République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique
Ecole Doctorale d'Informatique EDI-Pole Constantine
Département d'informatique

UNIVERSITE EL-ARBI BENMEHIDI – Oum El-Bouaghi

MEMOIRE
pour obtenir le grade de

MAGISTER de l'Université El-Arbi Benmehidi
d'Oum El-Bouaghi

Discipline : Informatique
Spécialité : Intelligence Artificielle (I.A)

Présenté et soutenue publiquement par
Seïdali REHAB

Contribution au développement d'un environnement collaboratif basé GRID : proposition d'un prototype autour de Sakai

Directeur de mémoire :
Pr. Mokhtar SELLAMI

Devant le Jury composé de :

Dr.	Chaoui Allaoua	Rapporteur
Pr.	BENMOHAMED Mohamed	Président
Dr.	KHOLLADI Med Kheireddine	Examinateur

Mémoire préparé au sein du Laboratoire LRI – Annaba
Année 2010

Résumé

Dans ce mémoire nous présentons une proposition d'architecture de plateforme collaborative flexible exprimée en termes de couches en s'appuyant sur les concepts clés d'OGSA ainsi que ses processus de partage efficace des ressources. La communication des participants de la communauté virtuelle peut être dans les modes synchrones et asynchrones en utilisant divers outils dont le bureau partagé d'Agora. En effet, cet outil s'adapte bien avec la classe virtuelle et les contextes d'apprentissage à distance en offrant des caractéristiques intégrées telles que la vidéoconférence, le bureau partagé, le tableau blanc et la messagerie instantanée. Pour finaliser notre architecture, nous avons exploité la nouvelle version de Sakai, qui a été choisie pour ses riches fonctionnalités afin de servir de modèle pour l'injonction de services Web/Grid. Le co-déploiement d'un Service Grid partagé a monté la flexibilité et l'efficacité de cette approche en palliant aux problèmes souvent rencontrés pour une telle collaboration: problème de la distance, ressources hétérogènes, nombre important de participants, coût élevé de la collaboration, absence d'historique, collaboration limitée dans le temps, rentabilité et productivité critiquables. Sur un plan opérationnel, cette architecture est expérimentée autour d'un réseau LAN constitué de quatre machines, sous le système Linux Fedora core 4. Les autres outils sont bâtis sur Apache Ant, un exécuteur de taches qui permet le déploiement des programmes (déploiement de services), JDK pour la compilation du code de Globus et PostgresSQL : système de gestion de base de données relationnelle. Une démonstration a illustré la flexibilité et l'efficacité de notre approche.

Mots clés : Service Web, Service Grid, OGSA, WSRF, Plateforme collaborative, Intergiciel, Bureau partagé.

Abstract

In this thesis we present a proposal for a flexible collaborative platform architecture expressed in terms of layers based on key concepts of OGSA and its reliable sharing process of resources. The communication of participants of the virtual community can be in both synchronous and asynchronous modes using various tools including Agora shared desktop. Indeed, this tool fits well with the virtual classroom and distance learning contexts by offering integrated features such as videoconferencing, shared desktop, whiteboard and instant messaging. To complete our architecture, we have exploited the new version of Sakai, which was chosen for its rich features to serve as a model for the injunction of Web/Grid services. A co-deployment of a shared Grid Service mounted the flexibility and the effectiveness of this approach in overcoming the problems often encountered in such collaboration: problem of distance, heterogeneous resources, large number of participants, high cost of collaboration, lack of historical, limited collaboration over time, criticism productivity and profitability. On an operational level, this architecture is experienced over a LAN network consists of four machines, under Linux Fedora Core 4 system. The other tools are built on Apache Ant, a spots executioner which allows the deployment of programs (service deployment), JDK for compiling Globus code and PostgresSQL (management system of relational database). A demonstration has shown the flexibility and effectiveness of our approach.

Keywords: Web Service, Grid Service, OGSA, WSRF, Collaborative Platform, Middleware, Shared Desktop.

ملخص

في هذه الرسالة نقدم اقتراحا لهيكل أساس تعاوني، هذا الهيكل عبارة عن طبقات مستندة إلى المفاهيم الرئيسية لـ: [أوكسا] المرفوقة بعمليات تقاسم الموارد الفعالة. اتصال المشاركين للمنظمة الإفتراضية يمكن أن يكون في وسائط متزامنة وغير متزامنة باستخدام وسائل مختلفة بما في ذلك المكتب المشترك [أكورا]. في الواقع ، هذه الأداة تتوافق جيدا مع الفصول الدراسية الافتراضية ومع حالات التعلم عن بعد من خلال تقديمها ميزات وخصائص متكاملة مثل محاضرات الفيديو المغلقة، المكتب المشترك، السبورة البيضاء المشتركة والمراسلة الفورية. لإتمام هيكلنا، قمنا باستغلال الإصدار الجديد لـ: [ساكاي]، الذي تم اختياره على أساس مميزاته الغنية ليكون بمثابة نموذج لخدمات الويب/الشبكة.النشر المشترك لخدمة الشبكة المشتركة زاد من مرونة وفعالية هذا الإقتراح الذي تغلب على مشاكل عادة ما تواجه في التعاون : مشكلة بعد المسافة، الموارد غير المتجانسة، عدد كبير من المشاركين، ارتفاع تكلفة التعاون، عدم وجود تسجيل التاريخ، محدودية التعاون مع مرور الوقت، الإنتاجية والربحية المنتقدة. على المستوى العملي أو التطبيقي، هذا الهيكل أنجز عبر شبكة اتصال محلية [لان] التي تتكون من أربع آلات حاسوب، تحت نظام التشغيل [لينكس فيدورا كور 4]. الأدوات الأخرى بنيت على : - [أباتشي آنت] : منفذ المهام الذي يتيح نشر البرامج (نشر الخدمات) - [ج د ك] : لتصنيف رموز [كلوبوس] – [بوستكريس س ك ل] : نظام إدارة قواعد البيانات العلائقية. الإثبات والبرهنة أظهرت مرونة وفعالية اقتراحنا.

كلمات مفتاحية : خدمة ويب، خدمة شبكة، [أوكسا]، WSRF، أساس تعاوني، برنامج وسيطي، مكتب مشترك.

Remerciement

Je remercie en premier et je ne cesserai jamais de le faire, le bon Dieu, que sans lui je n'aurai pas pu réaliser ce travail et que je l'aurai effectué avec peu de cœur, mon dieu qui m'a donné la foi, la santé et beaucoup de patience « وما بكم من نعمة فمن الله ».

En deuxième lieu, je tiens à exprimer toute ma gratitude envers mon Rapporteur de thèse Docteur CHAOUI ALLAOUA pour toute la confiance qu'il m'a accordée dans le cadre de ce projet.

Un remerciement particulier s'adresse au DOCTEUR BOUENADA MOURAD, enseignant à l'université MENTOURI de Constantine, grâce à qui j'ai pu avancer dans cette thèse et qui a apporté, sans relâche, son expertise et son soutien moral, ainsi qui m'a fait l'honneur de lire attentivement mon manuscrit.

Je remercie également trois de mes collègues: M. OTHMANI MARABOUT ZOHIR, M. LABOUDI ZAKARIA et M. BOUANAKA MOHAMED ALI, pour leur contribution dans ce projet par les conseils qui m'ont donné durant la rédaction de la thèse. Je remercie aussi les membres du jury de mémoire pour leur disponibilité, et, surtout, pour leur accompagnement ; la précision et la pertinence des remarques ont été déterminantes sur le fond et sur la forme du travail présenté : PROFESSEUR BENMOHAMED MOHAMED, DOCTEUR KHOLADI KHEIREDDINE.

Ce travail est aussi la contribution d'un ensemble d'acteurs qui se sont cordialement prêtés à la traduction de quelques articles pour la rédaction de cette thèse. Plus particulièrement à M. MOHCINE, Melle MANEL et Melle SAMEH étudiants à l'Université MENTOURI de Constantine, ainsi que mon frère RAFIK et ma sœur KENZA.

Enfin, l'aboutissement de cette thèse a été rendu possible grâce aux encouragements de toute ma famille, parents, frères, sœurs et amis durant ces trois années. Pour finir, un merci, tout particulier, à mes beaux parents pour leur implication de tous les instants.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

1. Contexte du travail

Le développement accéléré des technologies de l'information et de la communication d'une part, la complexité et l'interaction de plus en plus grande entre les disciplines scientifiques d'autre part, imposent la recherche d'outils et de moyens technologiques pour favoriser et faciliter la collaboration entre les membres d'une même communauté scientifique, généralement distribuée à travers l'espace, nous nous situons dans le contexte du travail collaboratif. Ce terme désignant à la fois de nouvelles pratiques et de nouvelles compétences, de nouvelles formes d'organisation et de communication ainsi que de nouveaux environnements numériques de travail sur les réseaux (Internet, Intranet, Extranet). Il est à noter que la dimension temps réel est prioritaire pour mobiliser en même temps les compétences multiples présentes au sein de différentes équipes. Elle représente un grand saut qualitatif, surtout dans les organisations où l'éclatement général des structures rend le fonctionnement en mode projet difficile à mettre en œuvre et à maintenir dans la durée. Le seul vrai vecteur de collaboration et de gain de productivité est donc le véritable partage. Il est, par conséquent, impératif d'amener les collaborateurs à franchir le cap entre une utilisation partielle et une utilisation complète des fonctionnalités proposées par les plateformes. Les outils de travail collaboratif envisagés couvrent globalement quatre grandes catégories de fonctions :

- La communication - constituant la base du concept de travail collaboratif - est composée des outils les plus utilisés tel que l'email.
- La deuxième catégorie est le partage d'applications et de ressources ; elle permet aux membres d'une équipe de projet de travailler, collaborer, sur un même document ou une même application, et ainsi d'élaborer un projet commun.
- Vient ensuite l'accès et le partage d'informations et de contenus, dont l'objectif est de permettre et faciliter l'accès aux informations ainsi que de cartographier les compétences présentes dans l'organisation.
- Enfin la coordination et la synchronisation, aident les membres d'une équipe projet à tenir leurs objectifs tout en satisfaisant aux contraintes de qualité, de coût et de délai.

De nombreuses problématiques se posent alors quant aux méthodes, aux formes d'organisation, et aux outils à exploiter en fonction des objectifs et du contexte dans lequel se déroule cette collaboration. Les conditions de travail des membres des diverses communautés doivent être suffisamment homogènes pour favoriser les échanges et exploiter de façon optimale les diverses ressources disponibles à travers les réseaux informatiques.

C'est dans ce contexte que l'architecture de service Grid OGSA est née en se basant sur une approche orientée service fondée sur des Standards ouverts de services à état (stateful service). Les services développés dans ce contexte ont pour but principal de proposer de nouveaux paradigmes de travail en équipe et de développer de nouveaux environnements collaboratifs qui s'appuient sur des réseaux informatiques à grande échelle ouverts au développement d'outils collaboratifs.

Partant de cette constatation, et vu que les architectures à base de Grid disposent d'énormes potentialités pour supporter la collaboration sur le Web selon une approche orienté service, il nous a semblé important de nous appuyer sur l'architecture des services Grid OGSA pour développer de nouveaux environnements collaboratifs synchrones et asynchrones qui permettraient à plusieurs utilisateurs distants au sein d'une organisation virtuelle de collaborer en ligne, et de partager des services dont ils auront besoin afin d'atteindre les objectifs assignés.

2. Problématique et objectif

Après le web qui a montré la puissance d'une infrastructure pour le partage d'informations distribuées, nous assistons à l'émergence d'une infrastructure d'un nouveau genre: la grille informatique ou Grid, une infrastructure pour le partage de ressources distribuées. Bien que l'usage de Grid demeure encore, et ce depuis plusieurs années, l'apanage du calcul intensif, Grid ouvre de nouvelles perspectives qui n'étaient pas envisageables auparavant. Ceci nous amène à penser que ce paradigme est en passe de révolutionner la vision réductrice que l'on se fait encore aujourd'hui de l'informatique distribuée. Pour illustrer notre propos, ce mémoire traite de la problématique de collaboration par ordinateur basée Grid. Construite sur une architecture Grid, la plateforme collaborative Sakai permet d'offrir une puissance calibrée selon les besoins du groupe de collaboration. Le tout est conçu de manière à rendre transparente à l'utilisateur toute la complexité de la technologie sous-jacente.

3. Cadre et structure de cette thèse

Cette thèse comporte quatre chapitres, décrits ci-dessous :

Chapitre 1 : Le travail collaboratif

Ce chapitre dresse un état de l'art pour caractériser le concept espaces collaboratifs, les principaux termes et mots clés qui font partie du travail collaboratif et des classifications des outils collecticiels afin d'analyser leur évolution au cours de ces dix dernières années, sur les plans fonctionnels, architecturaux et technologiques.

Chapitre 2 : Des services Web aux services Grid

Ce chapitre dresse un état de l'art et une prospective technologique autour des architectures susceptibles d'être utilisées pour déployer notre plateforme collaborative. Ce chapitre fournit donc toutes les technologies qui incluses dans ces architectures ainsi que l'apport des services Grid par rapport aux services Web.

Chapitre 3 : Le choix des plateformes

Ce chapitre décrit une partie de notre contribution, qui est l'adoption des deux plateformes Sakai et Globus Toolkit. La première est choisie parmi quatre plateformes collaboratives très célèbres au monde selon leurs fonctionnalités, leurs technologies en matière d'applications d'entreprise ainsi que leurs évolutions à venir. La deuxième consiste à étudier son architecture en faisant apparaitre celle qui constitue la base de nombreux travaux.

Chapitre 4 : La nouvelle plateforme collaborative proposée

Ce chapitre décrit notre méthodologie et nos résultats d'expérimentation. Il vise tout d'abord de proposer notre architecture de la nouvelle plateforme collaborative exprimée en termes de couches. Ainsi, des tests sur notre grille de calcul « Globus » sont fait afin de montrer le bon fonctionnement de celle-ci avec un déploiement d'un service Grid, en utilisant un outil de collaboration très simple et très fiable en même temps qui est Agora. Ces résultats sont essentiellement qualitatifs afin de comprendre le point de vue utilisateur.

On terminera par une conclusion générale qui va contenir le bilan de notre travail ainsi que l'apport de Grid par rapport au Web.

Chapitre I

Le travail collaboratif

Introduction

Dans ce chapitre nous présentons un état de l'art des applications destinées à offrir un support au travail collaboratif en s'articulant autour de trois sections :

Dans un premier temps, nous introduisons les principaux termes et mots clés qui font partie du travail collaboratif (collaboration ; sessions ; interactions synchrones/asynchrones ; collecticiel…). Nous abordons ensuite deux types de classification incontournables dans la littérature du domaine : une classification espace-temps et une classification par domaines d'application. Finalement, nous présentons des outils collecticiels avec quelques exemples classifiés selon leur mode d'interaction asynchrone ou synchrone. Cette présentation a pour but d'analyser l'évolution de ces outils au cours de ces dix dernières années, sur les plans fonctionnels, architecturaux et technologiques.

1. Concepts fondamentaux

1.1 Le travail collaboratif

Le travail collaboratif [43], appelé à l'origine Travail Coopératif Assisté par Ordinateur, ou TCAO (de l'anglais *Compurter Suported Cooperative Work*, ou CSCW), correspond à l'utilisation d'ordinateurs en réseau pour le travail de groupes d'utilisateurs. Pour clarifier nos propos, deux citations très générales, données par Ellis et Kraemer en 91 et 88, définissent bien le domaine des systèmes collaboratifs [21], [40] :

> Système à base d'ordinateurs qui supporte des groupes de personnes réalisant en commun une tâche ou un but et qui fournit une interface pour accéder à un environnement commun.
>
> Système informatique qui facilite la résolution de problèmes par un ensemble de décideurs travaillant en groupe.

Apparu dans les années 90, le travail collaboratif est issu de quatre grands domaines :

- Sciences sociales (sociologie, théorie des organisations), prenant en compte l'organisation des personnes, leurs apports, l'efficacité d'un groupe
- Intelligence artificielle distribuée, sciences cognitives pour l'interprétation de la sémantique des informations, la planification, l'aide à la réalisation de tâches en commun
- Interfaces homme machine, pour la conception d'interfaces multiutilisateurs
- Informatique répartie, systèmes distribués et réseaux pour les stockages, transferts et échanges d'information

1.2 La session

Le travail collaboratif et les activités de groupe sont organisés en sessions. Une session, unité essentielle du travail en groupe, se compose d'un ensemble d'utilisateurs qui partagent des intérêts communs [15]. Ces utilisateurs se connectent de divers emplacements et manipulent des données, partagées à l'aide des outils support de la collaboration [17].

Une session peut être synchrone ou asynchrone.

Dans une Session Synchrone, les membres du groupe sont présents en même temps pour réaliser le travail collaboratif. Ils dialoguent de façon interactive et interagissent en temps-réel sur les outils et les données manipulées dans la session. Un exemple de ce type de session, est une session de téléconférence.

