridlet finitions, un événement interne est livré pour signifier la fin de Job de Gridlet prévue. Le simulateur de ressources libère alors le PE lui est attribué et vérifie si il y a d'autres Job en attente dans la file d'attente. Se il y a des Jobs en attente dans la file d'attente, il sélectionne un emploi convenable en fonction de la politique et l'affecte à la PE.

Un algorithme complet pour la simulation de la planification et l'exécution de partage d'espace est illustré à la Figure 15. Si un événement nouvellement arrivés se trouve être un événement interne dont le numéro de tag est le même que l'événement le plus récemment prévu, alors il est reconnu comme un achèvement de l'événement de Gridlet. S'il y a des Gridlets dans la file d'attente de soumission, puis en fonction de la politique d'attribution (par exemple la première Gridlet dans la file d'attente si la politique PAPS est utilisé), GridSim sélectionne un Gridlet appropriée de la file d'attente et l'affecte à la PE ou PE appropriée si plus d'un PE est libre. Voir Figure 16 pour une illustration de la répartition des PE à Gridlets. Le Gridlet terminé est renvoyé à son auteur (Broker ou Utilisateur) et retiré de l'ensemble de l'exécution. Horaires GridSim une nouvelle de l'événement interne pour être livrés à la date de réalisation de Job Gridlet prévue.

Algorithm: Space-Shared Grid Resource Event Handler()
1. Wait for an event and Identify Type of Event received
2. If it external and Gridlet arrival event, then:
BEGIN /* a new job arrived */
• If the number of Gridlets in execution are less than the number of PEs in the resource, then
Allocate PE to the Gridlet() /* It should schedule an Gridlet completion event */
• If not, Add Gridlet to the Gridlet Submitted Queue
END
3. If event is internal and its tag value is the same recently scheduled internal event tag,
BEGIN /* a job finish event */
• Update finished Gridlet's PE and Wall clock time parameters and send it back to the
broker.
• Set the status of PE to FREE.
• Remove finished Gridlet from the Execution Set and add to the Finished Set.
• If Gridlet Submitted Queue has Gridlets in waiting, then
Choose the Gridlet to be Processed() /* e.g., first one in Q if FCFS policy is used */
Allocate PE to the Gridlet() /* It should schedule a Gridlet completion event */
END
4. Repeat above steps until end of simulation event is received

Figure 15 : Algorithme d'ordonnancement à l'espace partagé

Algorithm: Allocate PE to the Gridlet(Gridletgl)
BEGIN
1. Identify a suitable Machine with Free PE
2. Identify a suitable PE in the machine and Assign to the Gridlet
3. Set Status of the Allocated PE to BUSY
4. Determine the Completion Time of Gridlet and Set and internal event to be delivered at the completion time
END

Figure 16 : Algorithme pour la répartition des PE à Gridlets

7 l'architecture général de notre système

Nous proposons que l'environnement de la simulation se compose de trois parties principales : l'utilisateur, Centre de planification et les ressources.

Et le Centre de planification est composé de scinque sous-modules illustré a la Figure 17.

Les utilisateurs soumettent l'exigence d'application au récepteur de la tâche, qui l'envoie ensuite les tâches au répartiteur des Tâche basée sur un algorithme de planification pour le mappage des tâches aux ressources.

Le Répartiteur des Tâches ensuite distribue les tâches à un Resource spécifiques en fonction de l'algorithme sélectionné.

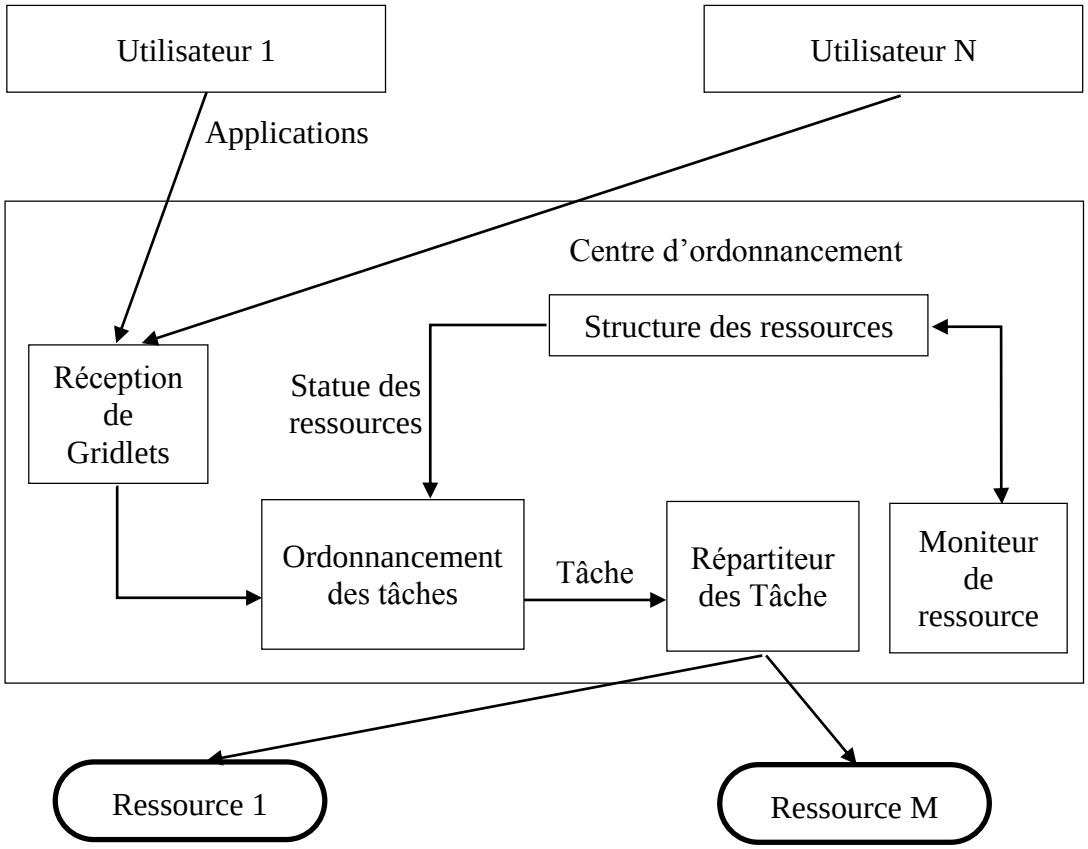

Figure 17 : La structure de notre simulation de l'environnement de la grille

Conclusion

En raison de la non-reproductibilité de l'environnement de la grille, la limitation arrive lorsqu'il effectue l'analyse des performances de grille dans un environnement réel. Par conséquent, l'outil de simulation de grille est utilisé comme un important outil de recherche. Nous avents propose Grisim.

A l'aide de Gridsim il est possible de simulée l'environnement de la grille de calcul, Nous avons abordé les étapes de la représentation de entités de la grille engendre par les principes algorithmes de simulation et d'ordonnancement Avant cela, bien sûr, la définition de chaque entités et les interactions entre eux en détails.

Chapitre III

Implémentation

Introduction

Nous avons présenté dans les chapitres précédents les notions de base concernant grille de calcul et le simulateur GRIDSIM. Nous avons exposé aussi notre simulation de grille en utilisant le simulateur GRIDSIM. L'objectif de notre projet était de simuler les diverses entités dans l'environnement de grille et l'interaction entre eux en utilisant un simulateur spécifique GRIDSIM tout en prenant en compte les caractéristiques fondamentales des grilles comme L'hétérogénéité et le passage à l'échelle. Nous allons présenter dans ce chapitre les démarches de la réalisation, la mise en œuvre de notre modèle de simulation et les résultats obtenus.

1Les outils du développement

Nous avons choisi pour développer notre approche un PC équipé par un processeur i3 CPU 2.20 Ghz, et 2 Go de RAM, avec un système d'exploitation Windows 8.1 Pro, et l'outil de développement NetBeans V8.0.1, et IDE Dev-C++. Nous avons utilisé la version 5.2 de GridSim.

1.1 NetBeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Développement and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme C, C++, JavaScript, XML, Groovy, PHP et HTML de façon native ainsi que bien d'autres (comme Python ou Ruby) par l'ajout de greffons. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, La refacturation, éditeur graphique d'interfaces et de pages Web). Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). Un environnement Java Développement Kit JDK est requis pour les développements en Java. NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)).

1.2 Java

Le langage Java est un langage de programmation informatique orienté objet créé par James gosling et Patrick naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au Sun World. La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou GNU/Linux, avec peu ou pas de modifications. Pour cela, divers plateformes et frame Works associés visent à guider, sinon garantir, cette portabilité des applications développées en Java.

1.3 Langage C

Le C est un langage de programmation impératif, généraliste, conçu pour la programmation système. Inventé au début des années 1970 pour réécrire UNIX, C'est devenu un des langages les plus utilisés. De nombreux langages plus modernes comme C++, Java et PHP reprennent des aspects de C.

1.4 Le simulateur GRIDSIM

GRIDSIM est un des simulateurs les plus utilisés dans les expérimentations de test des algorithmes sur les systèmes distribués et les grilles de calculs, nous l'avons choisi pour notre projet à cause de plusieurs facteurs. Parmi ces facteurs on peut citer :

- C'est un simulateur à événements discrets, ce qui permet l'envoi et la réception des messages entre les entités constituants la grille ou le système étudiés.
- Il utilise la notion d'entité. Les entités peuvent être des processeurs, des utilisateurs, des ressources, des informations …etc.
- Il offre la possibilité de créer des réseaux en utilisant les différentes entités et de tester de façon efficace et flexible leurs comportements.
- Il simplifie l'implémentation des algorithmes proposés sur les modèles de grilles.
- Il est basé sur un langage de programmation à usage général celui de JAVA, ce qui assure la portabilité des applications.
- Il fournit des modèles des composants réels des réseaux tels que les routeurs et les nœuds.
- Il permet la modélisation et la simulation des ressources et des applications d'ordonnancement dans des environnements distribués et parallèles

2 Étude de cas

Cette section décrit comment une simulation d'environnement de grille de calcul est créée à l'aide GridSim. Ceci peut être fait facilement en utilisant la boîte de dialogue de l'assistant, comme illustré à la Figure 1.