Dans une Session Asynchrone, la co-présence des différents membres du groupe n'est pas nécessaire. De ce fait, ne pouvant réaliser de dialogue interactif, les membres d'une session asynchrone ne communiquent que via des médiums asynchrones tel que l'email.

La séparation entre Session Asynchrone et Session Synchrone, introduite à l'origine à cause des caractéristiques techniques des réseaux sous-jacents, est bien entendu fortement liée au type de travail de groupe réalisé. Certains travaux collaboratifs peuvent cependant se dérouler dans des modes asynchrones, ou synchrones. Par exemple, la production d'un document au travers d'un éditeur collaboratif peut se faire suivant les deux modes. L'élaboration des parties du document peut se faire de façon asynchrone. Cependant, des phases de coordination

synchrones peuvent être nécessaires pour faciliter la production d'un document unique et cohérent.

Par la suite, nous avons gardé cette séparation asynchrone/synchrone comme base de notre classification, pour plus de clarté.

1.3 Collecticiel

Les systèmes informatiques, élaborés dans le cadre de la recherche en TCAO, permettant à des groupes d'utilisateurs de collaborer à des buts communs sont qualifiés en anglais de *groupware*. La traduction française couramment utilisée pour ce terme est "collecticiel" [42]. En fait, la pluridisciplinarité du TCAO rend difficile la définition de ces systèmes. Une définition assez courante que nous devons à Ellis [21] a été traduite par Karsenty [38] de la façon suivante :

"Système informatique qui assiste un groupe de personnes engagées dans une tâche commune (ou but commun) et qui fournit une interface à un environnement partagé".

Collecticiel est également la traduction officielle adoptée par l'Association Française pour la Cybernétique Économique et Technique (ASTI) [68]. Mais très souvent, nous rencontrons également les termes "système collaboratif", "système de TCAO" ou encore "application collaborative" pour désigner le *groupware*. Pour notre part, nous n'emploierons par la suite que les termes "collecticiel" ou "application collaborative" pour désigner ces systèmes informatiques.

Le collecticiel (ou groupware) [38] désigne donc tous les produits logiciels, outils, services ou plates-formes, conçus pour des groupes d'utilisateurs.

1.4 La coopération et la collaboration

Les termes "coopération" et "collaboration", et les concepts qu'ils représentent, sont important dans notre présentation. Néanmoins, leur distinction devient rapidement difficile. Dans la littérature ces deux termes sont très souvent utilisés comme synonymes. Les dictionnaires ne nous aident pas vraiment dans cette tâche : une des définitions pour "collaborer" étant "coopérer".

Dans les travaux de Ellis [22], nous trouvons cependant une distinction entre ces deux termes. Son "modèle des 3C" cherche à caractériser la coopération en trois niveaux distincts, selon l'intensité des relations établies entre les individus et les tâches réalisées : la communication, la coordination, et la collaboration. La communication représente le niveau de coopération le plus lâche, la coordination un niveau intermédiaire, et la collaboration représente le niveau de coopération où les tâches sont les plus imbriquées et fortement liées.

Dans "le modèle du trèfle fonctionnel" [30] [44] nous trouvons une classification plutôt fonctionnelle qui n'insiste pas sur une distinction entre coopération et collaboration. La notion de coopération est caractérisée par trois axes ou fonctions principales : la communication, la coordination, et la production. La communication correspond à l'échange direct d'informations entre les membres d'un groupe. La coordination consiste à définir les règles d'interaction pour

contrôler la contribution des membres du groupe qui participent à un travail commun. La production est caractérisée par le partage d'un espace où les membres peuvent stocker, traiter et partager un ou plusieurs objets communs. Dans [30], le lecteur trouvera plus de détails concernant les fonctionnalités relatives à chacun de ces trois axes.

Nous prenons en conséquence le parti de considérer les termes de coopération et de collaboration comme étant synonymes, comme l'usage commun et scientifique semblent l'avoir montré.

1.5 La télé-présence

La télé-présence, ou notion de présence, représente la conscience (de l'anglais *awareness*) commune dans un travail de groupe qui permet de définir le contexte dans lequel le travail se réalise. Cette conscience de groupe correspond à la compréhension que chacun a sur : (i) avec qui il travaille, (ii) l'activité de chacun et (iii) la manière dont les actions de chacun interagissent [16].

Dans un collecticiel, à la différence d'une application mono-utilisateur, plusieurs personnes peuvent agir en même temps sur les mêmes artefacts de travail. Dans un tel environnement partagé, les acteurs n'ont pas spontanément connaissance des actions des autres. Ainsi la conscience des activités individuelles et des activités du groupe s'avère une information critique pour rendre possible le succès de la collaboration. Elle peut présenter des avantages tels que : la réduction de l'effort de coordination des actions ; l'anticipation des actions des autres; ou même l'aide à des personnes pour s'intégrer aux activités [32].

Selon l'analyse détaillée de Dix [13], la notion de télé-présence se décline sous trois formes :

- conscience de la présence des membres du groupe et de leur disponibilité dans le travail coopératif ;
- conscience des actions réalisées par les membres du groupe ;
- conscience des effets consécutifs à ces actions.

Ainsi, la télé-présence, sous ses différentes formes, permet de contrôler les comportements de chaque participant. En effet, pour pouvoir travailler ensemble il est indispensable d'une part d'être conscient de la présence des autres participants et d'autre part d'être au courant de leur travail.

1.6 Collaboration et hypermédias

Selon Everardo Reyes García [54] La collaboration à base d'hypermédias a été un sujet étudié par le domaine nommé « hypermédias collaboratifs » [56]. Sous cette optique, un hypermédia collaboratif peut être défini à travers ses objectifs qui consistent à faciliter, garantir et proposer aux utilisateurs un environnement pour le travail à distance en groupe à travers la technologie des hypertextes. Il s'agit d'un système qui intègre dans son architecture des outils de collaboration à distance.

Parmi les applications les plus répandues pour proposer ces services, nous pouvons citer : les emails, les news, les calendriers groupaux, les systèmes d'écriture collaborative, les systèmes de dessin collaboratif, les *whiteboards*, les visioconférences, les audioconférences, les chats, les systèmes d'aide à la décision, les jeux en réseau.

1.6.1 Hypermédia collaboratif

Un système hypermédia collaboratif doit être configuré dans un entourage hypertextuel et conçu à partir d'une typologie de travail en collaboration. Il peut avoir également d'autres caractéristiques complémentaires telle qu'une couche d'adaptativité pour se rendre plus robuste. Les projets *ELMART* en Allemagne et *TANGOW* en Espagne [6] sont de bons exemples.

D'après Haake et Wang [33], il est possible de distinguer au moins cinq besoins dans un système hypermédia collaboratif :

- *gestion d'objets et de données partagés* : support de distribution des données entre divers utilisateurs ;
- gestion d'interfaces partagées entre utilisateurs : un système d'écran partagé du type WYSIWIS (*What You See Is What I See*) est parfois requis ;
- *support d'alerte au groupe* : un agent de détection de connexion des utilisateurs doit être présent ;
- *support à la coordination* : il facilite la coordination entre les membres du groupe ;
- *support à la communication* : il offre aux collaborateurs des moyens de communication soit dans l'environnement partagé, soit aux dehors.

1.6.2 Annotations, versions et droits d'auteur

Outre ces besoins, nous avons constaté d'autres exigences de contenus sur le Web, à savoir : les annotations, la gestion de versions d'un document et les droits d'auteur de matériels produits.

Dans ce sens, les annotations sont souvent considérées comme des contributions faites par des auteurs externes à un document publié sur le Web ; elles constituent un sujet largement étudié afin d'offrir une valeur ajoutée aux documents sur le Web.

Pour sa part, la gestion de versions de documents fournit un moyen pour traquer un historique des révisions ; il permet une exploration chronologique de productions, de collaborations asynchrones, du *workflow*, et d'autres stratégies pour optimiser le travail en équipe.

Quant aux droits d'auteurs, le problème semble s'acheminer vers la restriction et le contrôle d'accès aux documents. Généralement, on associe aux droits d'auteurs l'usage de *Digital Rights Management Systems* qui peuvent se diviser en deux générations : une première basée sur le cryptage, et une deuxième basée sur la définition de langages pour spécifier le détenteur de droits et les règles de protection. Ainsi, on utilise les *Document Object Identifiers* (DOI) pour identifier la propriété intellectuelle ou le langage ODRL (*Open Digital Rights Language*) du W3C pour insérer les informations de droits dans le contenu.

1.7 Web 2.0

Un dernier point de vue sur le binôme usage constructif/collaboration se trouve au sein du Web 2.0, qui est l'un des derniers termes en vogue définissant un type de participation sur le Web dans le but de former des collectivités virtuelles [49].

En effet, bien que le Web 2.0 soit un concept utilisé dans des cadres aussi divers que le *marketing* ou l'historicité du Web, il incarne l'appropriation des outils et des contenus par les usagers à travers de stratégies auparavant méconnues comme les « wikis », la « folksonomie » ou la « syndication ». Pour nous, le Web 2.0 constitue effectivement un type de création collaborative dans le sens de « manière de faire », comme l'a indiqué Michel de Certeau : « Ces manières de faire constituent les milles pratiques par lesquels des utilisateurs se réapproprient l'espace organisé par les techniques de la production socioculturelle » [10]. En ce sens, le travail collaboratif peut s'observer à deux niveaux. Premièrement, dans la manière dont les usagers partagent entre eux leurs découvertes et informations sur le Web. Deuxièmement, dans la manière dont les développeurs ou les usagers familiarisés avec les langages de développement réutilisent leurs applications et les fusionnent, créant de nouvelles versions, aussi nommées « *mash-ups* ».

À présent, nous analysons brièvement deux de ces manières de faire.

1.7.1 Folksonomie

Le vocable « folksonomie » provient du néologisme anglais « *folksonomy* » qui réfère à une pratique d'organisation collaborative par le biais de mots clés, c'est-à-dire qu'une collectivité coopère de façon simultanée afin d'organiser l'information en catégories.

La technique utilisée pour cette forme d'organisation est l'« étiquetage », ou « *tagging* » en anglais, dont la métaphore, souvent employée, consiste à imaginer les mots clés comme espaces dans un nuage où les catégories les plus populaires apparaissent en taille de police plus grande que les autres. Des sites comme flickr.com ou blogmarks.net sont des exemples utilisant des nuages de mot clés.

Ainsi, l'information organisée, catégorisée et partagée peut être un objet média quelconque sur le Web ou encore un signet, d'où la notion de « signets collectifs », qui offrent la possibilité aux auteurs d'enregistrer leurs favoris directement sur un autre site Web et de les rendre visibles à d'autres utilisateurs. Nous observons au sein de sites comme del.icio.us, que le contenu du site n'est rien d'autre que des listes de liens organisées par thème ou par l'auteur de l'enregistrement. De plus, ce site propose un service de notification aux auteurs lorsqu'un autre visiteur « marque » un même site afin de favoriser la mise en relation des utilisateurs. Cet échange de contenus se révèle de plus en plus automatique, comme le dévoile la « syndication ».

1.7.2 Syndication

Le terme « syndication » est associé à la pratique d'abonnement tel qu'elle est pratiquée par les autres médias. Sur le Web, le modèle confère la capacité à un usager de s'inscrire à un site afin de recevoir ses dernières mises à jour de manière automatique et sans avoir besoin d'y retourner périodiquement.

La syndication est basée fondamentalement sur des langages de description XML dédiés à des nouvelles et des contenus, comme RSS (*Really Simple Syndication*) ou Atom. Ces flux arrivent généralement sous la forme d'un titre et d'un extrait d'un article, plus un lien vers l'intégralité. Ils s'appliquent non seulement aux textes, mais aussi à tout autre objet média. La syndication consiste donc à identifier le fichier XML correspondant au flux de nouvelles et de l'enregistrer sur un « agrégateur ». Les navigateurs les plus récents peuvent également fonctionner en tant qu'agrégateurs.

2. Classification des applications collaboratives

Il est difficile de définir une classification (ou taxonomie) unifiée des applications de TCAO. En plus de la grande diversité du domaine, chaque classification peut privilégier un aspect particulier. Nous avons ainsi choisi de présenter deux classifications considérées comme incontournables dans la littérature du domaine : l'une concerne des caractéristiques temporelles et spatiales, et l'autre la catégorie fonctionnelle de l'application.

2.1 La classification espace-temps

L'espace et le temps constituent les dimensions les plus courantes dans les classifications d'applications collaboratives [38].

La première dimension, l'espace, concerne la distance géographique séparant les utilisateurs de l'application. Par exemple, les membres d'une réunion peuvent se trouver dans une même pièce ou être carrément situés dans des lieux distants (des bâtiments, des villes, ou bien même des pays différents).

La dimension temporelle caractérise plutôt le type d'interaction entre utilisateurs. Les membres du groupe peuvent interagir en même temps et en direct (les actions d'un participant sont immédiatement transmises aux autres), ce qui définit une collaboration synchrone. Dans ce cas, des problèmes de synchronisation et de gestion de la concurrence se posent. Il faut en effet résoudre les problèmes de conflit induits par exemple lors de modifications simultanées et contradictoires réalisées sur une même donnée. Si une gestion de la concurrence n'est pas mise en place, il est fort probable que des conflits et des incohérences se produiront dans le déroulement de l'application. Les problèmes de contrôle de concurrence et de synchronisation ont été beaucoup étudiés : dans le domaine des systèmes répartis conventionnels et dans le domaine du TCAO (où des problèmes d'ordre humain s'ajoutent aux problèmes techniques) [48] [52] [63].

Outre la collaboration synchrone, les membres d'un groupe peuvent également agir à des instants différents, c'est-à-dire, les actions sont espacées dans le temps. Il s'agit alors d'une collaboration asynchrone. Dans ce cas, il est important que l'état de l'activité soit conservé en permanence afin que les membres du groupe soient capables d'observer l'historique des actions qui ont été effectuées avant leur arrivée.

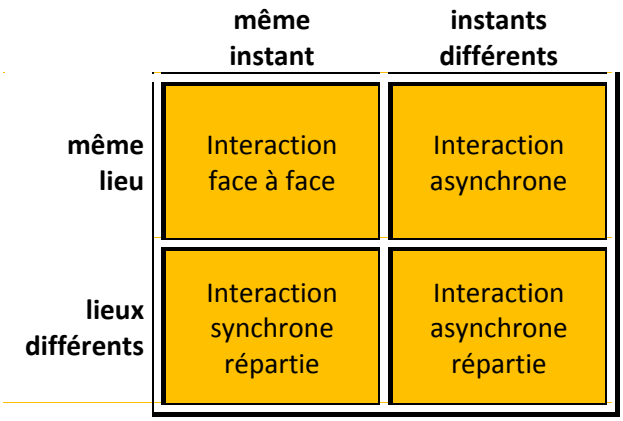


Figure 1.1. La matrice espace-temps

La figure 1.1 illustre la "matrice espace-temps", ou encore matrice de Johansen [34]. De nombreuses critiques ont été suscitées vis-à-vis de cette classification traditionnelle (de plus en plus les applications tendent à couvrir plusieurs quadrants) et plusieurs propositions ont été développées pour affiner cette classification [31].

2.2 La classification par domaines d'application

Beaucoup d'auteurs choisissent d'élaborer une classification par domaines d'application, où une liste de catégories fonctionnelles est définie pour regrouper les applications de TCAO [7] [41] [61]. Par la suite, nous présentons, à titre d'exemple, une liste non exhaustive de catégories d'applications collaboratives les plus souvent référencées:

• Courrier électronique (courriel) – Cette catégorie de collecticiels représente de loin le moyen de communication asynchrone le plus répandu et le plus utilisé de nos jours. Le courriel désigne le service de transfert de messages envoyés via Internet vers la boîte aux lettres électronique des destinataires choisis par l'émetteur. Une adaptation de ce service de messagerie a connu un grand succès dans le monde de la téléphonie mobile, appelé SMS (de l'anglais *Short Messaging Services*).

• Messagerie instantanée et le forum de discussion – Comme le courrier électronique, ces deux types de systèmes sont dédiés à la communication par échange de messages. Dans le premier cas (également appelé Chat), contrairement au courrier électronique, la communication est conçue pour être instantanée, *i.e.* synchrone. Le deuxième type de système représente des lieux (normalement des sites Web) où les individus peuvent partager leurs connaissances/expériences par des échanges de messages de manière asynchrone. La plupart des forums sont organisés en fils de discussion où un message ou un thème initial détermine un premier fil de discussion. L'ensemble des discussions est généralement visible par les participants, et éventuellement par tous les internautes. Les deux classes principales de forum de discussion sont les *newsgroups* et les *bulletin boards* (ou tableaux d'affichage).

• Gestion de flux de processus – Connue en anglais sous le nom de *workflow management*, cette catégorie représente des systèmes dédiés à la gestion de processus (industriels, commerciaux, administratifs, *etc.*) et à la coordination des différents intervenants au cours d'un processus [2] [20]. Également dénommé gestionnaire de procédés, il est souvent défini comme un outil qui prend en charge les documents en cours d'élaboration liés à des procédures et au routage des données. Ces systèmes sont très souvent utilisés par des entreprises dans le but de coordonner les taches exécutées par différents secteurs.