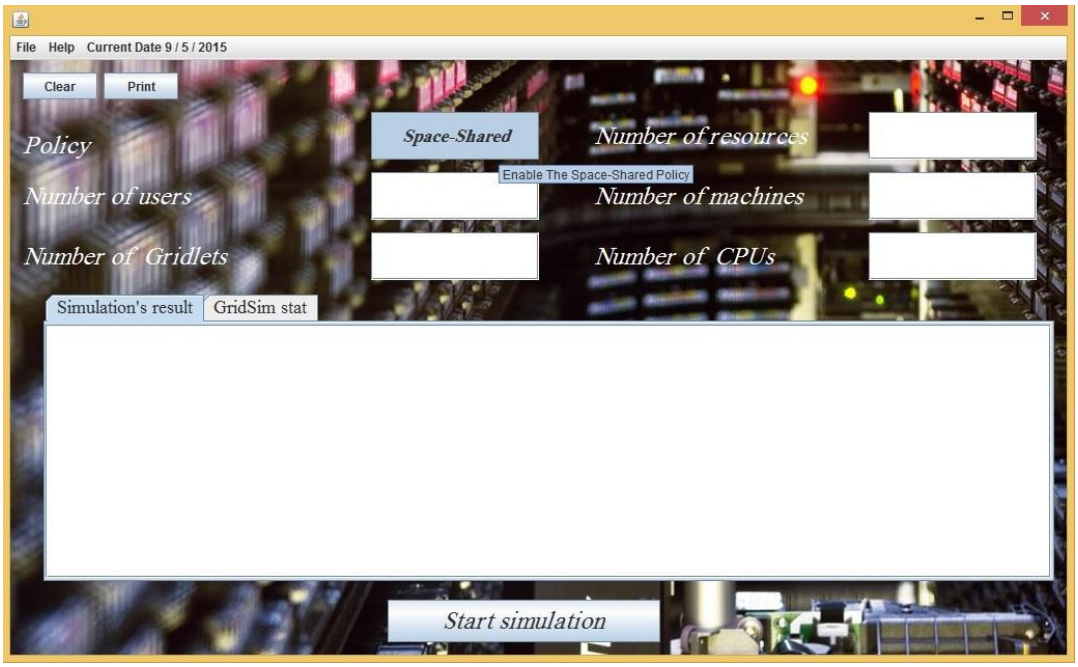

Figure 1 : Boîte de dialogue de l'assistant pour créer les utilisateurs du réseau et des ressources

L'utilisateur doit indiquer uniquement le nombre requis d'utilisateurs et de ressources à créer il peut aussi préciser le nombre de Gridlets de chaque utilisateur et le nombre de machines pour chaque ressource et le nombre d'éléments de traitement pour chaque machine. Des propriétés aléatoires peut aussi être généré automatiquement pour ces utilisateurs et ressources.

L'utilisation de cette application est très simple, juste il faut donner les paramètres de l'architecture de la grille puis click sur le bouton « Start simulation » et le résultat sera proche en détails comme celui présenté à la Figure 2

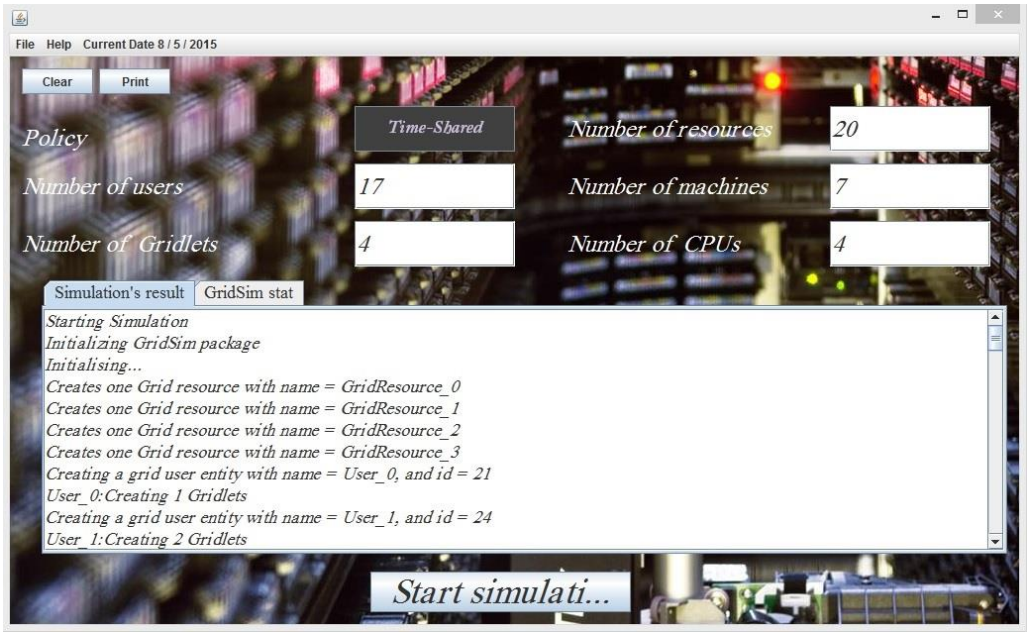

Figure 2: Presentation du résultats.

Dans cette figure nous choisissent la politique d'ordonnancement Space-Shared a l'aide du bouton coloré en noire, si nous cliquons ce bouton il sera changé la politique aux Time-Shared est-il sera changé pour nous offre le changement du politique si on a veut, comme celui représentera a la Figure 1. Nous proposons aussi que notre grille se compose de 17 utilisateurs et 20 ressources, et chaque utilisateur crée 4 Gridlets en maximum, nous proposons aussi que chaque ressource contienne 7 machines de la caractéristique aléatoire et chaque machine contient 4 éléments de traitement.

Les résultats présenté sur la fenêtre « Simulation's result » sont juste dépond de votre choix des paramètres en entrer, par exemple si on choisit de simulé 5 ressource vous peut voire la création de 5 ressource sur le résultat comme celui sélectionner dans la Figure 3.

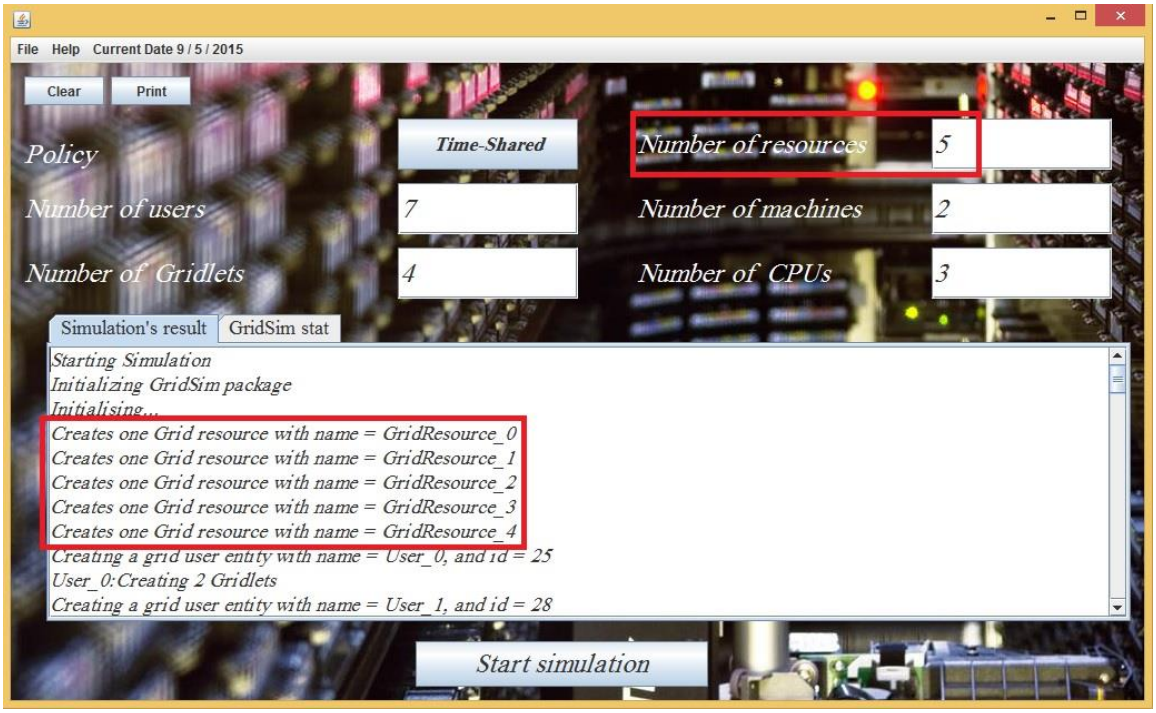

Figure 3 : Confirmation de la création des ressources.

En peut vérifier aussi si le nombre d'utilisateurs des résultats est réponde de notre choix par exemple à la Figure 4 nous avons choisi de simulée 2 utilisateur, le résultat obtenue satisfaisant notre choix comme celui sélectionner on rouge a la figure 4.

Figure 4 : Confirmation de création des utilisateurs.

Pour les Gridlets nous allons proposer un maximum de 4 Gridlets pour chaque utilisateur à travers le test représenter à la Figure 5, pour un résultat correct nous allons proposer que chaque utilisateur crée de 0 à 4 Gridlet.