• Système d'aide à la décision – Les GDSS (de l'anglais *Group Decision Support Systems*) sont conçus pour faciliter la prise de décisions en implémentant un sorte de salle de réunion électronique apportant de nombreux outils (par exemple votes, annotation d'idées, *brainstorming*, *etc.*) [47]. L'anonymat et le droit de parole sont des fonctionnalités normalement mises en œuvre dans ces systèmes pour encourager les utilisateurs à s'engager dans le processus de prise de décision.

• Outil de partage d'application – Ce type de logiciel permet à plusieurs utilisateurs (travaillant sur des ordinateurs différents) d'utiliser simultanément une application hébergée sur un seul ordinateur (normalement représenté par un serveur) [1] [5]. Cette fonctionnalité est très souvent implémentée en déportant la fenêtre partagée (voire tout le bureau [55]) vers les machines des autres utilisateurs.

• Editeur partagé – Ces systèmes d'édition conjointe (de l'anglais *shared editing* [55]) permettent à un groupe d'utilisateurs d'éditer collectivement un document partagé. Ils peuvent être utilisés de façon synchrone ou asynchrone, et ils offrent en général des mécanismes de gestion de versions. Mais la principale complexité de ces systèmes concerne la gestion des tâches concurrentes [45], lorsque des utilisateurs modifient un même document simultanément.

• Tableau blanc partagé – Le système de tableau blanc partagé, comme son nom l'indique, met à disposition une surface de dessin accessible par plusieurs utilisateurs. Il permet ainsi à des utilisateurs de travailler d'une manière synchrone sur des documents 2D. Il est par exemple possible de réaliser des captures d'écran pour les annoter dans le but d'expliquer une idée ou un concept [37].

• Audioconférence – Les outils d'audioconférence permettent aux utilisateurs de parler à plusieurs. Si d'un coté la qualité de transmission joue un rôle important pour la compréhension de la communication, les flux audio ne consomment pas énormément de bande passante. La communication audio est l'un des moyens de communication le plus riche au niveau signifiant, mais l'utilisation d'un système d'audioconférence à plusieurs comme seul moyen de communication peut entraîner des difficultés dans l'identification des interlocuteurs [39].

• Vidéoconférence – Ces systèmes permettent aux utilisateurs distants de se réunir et communiquer par l'intermédiaire d'un support audio et vidéo en même temps. Si la vidéo peut donner une sensation de présence plus forte que l'audioconférence seule, une vidéoconférence nécessite de disposer d'une bande passante capable de diffuser et recevoir des données audio et vidéo avec une qualité acceptable. Dans [60] les auteurs suggèrent que, l'audio ayant un rôle

plus important que la vidéo pour supporter la collaboration, les systèmes de vidéoconférence ont intérêt à dégrader en priorité la vidéo pour le bénéfice de la qualité de l'audio.

• Agenda partagé – Également connu comme calendriers partagés (de l'anglais *group calendars* [51]), ces systèmes permettent de résoudre des problèmes concernant la planification de tâches. Normalement ces systèmes incluent des fonctionnalités telles que la détection d'incompatibilités dans la planification d'une tâche ou la détermination de plages horaires communes aux membres d'un groupe. Cela permet, par exemple, à l'organisateur d'une réunion d'avoir une vision claire de la disponibilité des acteurs d'un projet.

• Environnement collaboratif virtuel – Les CVEs (de l'anglais *Collaborative Virtual Environments*) représentent une catégorie importante de systèmes de TCAO qui fournissent des facilités de collaboration à travers l'implémentation d'un espace virtuel réparti. L'espace virtuel peut être représenté par de simples environnements textuels, comme dans le cas des environnements *Multi-User Dimension* (MUD) tel que *MOOSE Crossing* [8]. Mais très souvent il s'appuie sur de riches scènes 3D partagées (ce qu'on appelle des mondes virtuels), comme par exemple [67] et [72]. L'utilisation de la réalité virtuelle est encouragée à cause de sa capacité importante de modélisation et d'interactivité, permettant aux CVEs de résoudre plusieurs problèmes concernant les applications de TCAO : (i) la notion de présence ; (ii) la représentation des métaphores du monde réel (des gestes humains) et des objets (simulations en 3D) ; (iii) la réduction des coûts de transmissions à travers le réseau (notamment par rapport aux systèmes de vidéoconférence).

• Plate-forme collaborative – Cette catégorie spéciale de collecticiels représente en fait des applications qui rassemblent dans un même environnent intégré différents outils associés aux catégories précédentes. Par ailleurs, pour Schmidt et Rodden [57], le caractère dynamique du travail coopératif et ses articulations sont tels que les efforts devraient se tourner vers la mise au point de ces plates-formes fournissant un large éventail de support pour le TCAO : support de partage et d'échange d'information (partage d'application, éditeur partagé, *etc.*) ; support pour la communication (messagerie instantanée, vidéoconférence, *etc.*) ; support pour la prise de décisions, *etc*.

3. Quelques exemples d'applications classifiés selon leur mode d'interaction

Thierry VILLEMUR [62] a classifié ces application selon leur mode d'interaction synchrone ou asynchrone pour le but de faire un bilan sur leur évolution au cours de ces dix dernières années, on a adopté cette classification afin d'éclaircir les étapes cruciales de cette évolution en émergeant les outils essentiels qui étaient la cause de cette dernière.

3.1 Collecticiel asynchrone

Les systèmes asynchrones ont été les premiers développés, à partir des années 90. Nous les avons classés en deux catégories :

- Les messageries
- Les systèmes basés sur le Web

3.1.1 Messageries avancées

Ces systèmes offrent des fonctionnalités supplémentaires par rapport aux messageries classiques, avec la prise en compte de groupes d'utilisateurs, de leur organisation et de leurs caractéristiques respectives. Les premiers systèmes ont été très utilisés dans les domaines d'assistance des procédures de bureautique ou de contrôle du déroulement d'opérations (workflow). Par la suite, ces messageries ont été couplées avec des bases de données classiques, puis elles ont évolué en des messageries multimédias complexes. Pour illustrer cette évolution, nous citons trois réalisations : un projet de recherche, un outil courant et un projet qui a débouché sur un prototype avancé.

3.1.1.1 GRoup Activity Environment

Le projet de recherche GRoup Activity Environment (GRACE), s'est déroulé sur deux ans pour développer des communications asynchrones de groupe dans un environnement Open System Interconnection (OSI). Il a permis de développer un modèle conceptuel de groupes asynchrones dont les membres partagent des informations, et de définir un service de communications de groupe pouvant être utilisés par un vaste ensemble d'applications collaboratives asynchrones. Ce service s'appuie sur la messagerie électronique X.400. Les informations manipulées sont structurées de façon hiérarchique, et sont regroupées en domaines qui permettent de relier les membres des groupes aux différentes données manipulées. Les objets du système sont nommés au moyen du répertoire X.500.

Ces travaux ont permis la réalisation de deux applications prototype. La première est un service d'aide (Help Desk) pour un département informatisé. Les personnes connectées au système émettent des demandes lorsqu'elles rencontrent des problèmes. Le système trie et aiguille les questions suivant les compétences des gens. Les personnes ayant un rôle de spécialiste reçoivent directement ces demandes, et peuvent formuler les réponses. La deuxième application proposée est une conférence asynchrone entre groupes d'utilisateurs, équivalente à un tableau de bulletins. Chaque domaine forme un forum portant sur un sujet précis, le contrôle des informations échangées étant fait par un membre modérateur. Les personnes joignent ou quittent les domaines définis, suivant leurs centres d'intérêt.

3.1.1.2 Lotus Notes

Lotus Notes, est un système collecticiel basé sur Internet pour la collaboration asynchrone entre groupes de travail. Sa première version a été réalisée en 1982. Lotus Notes fonctionne avec un type unique de base de données qui permet la combinaison de données structurées (relationnelles) et non structurées (documents, textes...). En liant les communications par courrier électronique à ce type de base de données, Lotus Notes permet la création de canaux de communication et de partage de documents qui peuvent être dédiés à certaines fonctions particulières ou à certains groupes de travail (groupe des développeurs, groupe marketing…), à l'intérieur d'une organisation.

Lotus Notes se compose de :

- une plate-forme de discussion avec une messagerie et des groupes de discussion,

- un générateur de base de données qui gère l'organisation des fichiers et des données,
- un créateur et indexeur de documents qui leur attribue différentes propriétés et différents niveaux d'accès.

Ce découpage facilite l'organisation, le partage, le stockage et les modifications qui peuvent être réalisées par plusieurs utilisateurs.

Les documents non structurés des utilisateurs de Lotus Notes sont inclus dans la base de données qui leur associe des champs utilisables pour la recherche et l'indexation. Un lien étroit est maintenu entre les messages échangés et les documents de la base de données. Les documents sont classés dans des sections organisées de façon hiérarchique. De plus, Lotus Notes autorise et maintient la création de fils de discussion, en ordonnant les messages requête et réponses, et en les attachant aux documents de référence de la discussion. Toute recherche renvoie le dernier état de la discussion, et garantit que les documents de toute discussion en cours, potentiellement modifiés, sont mis à jour avant d'y accéder.

Les dernières versions de Lotus Notes s'interfacent avec les navigateurs classiques.

Il existe d'autres environnements concurrents, mais qui offrent des fonctionnalités similaires. Les plus répandus sont :

- Microsoft Exchange
- Novell GroupWise
- Open-Xchange

3.1.1.3 Design Implementation and operation of a distributed Annotation Environment

L'application DIANE [66], [17], issue du projet européen Design Implementation and operation of a distributed ANnotation Environment (juillet 1996 - septembre 1998), est un système d'annotations multimédia distribuées capable de manipuler des données informatiques, des liens hypertextes, des sons, des images, des captures d'écran, et des mouvements de souris. Les annotations créées sur n'importe quelle application, sont stockées dans des espaces partagés qui autorisent leur recherche et leur présentation.

L'architecture de DIANE, distribuée, se compose de terminaux utilisateurs et de serveurs d'annotations, qui sont utilisés pour stocker et pour accéder aux documents annotés. DIANE inclut un système de transport de données qui autorise l'accès aux serveurs d'annotations soit de façon interactive (temps réel), soit de façon asynchrone (par message expédié, puis stocké dans une messagerie), suivant le type de réseau utilisé. DIANE permet ainsi l'utilisation de réseaux rapides tels que ATM ou plus conventionnels comme le réseau téléphonique commuté. La présentation des documents multimédias se fait en synchronisant les flux de données qui les composent.

Le développement de DIANE est fait en JAVA. Le prototype a été évalué grâce à des essais dans plusieurs domaines d'utilisation, en particulier :

La télé-médecine: Les médecins pathologistes du département des pathologies de l'Hôpital Général de Mandesa et de l'Hôpital Universitaire du Vall d'Ebron à Barcelone (Espagne) ont utilisé DIANE pour se consulter les uns les autres en cas de diagnostic difficile et pour préparer des sessions de téléenseignement sur des sujets spécifiques.

Le télé-apprentissage: Le laboratoire European Centre for Parallel Computing at Vienna (VCPC) est un centre spécialisé dans les outils et techniques de programmation parallèle. Il a produit avec DIANE des supports d'apprentissage sous la forme d'unités de cours.

3.1.1.4 Evolution des messageries avancées

La figure 1.2 résume l'évolution des systèmes de messageries collaboratives. Les messages textuels sont manipulés au travers de fonctions avancées par les systèmes des années 80. Dans les années 90, les systèmes de messageries collaboratives incluent des bases de données qui associent les messages à des contextes, maintenant ainsi l'ensemble des conversations liées à un sujet. L'arrivée du multimédia dans les années 95 enrichit et transforme les contenus textuels. Finalement, l'évolution la plus proche de nous concerne l'intégration des systèmes de messageries collaboratives aux navigateurs Web, ces derniers servant d'interface utilisateur universel pour l'accès au courrier électronique.

3.1.2 Systèmes basés sur le web

Le Web a été créé à l'origine dans les années 90 pour offrir une interface unique pour l'accès à des données diverses, hétérogènes et distribuées. Cette interface unique, appelée navigateur, communique avec des serveurs Web et utilisant le protocole Hyper Text Transfer Protocol (HTTP). Au fur et à mesure de l'évolution technologique du Web, les navigateurs ont été amenés à accéder et à traiter de nouvelles données dont les valeurs changent dans le temps (données dynamiques), qui évoluent de façon interactive (formulaires), et qui possèdent des contraintes temporelles (flux multimédias). La dernière évolution du Web concerne l'intégration de l'accès à des applications, les navigateurs devenant l'interface distante de ces applications. On parle souvent dans ce cas de portails Web.

D'après la définition du grand dictionnaire terminologique [65], un portail est un « Site Web dont la page d'accueil propose, en plus d'un moteur de recherche, des hyperliens avec une foule d'informations, et des services applicatifs utiles et attrayants ». Comme exemple de service, on peut trouver un serveur de courrier électronique gratuit, une sélection de moteurs de recherche, des actualités, la météo, les cotes de la Bourse, des raccourcis pour les achats en ligne.

« L'objectif des portails est, du point de vue de leurs créateurs, d'attirer et de fidéliser les internautes au point de devenir leur porte d'entrée dans Internet, c'est-à-dire la page de démarrage du plus grand nombre d'entre eux, ce qui peut entraîner des revenus importants, notamment en publicité et en vente de liens. Il existe des portails généralistes et des portails thématiques, dont certains sont personnalisables. Yahoo.com, Excite.com, France.com et Canada.com sont des exemples de portails. »

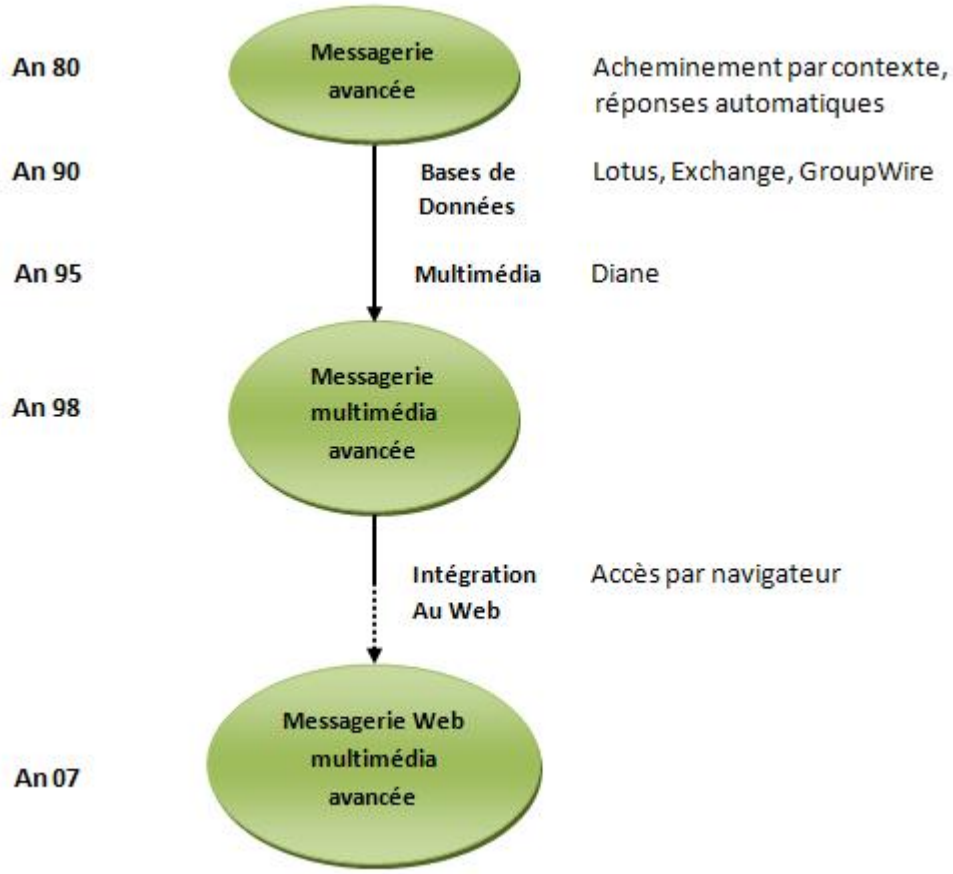


Figure 1.2. Evolution des systèmes de messageries collaboratives

La dimension collaborative a été ajoutée au Web dans les années 95, avec notamment l'outil Basic Support for Collaborative Work (BSCW). Cet outil généraliste, en plus de l'accès aux documents stockés sur un serveur, définit des groupes d'utilisateurs avec des droits d'accès. Ces droits utilisateurs structurent l'espace des documents et définissent quelles informations peuvent être accédées par les utilisateurs. D'autres outils sont plus ciblés vers des domaines spécifiques, comme le téléenseignement.

3.1.2.1 WebCT

La plate-forme WebCT (cf. Figure 1.3) est un système basé sur Internet qui permet le développement, la gestion, puis la diffusion asynchrone de modules de formation. Le cœur de WebCT s'appuie sur un serveur Web.