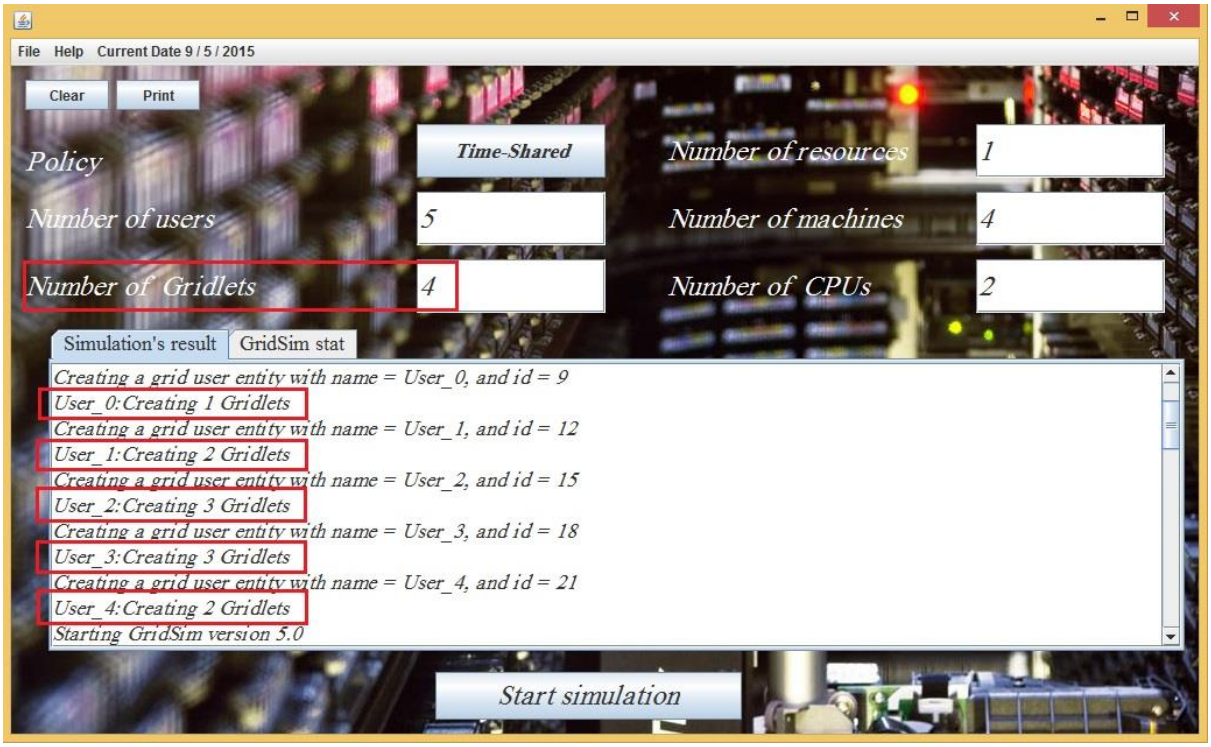

Figure 5 : Confirmation de création des Gridlets

La Figure 6 montre le résultat d'affectation des jobs aux ressources, dans cet exemple nous allons voire l'affectation de job Nº 1 à la ressource Nº 1 et la réception de cette Gridlets.

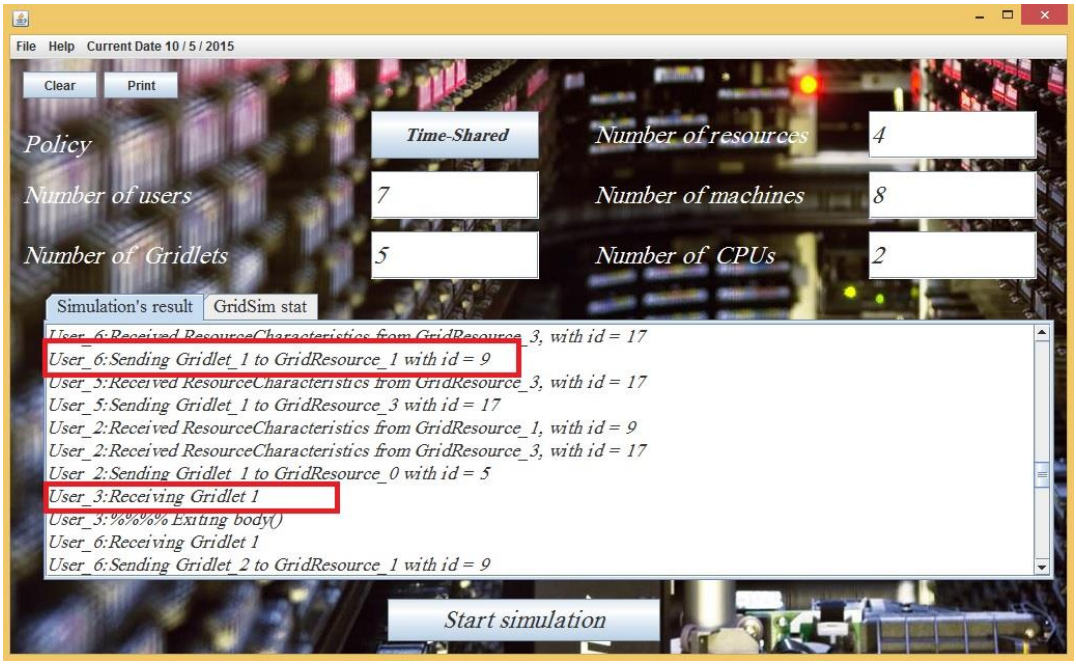

Figure 6 : L'affectation des jobs

La fenêtre Gridsim stat montre à vous le scenario entre la ressource et l'utilisateur durant la vie de simulation et le moment de temps de chaque opération (Des statistiques). (Figure 7)