Le mode auteur, accessible à distance, assiste le concepteur et lui permet d'organiser le contenu de ses cours en pages et en modules. Le contenu de ces modules peut être réalisé par d'autres outils de présentation, comme Macromedia Dreamweaver, Microsoft FrontPage ou Microsoft Powerpoint. WebCT peut récupérer tout document converti au format HTML.

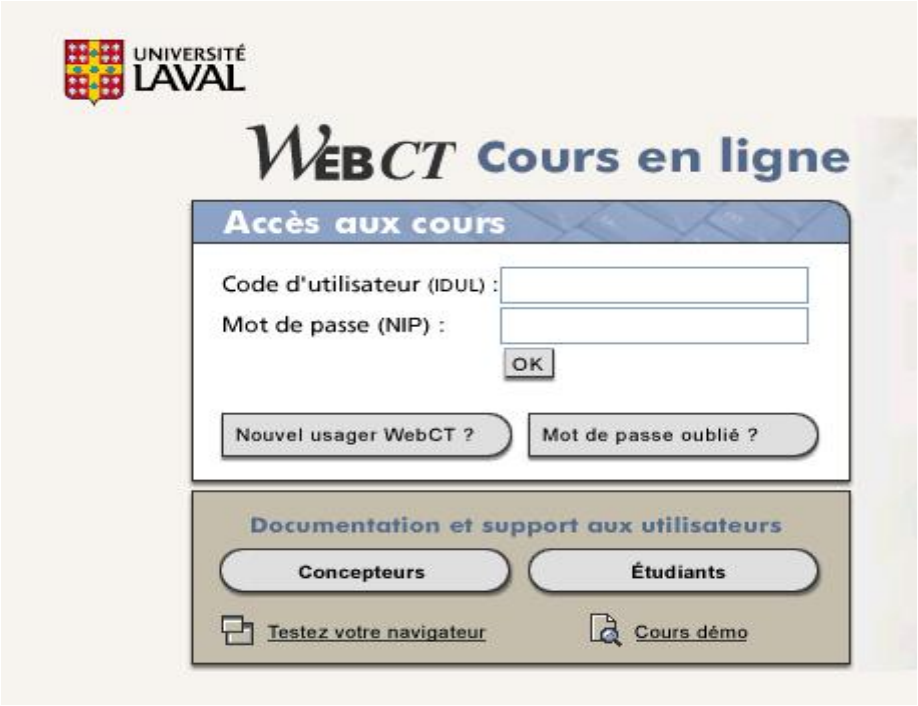


Figure 1.3. Page d'accueil de la plateforme WebCT

Les cours réalisés peuvent contenir :

- Tout document de type texte, image, vidéo et audio
- Des pages d'aide comme des index pour des recherches, des glossaires
- Des hyperliens vers d'autres documents ou sites externes

Les utilisateurs distants n'ont pas besoin d'installer de logiciel spécifique pour accéder aux cours : ils utilisent un navigateur standard. Les étudiants enregistrés à certains cours délivrés par WebCT communiquent avec leur instructeur en utilisant la messagerie électronique, des groupes de discussion ou un dialogue textuel.

L'évaluation des étudiants se fait au travers de Questionnaires à Choix Multiples et de Devoirs.

Les enseignants peuvent aussi recevoir des données qui mesurent l'impact de leur cours.

WebCT contient en dernier lieu des fonctions d'administration intégrées, pour le suivi et la gestion des apprenants. L'outil de syllabus décrit la structure de l'enseignement délivré. Un calendrier définit les dates de suivi des cours ainsi que les dates d'examen et de rendu des devoirs. En dernier lieu des outils de gestion de notes et de suivi de l'avancée des étudiants sont aussi disponibles.

WebCT est l'outil le plus utilisé dans le monde (des centaines d'établissement et d'entreprises réparties sur plus de 80 pays) pour la gestion et la diffusion de cours. Nous citerons notamment le programme d'étude numérique de l'université de Liverpool, qui est l'un des premiers et l'un des plus avancés en Europe. Tous les supports de cours des étudiants sont

numérisés dans le format HTML et accessibles depuis chez eux. L'Université de Toulouse 1 a aussi mis en place un enseignement juridique pour une licence en droit, appelée Cyberlicence, qui s'appuie sur WebCT. Finalement, la formation continue à l'INSA de Toulouse se sert de WebCT comme support de formation.

Quelques environnements importants avec des fonctionnalités similaires sont :

- Learning space
- Scolastance

3.1.2.2 Evolution des systèmes basés sur le Web

La figure 1.4 synthétise l'évolution des systèmes de Web collaboratif. Les premiers systèmes, généraux, introduisent à partir de 1995 une dimension de groupe sous forme de droits d'accès aux documents Web. Par la suite, ces systèmes se spécialisent selon certains domaines d'application. Le téléenseignement est l'un des plus caractéristiques, mais d'autres systèmes se développent, dédiés à la gestion de documents techniques ou bien de données produit. Ces systèmes évoluent naturellement vers des portails Web qui intègrent d'autres applications, pas forcément collecticiel, mais qui sont très liées au domaine d'application choisi. Par exemple, dans le cas du téléenseignement, des applicatifs de suivi des étudiants, ainsi que de gestion des salles, sont couplés au système de Web collaboratif et sont accessibles au travers d'une même interface portail Web.

3.2 Collecticiel synchrone

Les plates-formes de collecticiel synchrone [38] autorisent des échanges interactifs (temps réel) entre les membres distribués géographiquement, en co-présence virtuelle. Ils supportent des interactions synchrones entre utilisateurs.

Les premiers systèmes collecticiels synchrones ne manipulent que du texte. Par la suite, l'énorme progrès de l'informatique multimédia et l'avènement de réseaux à haut débit depuis les années 90 ont fortement dynamisé le collecticiel pour le travail collaboratif

synchrone. Le transport de flux vidéo multimédias est désormais possible entre groupes d'utilisateurs distribués géographiquement, ce qui a entraîné l'apparition de nouvelles plates-formes multimédia support de travail de groupes d'utilisateurs en co-présence virtuelle [38].

Au début, les plates-formes de collecticiel synchrone comprennent un ensemble d'outils peu couplés entre eux qui supportent les fonctionnalités élémentaires d'échange d'information pour assurer le bon déroulement du travail associé au groupe :

(i) communications informelles entre utilisateurs

(ii) partage d'applications et d'espaces de travail

(iii) partage de documents

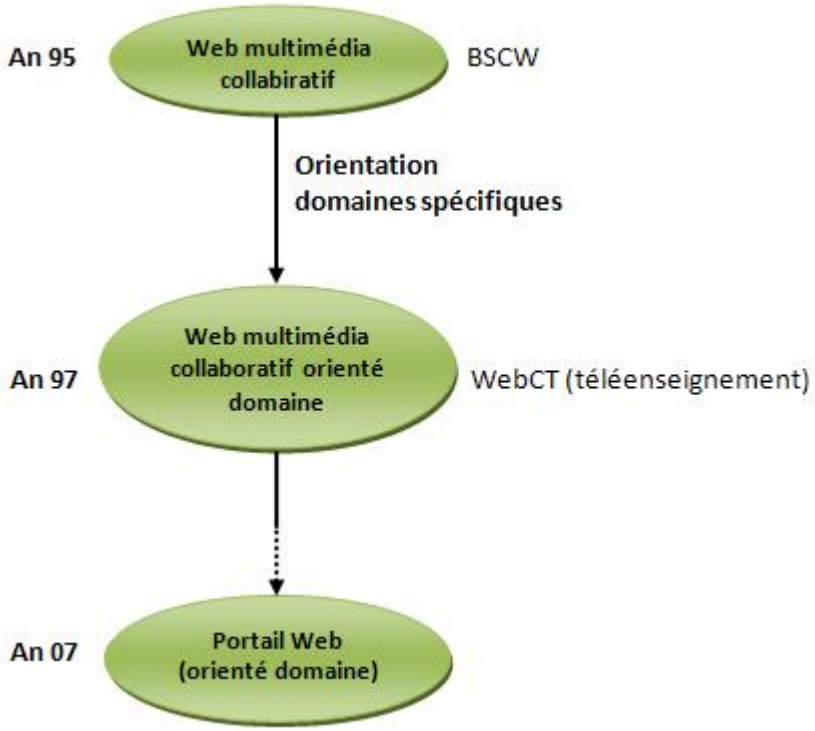


Figure 1.4. Evolution des systèmes de Web collaboratif

Chacune des fonctionnalités précédentes est illustrée par une sélection d'outils significatifs de collecticiel synchrone.

Par la suite, les plates-formes de collecticiel synchrone mettent en œuvre un ensemble de services qui coordonnent les outils collecticiels entre eux, et qui gèrent les utilisateurs et les ressources dont ils ont besoin pour accomplir leur travail de groupe. Ces services, souvent regroupés sous le terme "coordination", forment la "glue logicielle" des outils collecticiels.

3.2.1 Outils pour des communications informelles

Les outils de communication informelle supportent les dialogues réalisés au sein du groupe synchrone. Deux classes ont été isolées pour ces communications, selon le type de média utilisé dans les échanges :

- l'Internet Chat Relay, qui manipule des informations textuelles
- les vidéoconférences, qui échangent des flux audio et vidéo

3.2.1.1 Internet Chat Relay (IRC)

"Internet Relay Chat" (IRC), appelé couramment "Chat", date de 1988. Ce service n'utilise que le texte pour faire des conversations de groupe en temps réel, avec des personnes connectées à internet dans des "espaces" virtuels publics ("room"). Une fois connecté à un espace virtuel identifié, les messages des participants de cet espace apparaissent à l'écran les uns à la suite des autres. Chaque participant est identifié par un surnom. IRC est un protocole qui fonctionne selon le modèle client/serveur. Il faut simplement se brancher à un serveur,

choisir un canal de discussion, puis un espace de discussion, avec un logiciel client. Les logiciels client IRC les plus utilisés sont Visual IRC (VIRC) et mIRC.

L'IRC est encore actuellement très utilisé lors de communications de groupes synchrones comme solution de secours, du fait de sa simplicité, de sa robustesse et surtout de la très faible bande passante nécessaire pour dialoguer.

3.2.1.2 Vidéoconférence sur Internet

Les vidéoconférences ont été les premiers outils à traiter les données multimédias sur des ordinateurs. Cette intégration au monde informatique s'est faite [65], grâce à l'évolution des trois axes suivants :

- La technologie matérielle utilisée pour l'acquisition et la capture des données multimédias
- L'évolution des réseaux, non seulement en terme d'augmentation de débit mais aussi en terme de services nouveaux (garanties temporelles et diffusion multipoint)
- La proposition de normes de codage communément adoptées et implantées par les outils

Les paragraphes suivants illustrent ces trois axes, au travers de réalisations significatives sélectionnées.

a. Evolution matérielle

Les premières cartes de capture, de numérisation et d'affichage pour la vidéoconférence ont été disponibles dans les années 90. Cette évolution matérielle s'est faite selon trois grandes étapes.

Les premières cartes font des acquisitions en mode analogique, aux formats PAL, SECAM ou NTSC. Un exemple de ce type de matériel est les cartes Parallax. Les étapes de numérisation, de compression, puis d'affichage s'effectuent grâce aux processeurs dédiés de la carte, ce qui a pour principale avantage de ne pas alourdir la charge du processeur de la machine. Les données numériques obtenues, codées selon le principe de l'algorithme Motion JPEG, sont stockées dans un format propriétaire. Leurs trois grands inconvénients : (1) leur coût (de 10 kEuro à 150 kEuro) ; (2) la disponibilité que sur station de travail ; (3) des formats de données propriétaires, ont limité leur diffusion. Leur production a cessé fin 1998.

Les années 95 voient l'apparition de nouveaux types de cartes d'acquisition et de numérisation, telles que les cartes Osprey de la société Viewcast. Elles possèdent des entrées analogiques comme précédemment, mais aussi une entrée numérique de type DV. Leur coût raisonnable (de 165 Euros à 900 Euros), leur adaptation à différents types de matériel (PCs et stations de travail) et l'utilisation de formats de données standard H.261, favorisent leur diffusion. Leur inconvénient, relatif car vite compensé par la constante évolution de la puissance de traitement des processeurs, est d'utiliser le processeur de la machine hôte pour réaliser les traitements de compression.

La dernière évolution consiste en la technologie directement numérique de type Webcam. L'image, numérisée par un capteur qui saisit directement les images sous forme de matrice de pixels, est acheminée sur la machine hôte (un PC classique) au travers d'un bus numérique tel que l'USB. La compression et la mise en format standard sont faites par le processeur de la machine. Le prix des Webcam, très bas (moins de 50 Euros), et leur facilité d'installation sur une connectique externe, expliquent leur popularité d'utilisation. Elles supplantent les cartes d'acquisition et sont les périphériques les plus répandus pour la vidéoconférence de bureau.

b. Services réseau

Deux types de services réseau sont très importants pour l'utilisation de logiciels de vidéoconférence. Le premier type considère l'aspect temporel des données multimédia échangées. Le deuxième type étend les communications à des groupes multipoint, pour permettre des communications interactives entre groupes synchrones.

Services avec garanties temporelles

RTP (Realtime Transport Protocol) et son protocole de contrôle RTCP (Realtime Transport Control Protocol) permettent respectivement de transporter et de contrôler des flots de données qui ont des contraintes temporelles.

Chaque flux RTP est décomposé en un ensemble de paquets émis périodiquement. Chaque paquet est estampillé et contient une date d'horloge d'émission. Le protocole RTP introduit un retard supérieur à la gigue estimée du réseau. La restitution du flux est réalisée à la réception, en utilisant les informations d'horloge de chaque paquet : le récepteur restitue les paquets en respectant leur synchronisation temporelle et il supprime les paquets reçus en retard. Chaque flux multimédia est codé de façon séparée : ainsi, un flux RTP audio est toujours séparé d'un flux RTP vidéo.

Le protocole RTCP associe toujours un flux de contrôle à un flux de donnée. Le but de ce flux est de contrôler la qualité du flux monomédia reçu, en obtenant chez un émetteur un retour sur la qualité, en terme de taux de perte de paquets et de paquets arrivés en retard. Ce retour permet de modifier ou d'adapter la qualité du flux si nécessaire. La quantité de trafic RTCP se limite à 5% du trafic multimédia global.

Les protocoles RTP et RTCP utilisent le protocole UDP, et les flux transmis sont identifiés par deux ports consécutifs : le port pair pour le flux RTP, le port impair pour RTCP.

Diffusion

La diffusion permet d'envoyer en une seule fois une même information à destination d'un ensemble d'utilisateurs.

Proposé par S. Deering dans les années 90, IP multicast est le premier protocole de diffusion à être disponible au dessus d'Internet. Ce protocole définit des groupes ouverts d'utilisateurs, identifiés par une adresse IP spécifique et un numéro de port. Le service de diffusion mis en place est conforme aux caractéristiques d'IP, à savoir non fiable, sans garantie d'ordre, ni de délivrance. Le succès de ce protocole vient de son rapide déploiement, à partir de

92 dans le réseau expérimental Mbone (multicast bone), déployé au niveau mondial en utilisant une technique de tunneling (fabrication d'un réseau de diffusion statique au dessus de l'Internet mondial). Ce déploiement expérimental a permis de mettre au point et de tester des outils de vidéoconférence multiutilisateurs pour des téléséminaires et des réunions virtuelles.

Actuellement, du fait du succès du Mbone, de nouveaux protocoles de diffusion, notamment Protocol Independent Multicast Sparse Mode (PIM-SM) et Multiprotocol Extensions for Border Gateway 4 (MBGP), ont été intégrés dans les routeurs d'Internet, ce qui permet de supporter un mode de diffusion natif, de même niveau que les transmissions de paquets IP.

Outils du Mbone

Les logiciels "Vic" et "Rat" (cf. Figure 1.5) sont les outils de vidéoconférence les plus utilisés pour des vidéo et audioconférences au dessus du Mbone. Ils sont particulièrement illustratifs des possibilités et des limites du Mbone.

Vic (VIdeoConferencing) et Rat (Robust Audio Tool) sont deux logiciels de vidéo/audio conférence qui font partie d'une suite d'outils de collecticiel synchrone, développée par le "Network Research Group" du laboratoire "Lawrence Berkeley National Laboratory" en collaboration avec l'Université de Berkeley en Californie. Les premières versions furent développées par Steven McCanne, dans le cadre de sa thèse en 1996.

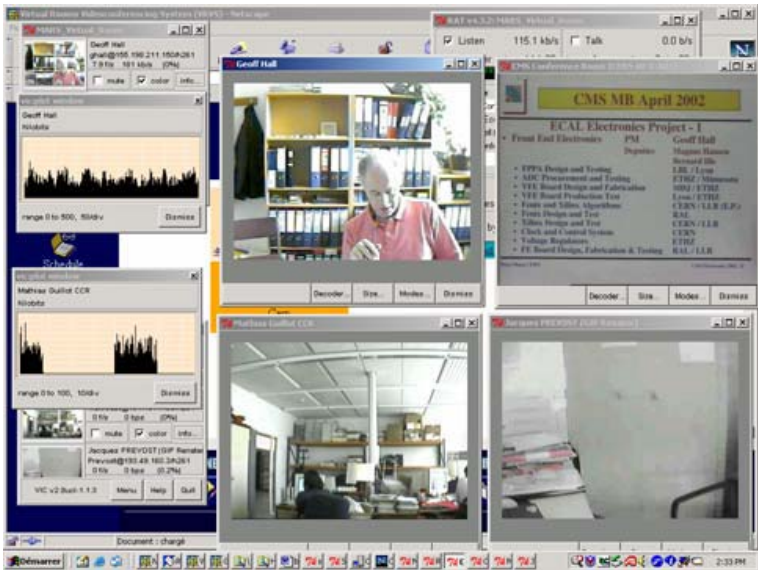


Figure 1.5. Utilisation de Vic et de Rat

Le codage et la compression des trames vidéo, originellement basés sur la recommandation H.261, ont évolué pour être compatibles avec la norme H.263 qui est la référence et la norme la plus utilisée en terme de codecs vidéo. Ces deux normes manipulent le plus souvent des images de 352x288 pixels, ou format CIF (Common InterMediate Format). Elles font partie du cadre de normalisation H.323, cadre le plus utilisé en terme de vidéoconférence.

Les formats de compression audio les plus couramment supportés sont G.711 PCM, G.726 ADCPM et GSM. La norme G.711 correspond au standard international pour transmettre la voix téléphonique sur un canal numérique au débit de 64 kbits/sec. G.726, plus performante,

requiert des canaux au débit de 16 à 40 kbits/sec, suivant la qualité audio souhaitée. La norme GSM, qui provient de travaux de l'ETSI (European Telecommunications Standards Institute), est un codec adapté au téléphone mobile. Elle offre un débit de 13 kbits/sec.

La synchronisation des images, ainsi que la synchronisation audio se font en utilisant le protocole Real-time Transport Protocol (RTP).

"Vic" et "Rat" peuvent initialement fonctionner en mode point à point en utilisant le protocole de Transport UDP. Cependant, le grand atout de ces outils est de supporter des communications multipoint, au dessus du protocole IP multicast. Il combine ainsi les avantages de la synchronisation multimédia fournie par le protocole RTP avec celui d'une vraie diffusion multipoint.

Ces deux outils sont très utilisés par la communauté réseau pour leurs nouveaux tests et développements, mais aussi par plusieurs communautés d'utilisateurs (physiciens, académiques...). Ils ont été portés sur de nombreuses plates-formes et systèmes d'exploitation Unix et Windows. Ils sont donc utilisés à titre expérimental, ce qui garantit leur évolution et maintenance, mais aussi en mode opérationnel pour des téléconférences et des téléséminaires.

c. Norme et architecture pour la vidéoconférence : H.323

La norme H.323 est un ensemble de standards et de recommandations pour les conférences multimédias sur des réseaux de paquets commutés, ces réseaux pouvant être de type intranet ou internet. H.323 est définie pour harmoniser la voix sur IP. Elle fait partie de la série H.32x qui traite de la vidéoconférence au travers différents réseaux, depuis des réseaux téléphoniques bas débit, de type ISDN, LAN ou Internet. Le trafic échangé peut également être géré et contrôlé.

Composants

Cette norme comprend quatre composants majeurs : les terminaux, les passerelles, les gatekeeper et les unités multipoint.

Les terminaux H.323 sont les terminaux clients finals des utilisateurs. Ils fournissent des communications bidirectionnelles vers d'autres équipements terminaux. La communication peut se faire par :

(a) un canal audio seulement,

(b) deux canaux audio et donnée,

(c) deux canaux audio et vidéo,

(d) trois canaux audio, vidéo et donnée.

Ces terminaux peuvent être matériels ou logiciels.

Les passerelles (gateways) sont un point terminal H.323. Elles fournissent des communications bidirectionnelles entre des terminaux qui appartiennent à des réseaux constitués de piles de protocole différentes, par exemple entre un réseau IP et le RTC ou RNIS.

Elles assurent, entre autres, le codage et le décodage de la voix, la mise en paquets, la suppression de l'écho.

Le gatekeeper H.323 est un élément optionnel pour définir et gérer des zones de conférence H.323. Lorsqu'un gatekeeper est activé dans un réseau local, tous les clients H.323 de ce réseau doivent l'utiliser. Un gatekeeper contrôle la qualité opérationnelle du réseau en réalisant les fonctions suivantes pour les composants de type terminaux, passerelles et unités multipoint déclarés dans sa zone.

- Contrôle d'admission : il autorise seulement l'accès aux composants déjà enregistrés et reconnus, limitant de ce fait le nombre de terminaux en cours de communication.
- Contrôle de bande passante : lors du contrôle d'admission, le gatekeeper vérifie que le réseau possède encore suffisamment de bande passante avant d'accepter l'appel.
- Translation d'adresse : le gatekeeper traduit les identifiants des terminaux, proches de numéros de téléphone, et leurs alias, en adresses reconnaissables par le réseau, par exemple des adresses IP.

Finalement, on peut considérer un gatekeeper comme une sorte de "routeur" applicatif H.323.

L'Unité de Contrôle Multipoint (MCU) supporte des conférences entre trois utilisateurs ou plus. Cette unité se compose d'un Contrôleur Multipoint (MC) obligatoire et d'un Processeur Multipoint (MP) facultatif. Le contrôleur MC se charge des négociations entre tous les terminaux pour déterminer les capacités de traitement audio et vidéo communes aux différents terminaux. Il contrôle les ressources de la conférence et détermine, si nécessaire, quels flux audio ou vidéo peuvent être émis en mode diffusion. Le processeur MP intervient lorsque le réseau n'offre que des communications point à point. Il centralise les flux audio et vidéo provenant des terminaux, les traite, les filtre, puis les rediffuse vers chacun d'entre eux. Ce mode centralisé simule une diffusion réelle.

Remarque : Bien que la norme H.323 prévoie un ensemble de communications allant du mode centralisé au décentralisé, la plupart des implantations actuelles ne supporte que le mode centralisé. Cependant, des initiatives apparaissent pour faire converger H.323 et la diffusion IP multicast, telle celle du logiciel Microsoft Exchange 2000 Conferencing Server.

Normes

La figure 1.6 présente la pile des principaux protocoles qui composent la norme H.323.

Protocoles de contrôle

La norme H.225 s'occupe de la signalisation des appels et contrôle la mise en paquets des flux de données multimédias.

La norme Registration Admission Status (RAS) gère la découverte des gatekeeper, l'enregistrement auprès d'un gatekeeper, la résolution de noms entre les alias H.323 et les adresses IP. Elle s'occupe aussi du contrôle d'admission et du contrôle de la bande passante.

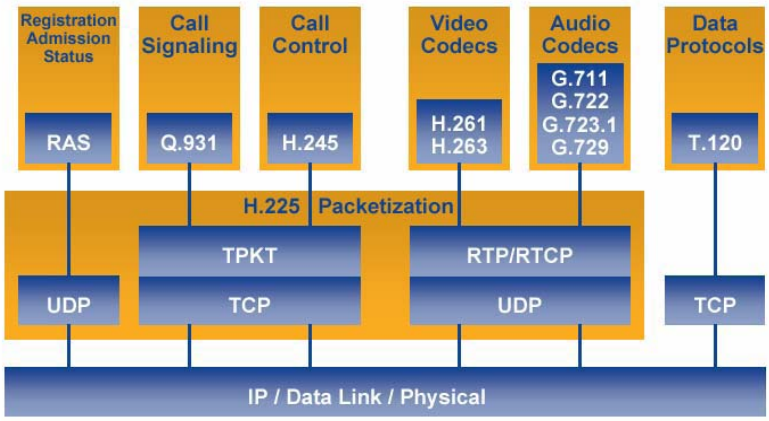


Figure 1.6. Principaux protocoles de la norme H.323

Le protocole de signalisation d'appel Q.931 est identique à celui utilisé pour des canaux ISDN.

La norme H.245 pour le contrôle d'appel se charge principalement de la gestion des médias et des flux de données.

Standards pour l'échange de données de conférences. Norme T.120

La norme T.120 regroupe un ensemble de spécifications pour standardiser les échanges de données entre plusieurs types d'applications collaboratives. Nous citerons brièvement les principales :

La norme T.126 spécifie le format des données échangées par une application de type tableau blanc. Elle gère l'espace multiutilisateur fourni par ce type d'application.

La norme T.127 précise le format d'échange des fichiers et des images partagées par un tableau blanc.

Le standard T.128 autorise le partage d'une application entre plusieurs utilisateurs. Un groupe d'utilisateurs peut ainsi partager et interagir sur une même instance d'application.

La norme T.134 décrit le dialogue textuel multiutilisateur ("chat").

Protocoles d'échanges multimédias

Les protocoles RTP et RTCP, décrits précédemment, sont utilisés pour les échanges multimédias, numérisés au travers des codecs H.261 et H.263 pour la vidéo, et des codecs G.711 à G.729 pour l'audio.

Vidéoconférences H.323

Les produits de vidéoconférence les plus répandus qui respectent le format H.323 sont :

- Microsoft Netmeeting
- Microsoft Messenger
- Cu See Me de la société First Virtual Communications

d. Tendance d'évolution des outils de vidéoconférence

La figure 1.7 synthétise les grandes étapes qui marquent le développement des systèmes de vidéoconférence, selon les trois axes précédemment identifiés :

- La technologie matérielle
- Les services réseau
- La normalisation des codages

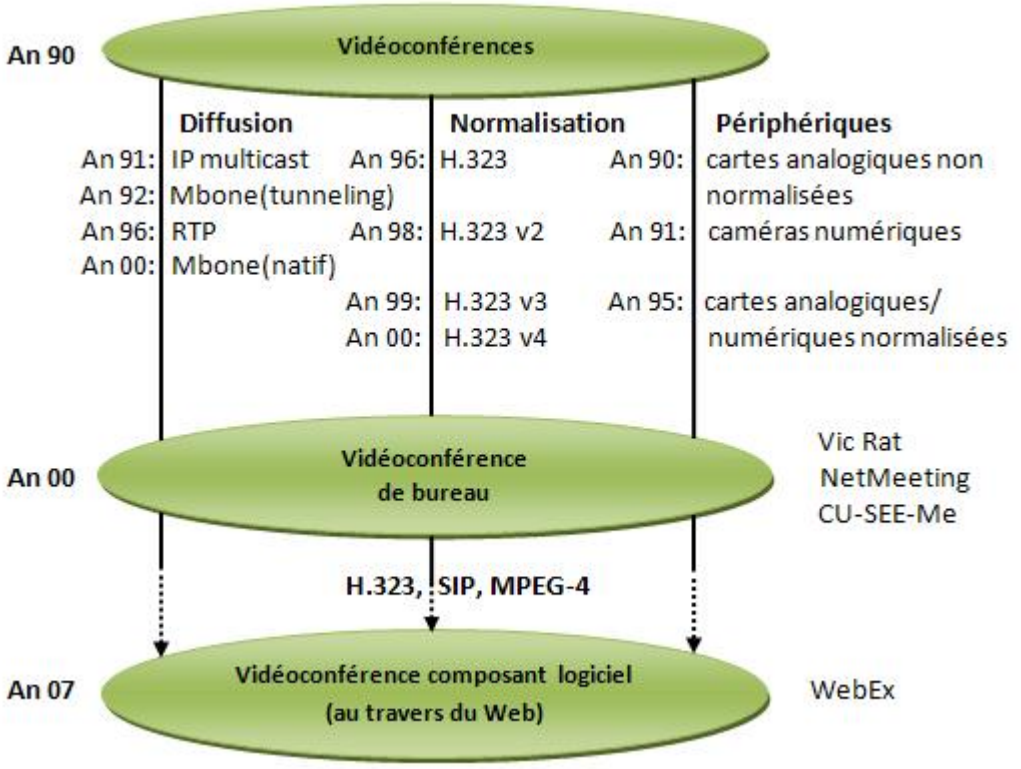


Figure 1.7. Evolution des systèmes de vidéoconférence

Les périphériques de capture et de manipulation de vidéo numérique poursuivent leur évolution en termes de miniaturisation et d'augmentation de la définition des images traitées en nombre de pixels.

Les protocoles de diffusion réseau actuels opérationnels (Protocol Independent Multicast Sparse Mode (PIM-SM) et Multiprotocol Extensions for Border Gateway 4 (MBGP)) ont été établis pour des groupes qui comportent un nombre réduit d'utilisateurs (quelques dizaines) fortement distribués. Ces protocoles sont mal adaptés (i) pour des groupes de taille plus conséquente (de milliers d'utilisateurs et plus) (ii) qui nécessitent des garanties de Qualité de Service ou (iii) qui ont besoin d'accès sécurisés. Ces deux derniers points font partie des évolutions nécessaires pour le support et la mise au point d'outils de vidéoconférence massivement multiutilisateurs.

La normalisation des échanges multimédias pour la vidéoconférence est principalement basée sur la norme H.323. Cependant, cette norme a été développée à l'origine pour des réseaux qui fonctionnent en mode connecté et qui supportent des débits de l'ordre de la

centaine de kbits/sec. Le contrôle et la gestion des flux, qui suivent une philosophie connectée, ne sont plus adaptés aux réseaux actuels, qui respectent une philosophie non connectée compatible avec IP. H.323 évolue vers une simplification des protocoles de gestion, avec notamment une intégration progressive et une adaptation au protocole SIP (Session Initiation Protocol).

De la même façon, les réseaux actuels supportent le transport de tailles d'images bien supérieures à celle définies par le codec H.263. De ce fait, H.263 commence à être supplanté par H.264, ou MPEG-4 Advanced Video Coding, une norme qui a émergé début 2003. H.264 est basé sur un algorithme de compression MPEG-4, et traite des images de taille et de qualité équivalentes ou supérieures aux formats des films actuels. L'outil Video LAN qui tend à supplanter "vic" et "rat" pour la diffusion des téléséminaires est l'exemple type de cette tendance.

Finalement, la tendance générale des systèmes de vidéoconférence est de passer de systèmes spécifiques, avec du matériel dédié, à des logiciels de vidéoconférence de bureau, fonctionnant sur des microordinateurs classiques, et de ce fait intégrables avec d'autres applications. Cette tendance se confirme et se renforce en transformant les applications de vidéoconférence en composants logiciels intégrables au travers de navigateurs Web. L'environnement de téléséminaires WebEx, qui comprend des systèmes de vidéoconférence, de partages d'applications et de documents, est un bon exemple de cette évolution.

3.2.2 Outils de partage d'application

La fonctionnalité de partage d'applications et d'espace de travail est supportée par des outils logiciels qui permettent à plusieurs utilisateurs travaillant sur des ordinateurs différents d'utiliser simultanément et à distance un groupe d'applications hébergé par un autre utilisateur. Les utilisateurs peuvent ainsi travailler de façon collaborative sur des présentations Powerpoint, des programmes graphiques de conception, ou n'importe quel autre applicatif. Les applications partagées tournent chez un seul utilisateur, mais sont rendues visibles chez tous les utilisateurs distants qui se servent du partage d'application. Cette vue est fournie à chaque utilisateur par une fenêtre spéciale de partage d'application qui reproduit exactement l'interface des applications partagées. Au travers de cette vue, chaque participant peut utiliser son clavier et sa souris pour effectuer un contrôle et une commande à distance des applications partagées.

Le partage d'application est très utilisé dans les domaines de :

- La téléassistance, avec l'aide et le dépannage à distance;
- La téléadministration de parc de machines ;
- Le contexte éducationnel, pour partager des applications entre un professeur et un groupe d'élèves.

Les deux exemples de logiciels de partages d'application que nous avons sélectionnés couvrent différemment les domaines d'applications identifiés. Le premier, pcAnywhere est dédié à la téléassistance et la téléadministration, tandis que deuxième, plus universel, peut être utilisé dans les trois domaines d'applications.

3.2.2.1 Symantec pcAnywhere

PcAnywhere (version 11) est un des outils de référence pour la prise en main à distance. Combiné à des fonctions de transfert de fichiers et de gestion distante, il permet de résoudre rapidement les problèmes d'assistance et d'administration de serveurs. La fonctionnalité de prise en main à distance, fiable et sécurisée, permet de se connecter et de dépanner les postes de travail et les serveurs distants, afin d'optimiser la résolution des problèmes rencontrés.