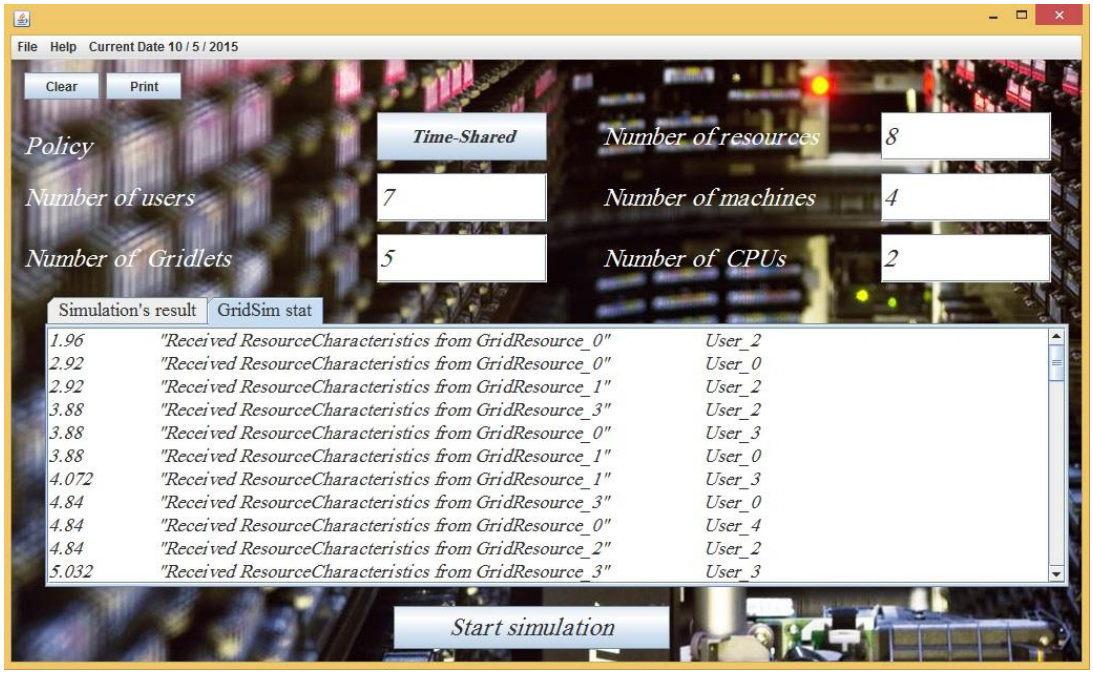

Figure 7 : Le statut de Gridsim

Le bouton Print vous permet d'imprimer le résultat de la simulation et l'état de GridSim durant la simulation. (Figure 8)

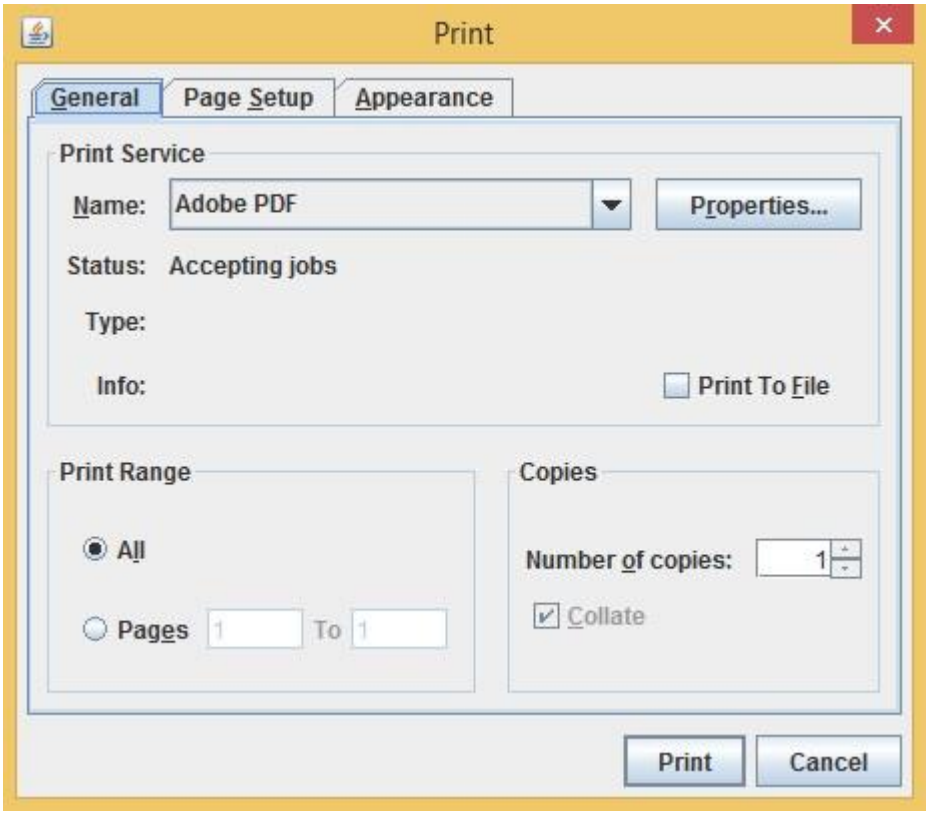

Figure 8 : Boite de dialogue d'impression.

On peut spécifie le nombre des copiées, l'imprimante et ses propriétés, …Etc. ou vous pouvez les sauvegardé dans un fichier.

Pour un utilisateur qui n'a aucune idée sur la grille de calcul et l'objectif de la simulation de la grille de calcul, nous avons ajouté ce document a l'application sous forme de PDF, pour accéder à ce document à partir de l'application il faut cliquer sur Help ensuite sur Document ou bien le raccourcie du clavier Ctrl + D. (Figure 9)