La suite d'outils de gestion à distance exploite la connexion sécurisée entre un système Elève et Maître, permettant ainsi de résoudre les problèmes sans avoir à ouvrir une session de prise en main à distance complète. Les fonctions de gestion à distance intègrent des outils classiques tels que le gestionnaire des tâches, la ligne de commandes et la fonction de modification à distance du registre. La fonction ligne de commandes permet aux administrateurs de mettre en file d'attente plusieurs fichiers et commandes DOS et de les traiter dans l'ordre. Il est ainsi possible de réorganiser, de mettre en attente et de supprimer les fichiers de la file d'attente, le cas échéant. Les transferts de fichiers s'effectuent en tâche de fond afin que les administrateurs puissent travailler sans interruption et continuer à sélectionner des fichiers tout en en transférant d'autres. Les administrateurs peuvent même envoyer des fichiers à plusieurs ordinateurs l'un après l'autre. Une interface utilisateur graphique facilite la navigation entre les applications de transfert de fichiers, de prise en main, et de gestion à distance.

Au niveau technique, pcAnywhere ne fonctionne qu'en mode point à point entre un système Elève et un Maître. L'attribution de ces deux rôles entre les systèmes change au cours de sessions pcAnywhere.

3.2.2.2 Virtual Network Computing (VNC)

L'outil Virtual Network Computing (VNC) a été conçu en 1994 dans les laboratoires AT&T de l'Université de Cambridge (UK). Il s'agit d'un logiciel de contrôle à distance qui permet d'interagir sur un ordinateur (le serveur) en utilisant un autre programme (le visualisateur) qui tourne sur une autre machine connectée à l'Internet (cf. Figure 1.8).

Les deux ordinateurs ne sont pas forcément de même type, et il est possible au travers de VNC de visualiser une machine LINUX serveur depuis un PC Windows client. Plusieurs versions interopérables entre elles existent, une pour les systèmes Windows (appelée WinVNC), et une pour chaque plate-forme UNIX de type SUN, HPUX, Silicon Graphics, et Macintosh. La figure 1.8 montre l'écran d'un client VNC qui tourne sur une station UNIX alors que les applications partagées sont sur un serveur Windows.

La technique de visualisation des applications partagées est originale car elle s'appuie directement sur des captures de zones d'écran sous forme de rectangles bitmap qui sont transmises de façon plus moins compressées. Le modèle d'entrée pour les applications distantes se compose d'un clavier générique et d'une souris multiboutons. Cette technique de visualisation garantit une large adaptation de VNC vis-à-vis des types de machines et des systèmes d'exploitation.

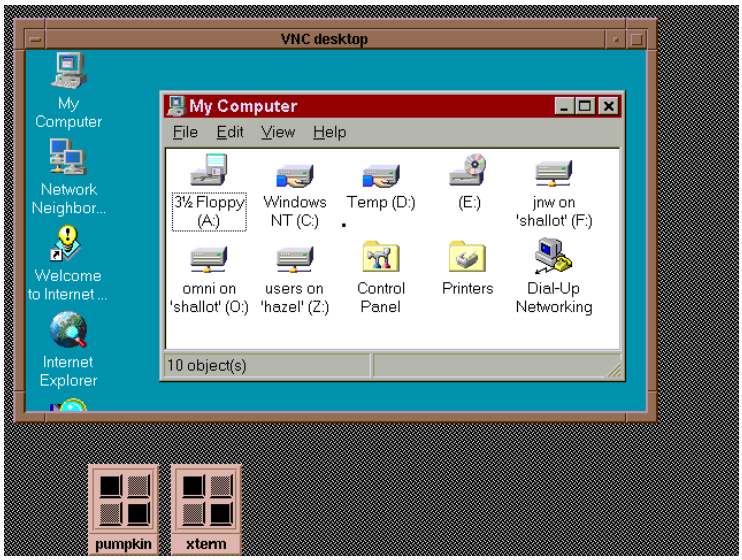


Figure 1.8. Ecran d'un client VNC

La communication entre le serveur et un client se fait avec un lien TCP/IP. VNC possède un mécanisme d'authentification basé sur un login et un mot de passe entre un client et le serveur. Ce mécanisme peut être couplé avec SSH pour augmenter le niveau de sécurité si nécessaire. Deux types de clients sont disponibles: un client natif écrit en C++ et un client universel écrit en JAVA. Ce dernier choix est garant d'une portabilité maximale, ce client JAVA pouvant être récupéré de façon interactive depuis le serveur VNC via une page Web.

Plusieurs clients peuvent se connecter et dialoguer en même temps avec un seul serveur. Les applications partagées sont dans ce cas, visibles et manipulables par l'ensemble des clients.

VNC est développé sous licence logiciel libre, les codes source sont donc gratuits et accessibles.

3.2.2.3 Tendance d'évolution des partages d'application

En s'appuyant sur les deux exemples précédents, la figure 1.9 résume les tendances relevées dans les outils de partage d'application :

- Partage en groupe
- Intégration plus forte aux systèmes d'exploitation

La première est une évolution vers le partage d'applications en groupe. En effet à un instant donné, l'espace n'est pas uniquement partagé entre un utilisateur serveur et un seul utilisateur client en mode point à point (comme pour pcAnywhere), mais est transmis en même temps et en mode multipoint vers un ensemble de clients. Dans ce dernier cas, l'utilisateur serveur partage sa vue d'écran. Cette vue est rediffusée vers tous les autres clients participant au groupe.

La deuxième tendance est une intégration plus forte avec les systèmes d'exploitation. De simples applicatifs autonomes, les partages d'applications sont de plus en plus souvent associés au démarrage d'un système, sous la forme de démons. Cette tendance, possible avec pcAnywhere et VNC, se renforce avec les nouveaux partages d'applications tels XP Messenger.

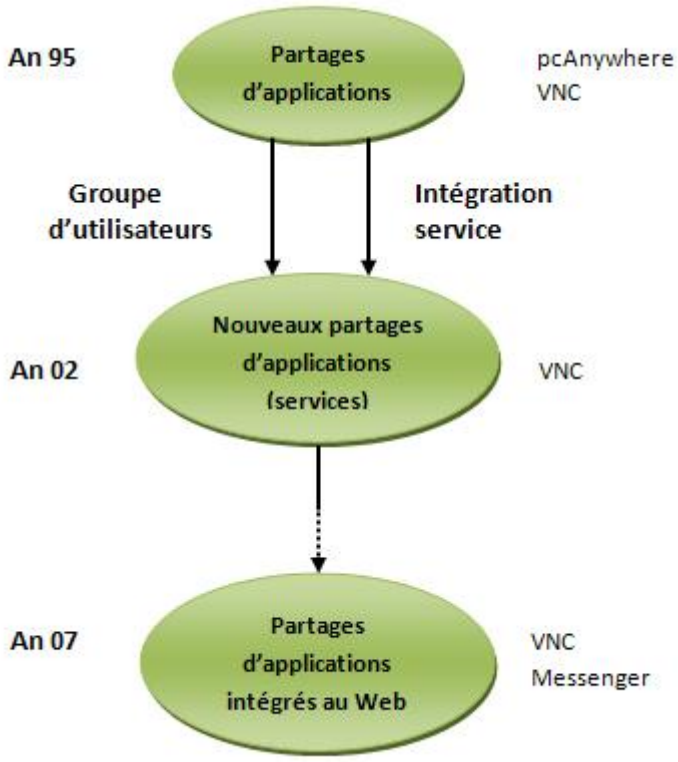


Figure 1.9. Evolution des partages d'application

Finalement, l'intégration au Web se fait par des applicatifs clients spéciaux, tels ceux développés en JAVA pour l'outil VNC. Cette intégration permet de se connecter au partage d'application au travers de pages Web.

3.2.3 Outils de partage de documents

La fonctionnalité de partage de document permet à un ensemble d'utilisateurs d'accéder, de modifier voire d'annoter un document mis en commun au sein du groupe synchrone. Trois classes d'outils de partage ont été proposées selon le type de document utilisé et manipulé, type de complexité croissante :

- les éditeurs multiutilisateurs, qui manipulent des informations textuelles
- les tableaux blancs partagés, qui traitent des informations graphiques et des images fixes
- les navigateurs collaboratifs, qui prennent en compte des documents Web pouvant contenir tout type de média textuel, des graphiques, des images fixes et des flux multimédias

3.2.3.1 Editeurs multiutilisateurs

Les éditeurs multiutilisateurs sont les premiers types d'outils de partage, le document étant de type texte. Un groupe d'utilisateur manipule et modifie de façon interactive un même texte. L'éditeur gère les accès concurrents pour garantir la consistance du fichier partagé.

Un exemple d'éditeur synchrone est le logiciel DistEdit qui permet de partager un fichier entre plusieurs utilisateurs et de maintenir la vue de ce fichier consistante chez chacun d'eux.

Les utilisateurs peuvent joindre ou quitter la session d'édition à tout moment. Un seul utilisateur peut modifier le fichier à la fois. Les autres ne sont que de simples observateurs des modifications.

3.2.3.2 Tableau blanc partagé

Un tableau blanc partagé reproduit le comportement d'un tableau blanc classique. Son but est ainsi de permettre à un groupe d'acteurs de dessiner ou d'annoter des images dans un espace partagé de l'écran, de manière synchrone, les acteurs étant physiquement distants, et interagissant au travers de l'ordinateur.

L'interface utilisateur de cette application se compose d'une fenêtre graphique pour chaque membre. Un membre autorisé du groupe affiche ses transparents. Ce membre peut annoter les transparents affichés. Selon les possibilités de ce tableau, les autres membres du groupe peuvent éventuellement annoter les transparents qui s'affichent.

Cet outil est très utile dans le cas de télé-séminaires ou de télé-réunions. Les supports de la réunion sont visualisés, discutés et annotés au travers du tableau blanc partagé.

Tableau blanc multiutilisateur de l'environnement Platine

Le tableau blanc de l'environnement prototype Platine est entièrement développé en JAVA (cf. Figure 1.10). Plusieurs utilisateurs peuvent travailler ensemble au travers de cet outil. Un membre autorisé du groupe affiche des transparents au format GIF ou JPEG. Il peut ajouter des annotations sur le transparent affiché. Les autres membres du groupe, s'ils en ont l'autorisation, peuvent aussi ajouter d'autres annotations qui sont visibles par l'ensemble des participants connectés. Les annotations sont échangées sous forme d'événements graphiques qui nécessitent peu de bande passante pour leur transmission.

3.2.3.3 Co-navigation

L'idée de la co-navigation est une extension de la navigation à un groupe d'utilisateurs, pour que les membres du groupe naviguent en commun sur des documents Web. La manière classique, rendue disponible actuellement par les navigateurs traditionnels, conduit à une interaction inexistante, voire très faible, des utilisateurs entre eux. Chacun accède à un document de façon indépendante de celle des autres. La co-navigation propose un concept de navigation synchrone dans lequel un membre privilégié d'un groupe pilote l'accès à un document Web et force une présentation identique de ce dernier chez tous les autres membres du groupe. L'attribut de membre privilégié peut bien entendu changer de personne dans le temps.

Navigateur collaboratif Colab

Le navigateur collaboratif Colab, développé par G. Hoyos, est un environnement prototype basé sur le Web.

Le principal intérêt de cet environnement prototype est de proposer une extension paramétrable et plus souple, de la façon dont un groupe d'utilisateurs peut naviguer en commun sur des pages Web. En effet, le mode de co-navigation, implanté par les

environnements tels que WebEx, force une présentation identique des pages Web chez tous les membres du groupe. Cette présentation est pilotée par un membre privilégié.

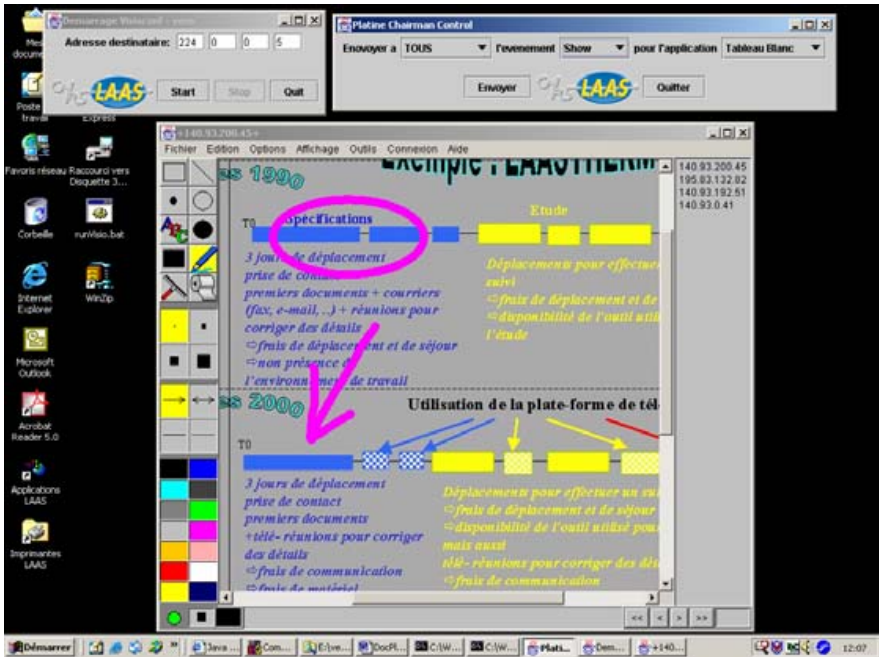


Figure 1.10. Tableau blanc partagé

La navigation collaborative, implantée par Colab, assouplit le mode de co-navigation. Elle propose ainsi une palette de modes de navigation de groupe possibles, depuis le fortement contraint jusqu'au faiblement contraint en passant par un ensemble de modes intermédiaires définis dynamiquement en fonction des pages visitées, des méta-données associées, du groupe, et de leur évolution conjointe dans le temps.

L'architecture de Colab se compose (i) d'un serveur Web qui joue le rôle de Proxy et (ii) d'un moteur de collaboration. Le serveur Web Proxy se charge de synchroniser l'activité de navigation collaborative. Le moteur de collaboration définit dynamiquement les droits d'accès aux ressources Web, selon des politiques définies sous forme de règles.

Les clients de ce navigateur collaboratif sont des applets JAVA qui sont synchronisées par le serveur Web pour visualiser les pages Web chez chacun des participants.

3.2.3.4 Tendance d'évolution du partage de documents

La fonctionnalité de partage de documents évolue technologiquement (cf. Figure 1.11). On passe de documents texte manipulés dans les années 90 à des partages d'objets graphiques et d'images vers 1995. Par la suite, les documents partagés deviennent des pages Web au format HTML, puis sont remplacés par des documents XML. Actuellement, grâce aux outils de conavigation, le partage se fait sur des objets plus complexes, tels que des documents multimédias interactifs au format SMILE et des scènes au format VRML. Les co-navigateurs qui manipulent des scènes VRML forment par ailleurs une classe spécifique. Ils sont appelés Serveurs Multi-utilisateurs (ou Multi-User servers (MUS)). Un exemple est le MUS Cortona,

développé par la société Parallel Graphics, qui autorise un groupe d'utilisateurs à visualiser et à interagir sur une scène VRML partagée.

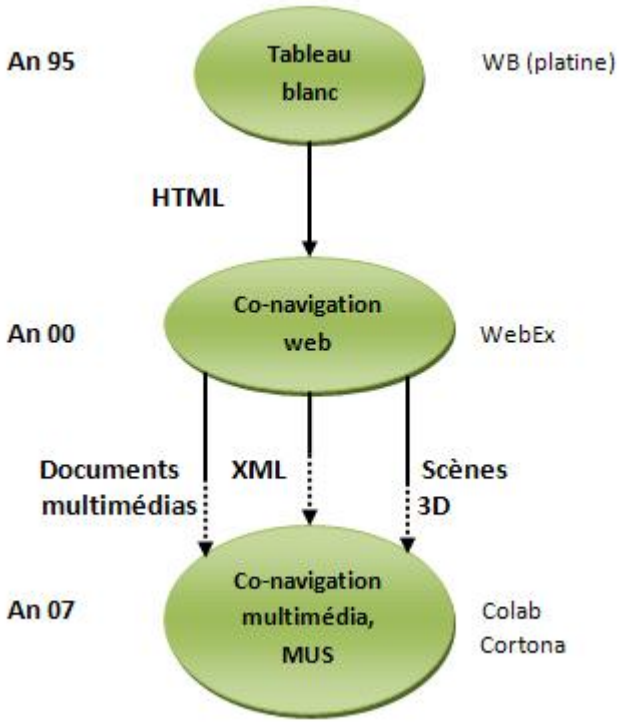


Figure 1.11. Evolution du partage de documents

3.2.4 Plates-formes de collecticiel synchrone

Les outils collecticiels synchrones sont rarement utilisés directement tels quels. La plupart du temps, ils se trouvent rassemblés dans des plates-formes collecticielles dont le premier but est d'offrir une interface unique pour leur accès.

Le deuxième rôle des plateformes collecticielles est la mise en oeuvre d'un ensemble de services qui coordonnent les outils collecticiels entre eux, et qui gèrent les utilisateurs et les ressources dont ils ont besoin pour accomplir leur travail de groupe.

On peut ainsi citer comme plates-formes commerciales WebEx ou Microsoft Messenger, et comme plates-formes prototypes les environnements CAliF Multimédia et Platine.

Centre de conférence Active Touch WebEx

La plate-forme WebEx Communications offre des services de réunion en ligne, de conférence Web, de téléconférence et de vidéoconférence. Ces services intégrés de travail collaboratif répondent à divers besoins de réunions multimédias. Ils sont accessibles à l'aide d'un navigateur et d'un téléphone. S'appuyant sur la technologie MediaTone, WebEx a déployé un réseau distribué dans le monde entier conçu pour faciliter les réunions interactives en ligne, les sessions de support client, les programmes de formation et les séminaires Web.

Toute conférence WebEx tourne dans la fenêtre d'un navigateur classique et ne nécessite aucune installation, configuration ou mise à jour logicielle. Les nouvelles fonctionnalités sont

ajoutées automatiquement au navigateur lorsque des nouvelles versions apparaissent. Au niveau sécurité, WebEx fonctionne au travers des pares-feux, offre des réunions privées non listées, des protections par mot de passe et des possibilités de cryptage des données sensibles. Le système WebEx supporte plusieurs milliers de participants et plusieurs centaines de réunions concurrentes.

Les principales caractéristiques de cette plate-forme sont résumées dans le tableau 1.

Fonctionnalité	**Description**
Vidéoconférence	La vidéoconférence depuis une caméra de bureau est activée sans autre installation logicielle nécessaire. Les participants peuvent voir les vidéos des autres participants dans leur navigateur.
Partage de présentation	Un participant à une réunion peut spontanément partager une présentation sans changer cette dernière sur le serveur WebEx, où la sécurité du fichier transféré pourrait être compromise. La visualisation peut se faire en mode plein écran.
Partage de document	N'importe quel document ou graphique peut être partagé et étendu par des annotations. Le propriétaire du document peut autoriser ou interdire les copies locales chez les participants.
Partage d'application (ou du bureau)	Des applications peuvent être exécutées et partagées pour des démonstrations ou des apprentissages interactifs.
Contrôle des applications partagées (ou du bureau)	Le présentateur peut transmettre le contrôle des applications partagées à n'importe quel autre participant de conférence.
Co-navigation par Web	Toute information basée sur le Web peut être partagée. Un participant peut naviguer sur le Web et synchroniser les navigateurs des autres participants. La co-navigation est pleinement interactive et son contrôle peut être passé à n'importe que participant, permettant ainsi de remplir des formulaires Web ensemble et d'annoter des pages Web.
Vote	Le présentateur peut solliciter des retours des participants en ligne.

Tableau1.1. Caractéristiques de WebEx

Conclusion

Après avoir représenté les aspects essentiels et les concepts fondamentaux du travail collaboratif, nous pouvons conclure que l'évolution des grandes classes d'outils collecticiels nous montre que ces outils convergent tous vers les technologies basées sur le Web. En effet, leur conception se fait de plus en plus selon une approche "composant Web". Ils deviennent accessibles soit directement à travers des navigateurs classiques, soit à travers des "plug-ins" qui s'exécutent aussi à travers des navigateurs classiques.

Le prochain chapitre va traiter un point pertinent dans le domaine du travail collaboratif qui est les services Web là où nous pouvons voir le passage de ces derniers vers la technologie des services Grid.

Chapitre II

Des services Web aux services Grid

Introduction

Ce chapitre a pour objectif de présenter le passage des services Web aux services Grid. Dans un premier temps, nous introduisons ce que nous entendons par architecture orientée service et décrivons les caractéristiques qui les rendent attrayantes pour les concepteurs d'architectures distribuées. Ainsi que les principaux standards des Services Web que sont SOAP, WSDL et UDDI. Nous définissons ensuite les apports des services Grid par rapport aux services Web en décrivant leurs principales caractéristiques. Nous pouvons voir cela à travers l'apparence des technologies OGSA OGSI et WSRF.

1. Services Web

Apparus dès la fin des années 1990, à l'aube du 21ème siècle, les Services Web ont provoqué une forte évolution dans le monde de l'informatique distribuée et un bouleversement majeur dans la façon de concevoir des architectures. Un des intérêts de l'informatique distribuée est de faciliter l'interconnexion entre applications distantes, indépendamment des plates-formes et des langages utilisés. Les trois derniers standards CORBA/IIOP (*Common Object Request Broker Architecture / Internet Inter-ORB Protocol*), DCOM (*Distributed Component Object Model*) et RMI (*Remote Method Invocation*) ont été créés dans ce but. Cependant, ces modèles, de part leur complexité, leur aspect fortement couplé sont en fait, incapables de passer à grande échelle. Ils restent donc, le plus souvent, confinés à l'intérieur des entreprises.

Créées, à la base, pour permettre les échanges commerciaux sur Internet, les technologies des Services Web sont, de par leur nature, très ouvertes. Grâce à des standards d'interopérabilité, ces technologies uniformisent la présentation des services offerts par une entreprise et rendent l'accès transparent à tout type de plate-forme. Le développement d'Internet, la démocratisation du haut débit, la structuration des données via XML et la recherche d'interopérabilité sont autant de facteurs qui ont favorisés l'essor des Services Web. Ces composants sont faiblement couplés permettant ainsi la réalisation d'application dynamique, flexible et évolutive à grande échelle. Les entreprises publiaient déjà de l'information via des sites web, utilisaient la messagerie et faisaient du commerce électronique. Elles l'utilisent maintenant pour leurs applications métiers (e-commerce, gestion de multinationale, etc.). La première société à avoir introduit un concept de services distribués, proche de ce qui existe aujourd'hui, est Hewlett-Packard avec son produit e-Speak, et ce dès 1999. La suite fut une grande course entre les principaux acteurs du marché qui ont progressivement, par souci d'interopérabilité, adopté les principaux standards des Services Web que sont SOAP [76], WSDL [81] et UDDI [50].

Cette nouvelle technologie permet donc la création de nouvelles applications aux frontières non délimitées et pousse aujourd'hui les développeurs à privilégier la réutilisation de services « sur étagère », plutôt que de procéder à des développements spécifiques pour chaque projet. L'utilisation de services sur étagère permet aux industriels de se concentrer sur leur domaine de compétence, tout en s'épargnant de l'effort de réaliser des fonctionnalités déjà développées dans d'autres secteurs. Cette tendance à la réutilisation, et surtout l'interconnexion croissante des systèmes, ont ainsi favorisé l'émergence de nouveaux concepts d'architectures tels que les AOS (Architectures Orientées Services ou *Service Oriented Architecture*), qui permettent l'interopérabilité de systèmes même lorsqu'ils sont développés par des organisations indépendantes.

Basés sur les protocoles XML, les Services Web (SW) constituent la technologie de base pour le développement d'Architectures Orientées Services (AOS) [46]. Ces architectures permettent de mettre en place des applications faiblement couplées avec un fort degré de configuration dynamique. Elles se basent sur la notion de relations de "service" formalisée par un contrat qui unit le client et le prestataire de services. Ce contrat est le point charnière de ce type d'applications.

1.1 Les Architectures Orientées Services (AOS)

L'architecture orientée services est le terme utilisé pour désigner un modèle d'architecture pour l'exécution d'applications logicielles réparties. Les deux derniers modèles (CORBA et DCOM) relèvent de l'architecture par composants logiciels répartis plutôt que de l'architecture orientée services, et le terme « service » est généralement absent de leur terminologie (sauf, par exemple, dans CORBA où l'on parle de services CORBA à propos de fonctions offertes par la plate-forme middleware aux composants applicatifs). De plus, un des avantages des AOS réside dans le fait que les applications réparties n'ont plus besoin d'un système de middleware réparti **commun** pour communiquer, mais seulement des protocoles et des technologies de communications intéropérables sur Internet. Ces protocoles et technologies seront d'ailleurs présentés par la suite.

Construire une architecture orientée services signifie donc d'abord concevoir une architecture en réseau de relations de services, entre applications réparties. La description d'une relation de services est formalisée par un contrat de services. Ce contrat décrit les engagements réciproques du prestataire et du client du service.

L'émergence des technologies de Services Web est censée apporter un niveau d'interopérabilité très élevé avec un degré de couplage très faible et donc d'établir les fondations des architectures orientées services à haut niveau de configuration dynamique. Une architecture d'applications réparties faiblement couplées est constituée d'un ensemble décentralisé d'applications réparties autonomes (Services Web), lesquelles interagissent sur la base de protocoles de communications, et sont mises en œuvre à l'aide de technologies ouvertes et non intrusives.

1.1.1 Qu'est-ce qu'un service ?

La notion de service fait remonter d'un cran le niveau d'abstraction auquel les développeurs d'architectures à composants ont été habitués. Traditionnellement, le composant est représenté par une boîte constituée de plusieurs facettes et/ou réceptacles (des entrées et des sorties). Lorsqu'on parle de service, celui-ci est caractérisé par un point d'accès. Une interface, le contrat de service (le document WSDL) précise les messages d'entrée que peut recevoir ce point d'entrée pour réaliser la prestation. Une architecture orientée services peut donc être représentée par une interconnexion de multiples points d'accès.

En résumé, la notion de service est le produit d'une démarche d'abstraction par rapport au logiciel qui l'implémente, au processus qui l'exécute et au port qui le localise. Cependant, le service reste un objet très concret et technique. La réalisation d'un service passe par des messages échangés, des transitions d'état et des actions que l'application cliente **suppose** assurés de la part de l'application prestataire du service. Les caractéristiques fonctionnelles, opérationnelles et d'interface d'un service sont consignées dans un contrat. Un Service Web n'est donc rien d'autre que la réalisation supposée de la prestation définie dans le contrat de service.

1.1.2 Le contrat de service

Le contrat de service du modèle des AOS s'inspire directement du modèle des contrats professionnels de service. Il s'agit d'un document qui développe l'ensemble des points permettant de décrire et donc de définir la relation de service.

Dans le monde des relations professionnelles, un contrat de service est un document comportant plusieurs parties et dont les éléments terminaux sont généralement appelés *articles* et *clauses*. Le contrat de service contient des articles et des clauses consacrés à des sujets tels que: l'identification des parties contractantes, la description de la prestation objet du contrat, les modalités d'exécution, les modalités d'interaction entre les parties (il est aussi courant que le contrat de service décrive les actions à entreprendre en cas de défaillance d'un des contractants ou d'impossibilité de réalisation de la prestation).

Dans le monde des AOS, les éléments du contrat de service sont organisés en six thèmes majeurs (cf. Figure 2.1): l'identification des parties, la description des fonctions du service, la description de l'interface du service, la description de la qualité de service, la description du cycle de vie du service et du contrat, la description des termes de l'échange.

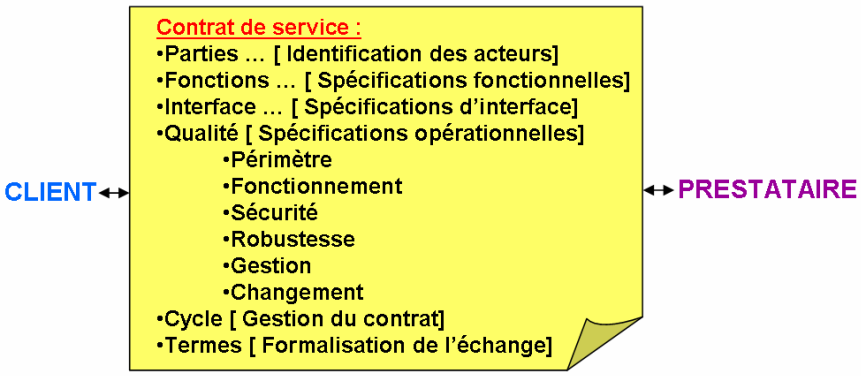


Figure 2.1. Le contrat de service

Parmi ces termes, la qualité de service est un élément majeur puisqu'elle est:

- Un terrain de compétition entre prestataires qui offrent des services exhibant le même modèle fonctionnel et la même interface.
- L'objet d'observation et de suivi de la prestation de la part des clients et des tiers, ainsi que de notation des prestataires par des tiers indépendants jouant le rôle d'organismes d'évaluation.

1.1.3 L'agrégation et la dissémination de service

Une architecture orientée services peut être conçue par une approche incrémentale, résultat de la combinaison de deux démarches de base, présentées ci-dessous, qui sont :

- **L'agrégation de services:** La notion d'agrégation fait monter le niveau de découplage entre service et implémentation. La figure 2.2 représente le service agrégeant de l'agence de voyage. Ce service est issu de la composition (ou de l'orchestration) de services atomiques (ici, les services de réservation d'avions, d'hôtels et de voitures). L'implémentation d'un service par

agrégation d'autres services est inconnue du client qui utilise le service agrégeant. En effet, celui-ci ne connaît pas et n'a pas besoin de connaître l'éventuelle complexité de l'implémentation du service en ce qui concerne l'agrégation d'autres services. Du point de vue du client, le service agrégeant est un service atomique décrit de façon usuelle par un contrat. Un service atomique peut être implémenté par agrégation récursive de services atomiques. En guise de remarque, l'agrégation de services se fait par la mise en œuvre de processus métier à l'aide d'outils tel que BPEL4WS [4]. Il s'agit ici d'établir la coopération entre les services agrégés, à savoir : la division du travail et l'échange d'informations nécessaires pour que les activités de ces services agrégés contribuent efficacement à la réalisation du service agrégeant. Il est également nécessaire de coordonner les services agrégés en synchronisant les étapes de leurs activités. L'implémentation du service agrégeant organise la coopération et coordonne la réalisation des services agrégés, en faisant éventuellement appel à des services spécialisés de coordination comme dans le cas de la gestion des transactions avec des protocoles tels que WS-Coordination [13] et WS-Transaction [83].

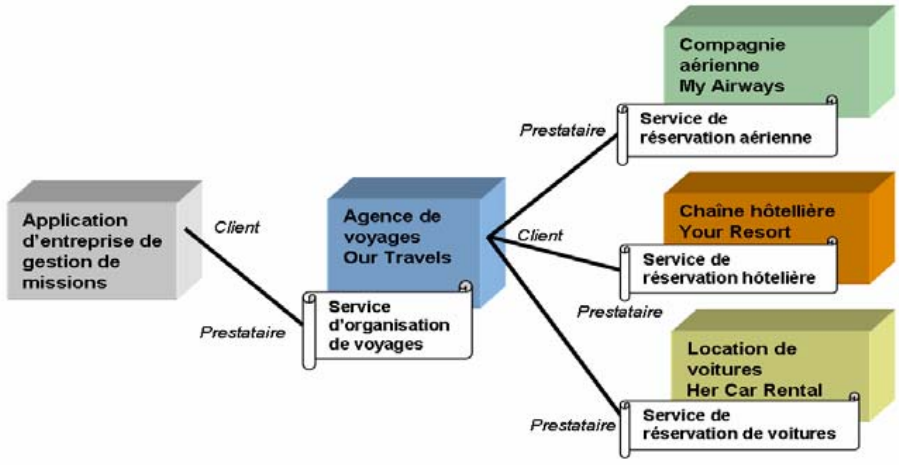


Figure 2.2. Agrégation de services

La dissémination de services: La dissémination de services permet d'effectuer la décentralisation des données. Comme on peut le voir sur la figure 2.3, plusieurs applications sont prestataires du même service sur des données différentes. Une entreprise peut donc implémenter un ensemble de services modulaires à partir d'applications différentes. Le point essentiel est que le client d'un des services issus d'une démarche de dissémination bénéficie de la modularité du service sans que la question de la modularité de l'implémentation ne soit même posée. En informatique, on parle traditionnellement, à ce propos, de « boîte noire » et d'information « hiding ». Dans l'architecture orientée services, le découplage entre interface et implémentation est poussé aux extrêmes limites : un service est tout simplement un contrat, et une occurrence d'un service en exécution est tout simplement un port (dont l'adresse peut être connue dynamiquement).