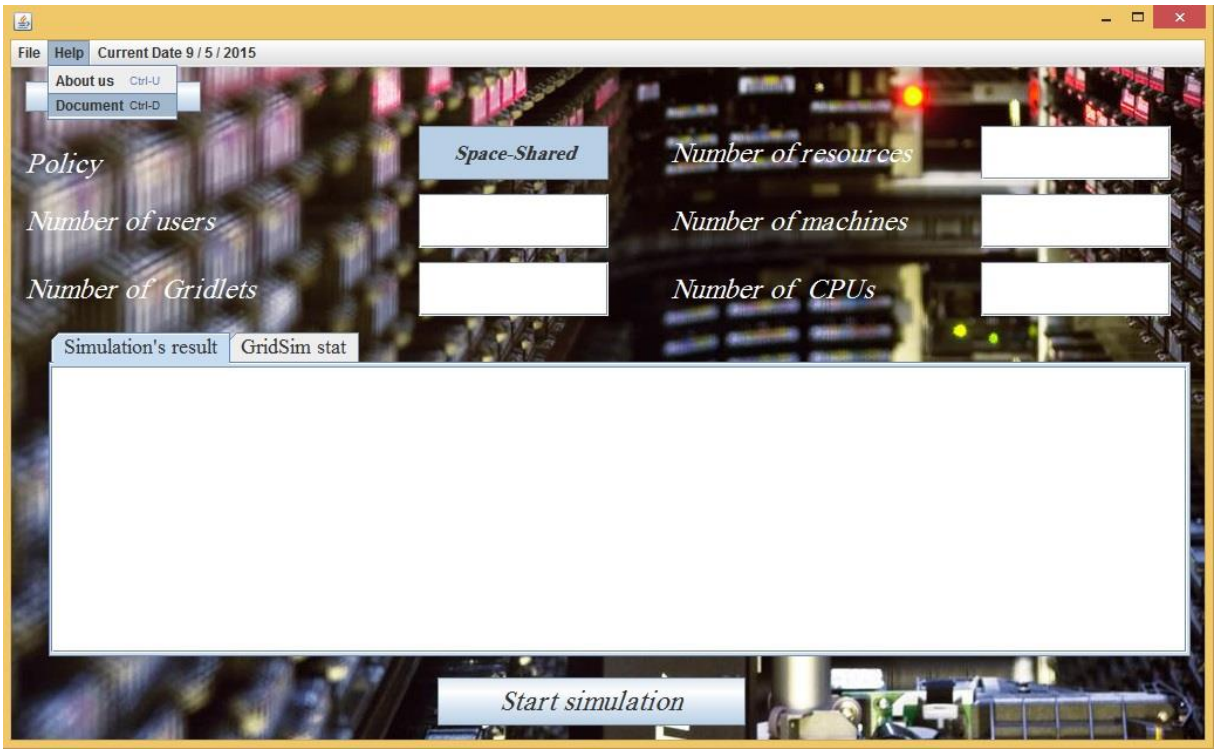

Figure 9: Accéder aux ce document sou format PDF

La Figure 10 montre une fenêtre représente notre université est les noms des producteurs de ce travail.

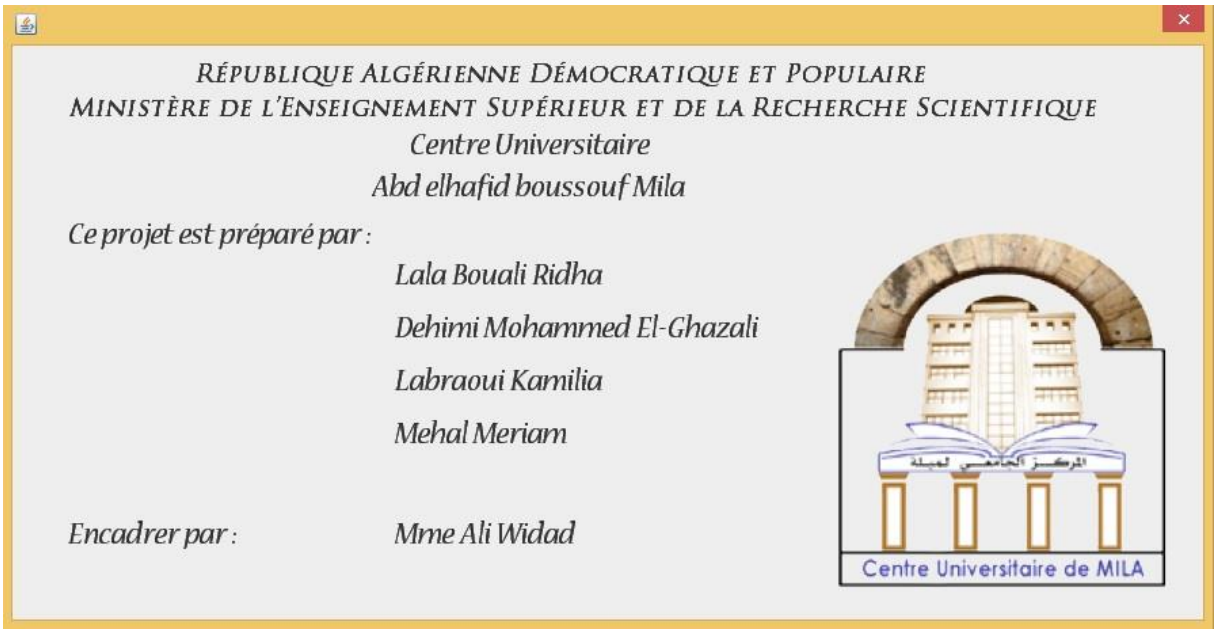

Figure 10 : Représentation des développeurs de cette application

Conclusion

Dans ce dernier chapitre nous avons présenté tout ce qui est lié au développement de notre application. Le chapitre est clôturé par une description détaillée des interfaces graphiques de notre application. Cette application simplifie a l'utilisateur pour fait une analyse de base sur l'environnement de la grille de calcul, on 'a parlé sur quelque outil quand a utilisé pour l'implémentation de simulation des grille de calcul.

Conclusion générale

Les scientifiques désirent une puissance de calcul toujours plus importante afin de pouvoir travailler sur des modèles d'une précision toujours plus grande.

Classiquement, des supercalculateurs sont développés pour réaliser ces calculs. Mais l'accès à ces supercalculateurs est très limité car leur achat et leur entretien sont coûteux.Pour pallier cette difficulté, les scientifiques cherchent à fédérer leurs ressources informatiques à l'aide des grilles de calcul dont le rôle primordial est de fournir un accès aux ressources non utilisées et ainsi, les scientifiques peuvent réaliser leurs calculs à moindre coût. Sur le principe que l'union fait la force, l'objectif est de fédérer le plus grand nombre de ressources possible pour disposer d'une importante puissance de calcul.

Dans notre étude, nous nous sommes intéressés au faire une simulation d'environnement de grille de calcul avec quelque ses services en utilisant le simulateur Gridsim

Les principales caractéristiques des grilles de calculs, sont l'hétérogénéité, le passage à l'échelle, la dynamicité. Ces caractéristiques font que la simulation est beaucoup plus complexe.

Partant de ce contexte, nous avons présenté notre modèle de simulation de grille de calcul avec quelque ses services. Adapté aux grilles de calcul qui tient compte surtout de la dynamicité des ressources et qui est totalement indépendant de toute architecture physique.

Pour implémenter notre modèle de simulation, nous avons utilisé un simulateur de grilles GridSim sous java. Les résultats que nous sommes obtenus, et que nous avons présentés et discutés dans cette mémoire, sont assez encourageants et mettent en évidence les gains que nous avons pu obtenir.

Dans l'avenir, nous voulons améliorer notre modèle par l'intégration des autres services. Nous voulons également définir d'autres paramètres de performance pour évaluer et comparer notre modèle avec d'autres déjà existantes.

Bibliographies

1. V. Berstis. Fundamentals of Grid Computing, IBM Red paper, October 2002.

2. Andrew S. Grimshaw, William A. Wulf, and the Legion team. The Legion vision of a worldwide virtual computer. Communications of the ACM, 1(40):39.45, January 1997.

3. Ian Foster, Carl Kesselman, and Steven Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputer Applications, 15(3), 2001.

4. Ian Foster. What is the Grid? A three-point checklist. GRIDtoday, 1(6), July 2002. http://www.gridtoday.com/02/0722/100136.html.

5. M. Chetty and R. Buyya. Weaving electrical and computational grids: How analogous are they?. Computing in Science and Engineering, May/June 2002.

6. I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. Intl J. Supercomputer Applications, 11(2):115 –128, 1997.

7. Karl Czajkowski, Carl Kesselman, Steven Fitzgerald, and Ian Foster. Grid information services for distributed resource sharing. hpdc, 00 :0181, 2001.,/ B. Plale, P. Dinda, M. Helm, G. von Laszewski, and J. McGee. Key concepts and services of a grid information service. 2002.

8. Rich Wolski, Neil T. Spring, and Jim Hayes. The network weather service : a distributed resource performance forecasting service for metacomputing. Future Generation Computer Systems.

9. Rajesh Raman, Miron Livny, and Marvin Solomon. Matchmaking : Distributed resource management for high throughput computing. In Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, July 1998.

10. Ian T. Foster, Carl Kesselman, Gene Tsudik, and Steven Tuecke. A security architecture for computational grids. In ACM Conference on Computer and Communications Security, pages 83-92, 1998.

11. http://toolkit.globus.org/toolkit/about.html

12. Foster. I, Kesselman. C and Tuecke. S. The anatomy of the grid: Enabling scalable virtual organization. International Journal of Supercomputing Applications, 2001.

13. Carl Kesselman and Ian Foster. The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers, November 1998.

14. Gestion des ressources dans les cloud computing à base des modèles économiques Présenté par: Bouamama samah.

15. Rajkumar Buyya and Manzur Murshed , GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing.

16. Rajkumar Buyya and Manzur Murshed , GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing.

17. http://www.buyya.com/gridsim/

18. Foster I., "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International Journal of High Performance Computing Applications, 2002, vol. 15, no. 3

19. S. Fitzgerald et al, "A Directory Service for Configuring High-Performance Distributed Computations", Proc. 6th IEEE Symposium on High-Performance Distributed Computing, pp. 365-375, 1997

20. I. Foster et al, "Chimera: A Virtual Data System for Representing, Querying and Automating Data Derivation", Proceedings of the 14th Conference on Scientific and Statistical Database Management, Edinburgh, Scotland, July 2002

21. http://www.cs.mu.oz.au/~raj/gridsim/

22. C. Simatos, "Making simjava count," MSc. Project report, The University of Edinburgh, September 12 2002

23. On Incorporating Differentiated Network Service into GridSim Anthony Sulistio, GokulPoduval, RajkumarBuyya, and Chen-KhongTham

24. GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing , RajkumarBuyya and ManzurMurshed