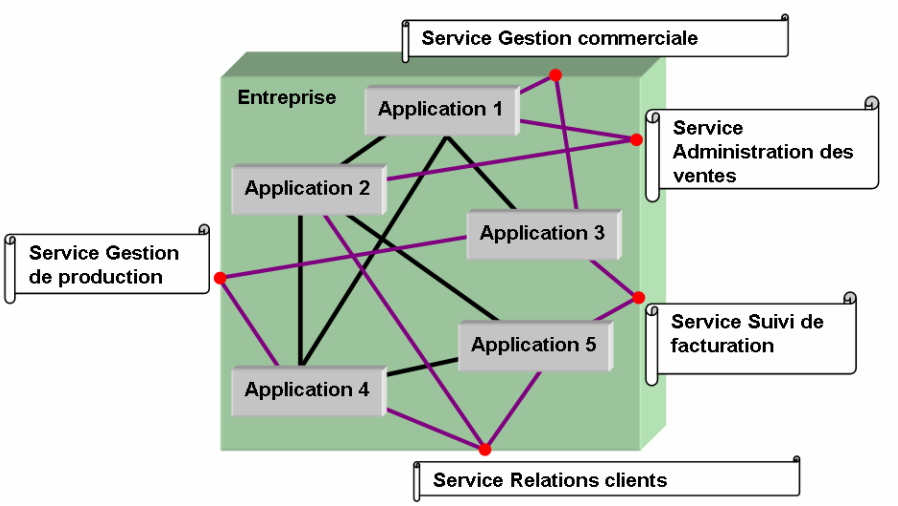


Figure 2.3. Dissémination de services

L'agrégation et la dissémination des services constituent les approches de base de conception et de mise en œuvre d'architectures orientées services. Les architectures orientées services complexes qui vont se déployer dans les années à venir seront sans doute le résultat de la combinaison de ces deux approches. Il est important d'insister sur le caractère incrémental et étalé dans le temps de telles démarches. Le modèle de l'architecture orientée services se prête parfaitement à l'urbanisation des systèmes d'information au sens large (intra et interentreprises). Il permet ainsi de planifier dans le temps et dans l'espace la réutilisation et la fin de vie des applications existantes, la refonte des applications, la mise en œuvre de nouvelles applications à valeur ajoutée, tout en garantissant leur interopérabilité. En effet, le concept de l'urbanisation de l'habitat (organisation des villes, du territoire) a été réutilisé en informatique pour formaliser ou modéliser l'agencement du système d'information (SI) de l'entreprise. On utilise le terme d'urbanisation pour mettre l'accent sur le travail progressif nécessaire pour faire évoluer le système d'information vers une cible correctement urbanisée. Les évolutions des stratégies des entreprises (regroupement et fusion, acquisition, diversification des offres commerciales, e-commerce, gestion de la relation client, nouveau mode ou canal de distribution, partenariat, réorganisation, externalisation, redéploiement des fonctions de back et front office, etc.) impliquent des changements structurels importants et accroissent l'interdépendance (dépendance mutualisée) et l'imbrication des applications informatiques avec le risque de renforcer l'effet « plat de spaghettis » du système d'information. Cette complexité croissante a des conséquences sur les coûts, les durées et les risques des projets d'évolution des SI. Les architectures orientées services ont été créées dans le but de faciliter le travail d'urbanisation des grands systèmes informatiques.

1.1.4 Des architectures dynamiques

Cet aspect dynamique de la configuration de l'architecture est au cœur même du concept d'architecture orientée services (ce qui n'empêche pas par ailleurs de mettre en œuvre des architectures orientées services totalement statiques).

Dans une architecture dynamique, les services qui la composent, les applications prestataires qui interviennent, ainsi qu'un certain nombre de propriétés opérationnelles des

prestations de services ne sont pas définies avant sa mise en place, mais sont composés, configurés, établis, voire négociés, au moment de l'exécution. Ce processus peut être itératif : il est possible de reconfigurer une architecture dynamique à la volée lors de son fonctionnement normal, ou bien à l'occasion d'un dysfonctionnement.

Le degré de couplage des architectures réparties évolue sur un continuum qui va du très fortement couplé au très faiblement couplé. Les architectures orientées services permettent de modéliser des architectures à n'importe quel degré de couplage (cf. Figure 2.4).

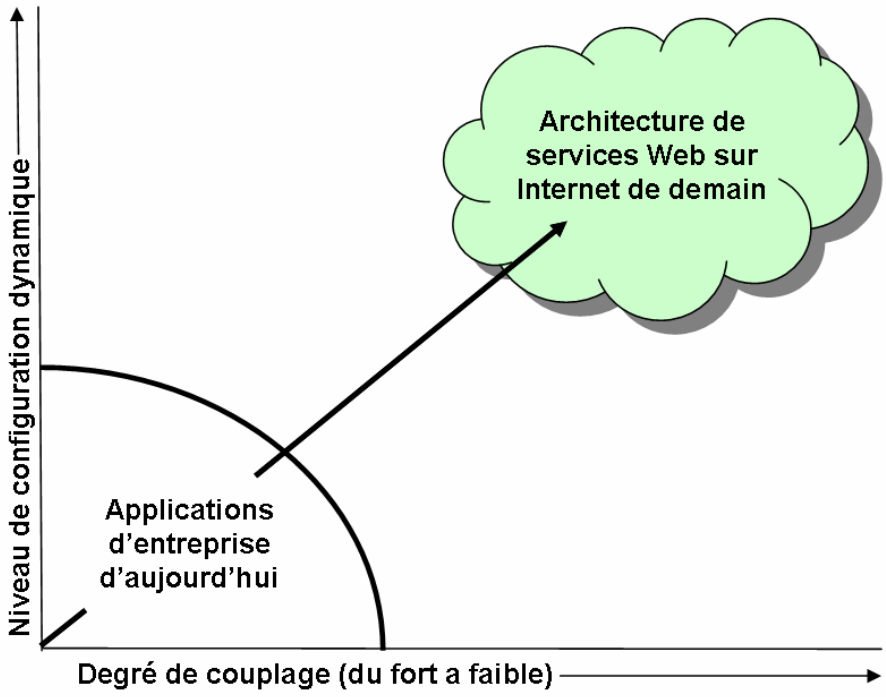


Figure 2.4. Degré de couplage et niveau de configuration dynamique

Les architectures dynamiques permettent aux applications qui les constituent de choisir dynamiquement (à l'exécution)

- les points d'accès des prestataires ;

- les techniques de liaison (implémentations des interfaces) avec les prestataires

- les prestataires des services ;

- les services (contrats) qu'elles utilisent en tant que clientes.

Avec les technologies des services Web disponibles actuellement, on peut notamment établir des architectures dans lesquelles les applications participantes peuvent choisir dynamiquement les services qu'elles consomment, les prestataires de ces services, les ports d'accès de ces prestataires.

1.1.5 Des architectures « boîtes noires »

Dans une AOS (cf. Figure 2.5), les relations de service qui lient les applications participantes sont régies par des contrats de service. Les prestations, issues des contrats, doivent

impérativement être décrites au niveau fonctionnel, et il est incorrect d'inclure les modèles d'implémentation des applications prestataires dans les contrats de service. L'implémentation de l'application prestataire est donc une boîte noire par rapport au contrat de service, sauf pour ce qui touche l'implémentation de la communication.

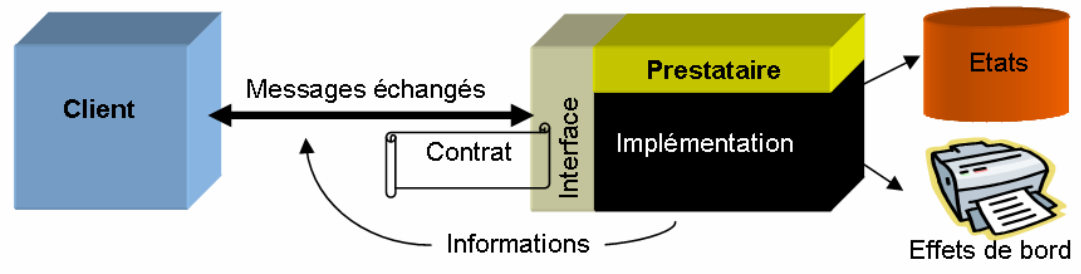


Figure 2.5. Architecture boîte noire avec une interface transparente

Par ailleurs, les fonctions publiées dans le contrat de service ne sont, peut être, qu'une partie des fonctions mises en œuvre par l'application prestataire. D'autres fonctions peuvent être publiées dans d'autres contrats pour lesquels l'application joue également le rôle de prestataire. Dans tous les cas, la description fonctionnelle complète de l'application, en termes d'objectifs, d'actions, d'informations et de règles, à savoir son coeur métier, constitue également une boîte noire pour les clients et les autres prestataires de services. Plus précisément, certaines parties du modèle fonctionnel sont publiées dans les contrats de service (avec différents niveaux de visibilité), alors que d'autres parties restent cachées aux clients et aux autres prestataires.

1.2 Présentation des Services Web

Les Services Web sont, à l'heure actuelle, le seul moyen de mettre en place des architectures orientées services. Dans cette partie, nous allons nous échapper un peu de l'aspect conceptuel des architectures orientées services pour nous attarder sur l'aspect fonctionnel, technique, des Services Web. Nous étudierons tout d'abord les différents protocoles de base permettant de mettre en place une architecture à base de Service Web. Nous verrons enfin les outils existants permettant d'implémenter facilement un Service Web.

1.2.1 Les protocoles de base des Services Web

Les concepteurs de la pile des technologies de Services Web ont commencé à proprement parler des protocoles de base par WSDL et SOAP. Ces protocoles imposent un format de message XML. WSDL (*Web Services Description Language*) est le langage de description des Services Web, même s'il n'est pas formellement imposé par l'architecture de référence du W3C. On peut cependant considérer aujourd'hui qu'une description WSDL est nécessaire pour qu'une application puisse revendiquer la qualification de service Web. SOAP (*Simple Object Access Protocol*) est, quant à lui, le protocole standard d'interaction, d'échange d'informations entre un client et un prestataire.

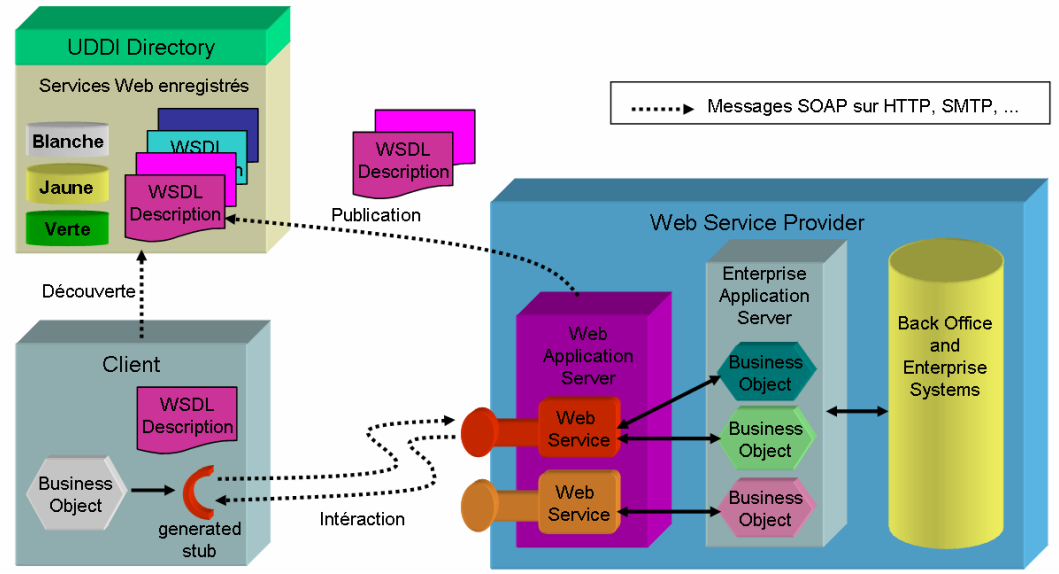


Figure 2.6. Les protocoles de base des Services Web

Les prestataires des Services Web, leurs interfaces et leurs points d'accès, peuvent être enregistrés, découverts et localisés via des technologies d'annuaire comme UDDI (*Universal Description, Discovery and Integration of Web Services*). Autant un standard ouvert (nonpropriétaire) sur les annuaires de services semble indispensable, surtout pour la mise en œuvre d'architectures dynamiques, autant la technologie UDDI, qui est clairement une technologie de Services Web, n'est pas encore formellement considérée aujourd'hui comme le standard des annuaires.

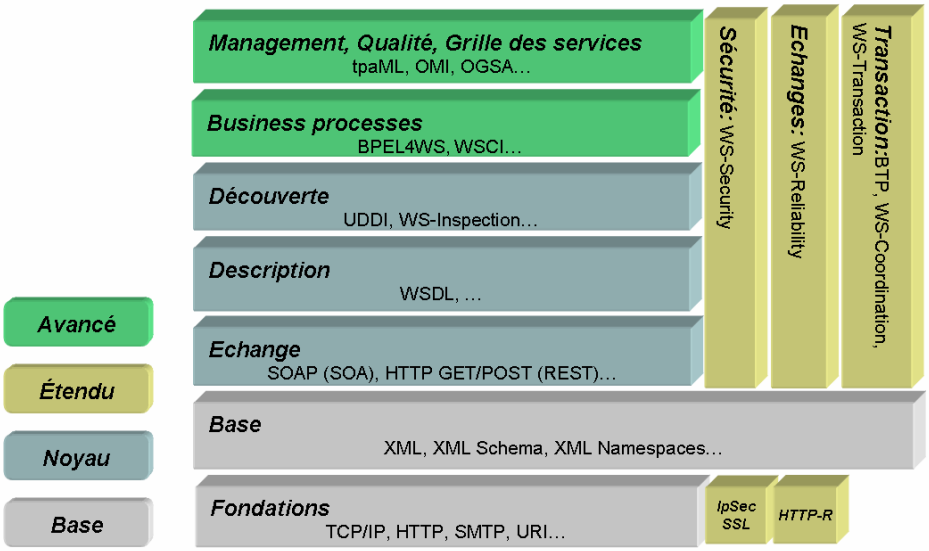


Figure 2.7. La pile des protocoles WS

WSDL, SOAP et UDDI (cf. Figure 2.6) constituent l'ensemble des technologies clés de Services Web, sur lesquelles d'autres technologies plus proches de la problématique applicative peuvent être spécifiées et mises en œuvre. La figure 2.7 présente de façon succincte les différents protocoles à base de Services Web qui peuvent se greffer autour des protocoles de base.

Dans la partie qui suit, nous présenterons en détail certaines caractéristiques des protocoles de base.

1.2.1.1 XML

XML (*Extensible Markup Language*) est un langage de balisage extensible [73] qui a été mis au point par le XML Working Group sous l'égide du *World Wide Web Consortium* (W3C). Les spécifications XML 1.0 sont reconnues comme recommandations par le W3C, ce qui en fait un langage reconnu. XML est un standard qui sert de base pour créer des langages balisés spécialisés; c'est un « méta langage ». Il est suffisamment général pour que les langages basés sur XML, appelés aussi dialectes XML, puissent être utilisés pour décrire toutes sortes de données et de textes. Il s'agit donc partiellement d'un format de données. Son objectif est, dans un échange entre systèmes informatiques, de transférer, en même temps, des données et leurs structures. Permettant de coder n'importe quel type de donnée, depuis l'échange EDI (*Electronic Data Interchange ou Echange de Données Informatisées*) [77] jusqu'aux documents les plus complexes, son potentiel est de devenir le standard universel et multilingue d'échange d'informations.

XML Namespaces est une extension de la recommandation XML qui permet de créer des espaces de nommages. Les espaces de noms d'XML permettent de qualifier de manière unique des éléments et des attributs. On sait alors à quel domaine de définition se rapporte un objet et comment il doit être interprété, selon sa spécification. Différencier des espaces de noms permet de faire coopérer, dans un même document, des objets ayant le même nom, mais une signification différente, souvent liée à un modèle de contenu différent. Cette spécification est une avancée importante car, à partir du moment où beaucoup de formats s'expriment selon XML, les risques de "collision de noms" deviennent plus importants et cette spécification prend alors toute son importance.

1.2.1.2 WSDL

WSDL [82] est l'outil pivot de la technologie des Services Web car il permet véritablement de donner une description d'un Service Web indépendante de sa technologie d'implémentation. Les traits principaux du langage sont présentés via l'exemple d'un des services Web les plus populaires : l'accès programmatique par SOAP au moteur de recherche Google.

Dans la mise en place des architectures de Services Web aujourd'hui, la fonction de contrat de service est portée par le document WSDL. Le choix d'un format universel et extensible de structuration de documents basé sur XML s'impose, pour satisfaire les deux contraintes principales qui pèsent sur le contrat de service :

- Le contrat de service doit être, en même temps, lisible par des acteurs humains et exploitable (généré, interprété, agrégé, décomposé, indexé, mémorisé, …etc.) par des agents logiciels.

- Le contrat de service doit être tout aussi facilement extensible, c'est-à-dire adaptable à l'évolution des services, des architectures et des technologies, et capable d'accueillir les nouveaux formalismes propres à de nouveaux types d'engagements.

Actuellement, le document WSDL ne permet de définir que l'implémentation de l'interface, à savoir : les styles d'échange, les formats des messages, les conventions de codage, les protocoles de transport, les ports de réception.

Un document WSDL définit une suite de descriptions de composants (cf. Figure 2.8). Le fichier WSDL est principalement composé de 6 éléments, avec d'autres éléments optionnels:

- **definitions:** élément racine du document, il donne le nom du service, déclare les espaces de noms utilisés, et contient les éléments du service (Obligatoire).
- **types:** décrit tous les types de données utilisés entre le client et le prestataire. WSDL n'est pas exclusivement lié à un système spécifique de typage, mais utilise par défaut la spécification XML Schema (Définition abstraite).
- **message:** décrit un message unique, que ce soit un message de requête seul ou un message de réponse seul. L'élément définit le nom du message et peut contenir (ou pas) des éléments « **part** », qui font référence aux paramètres du message ou aux valeurs retournées par le message (Définition abstraite).
- **portType:** combine plusieurs messages pour composer une opération. Chaque opération se réfère à un message en entrée et à des messages en sortie. (Définition abstraite).
- **binding:** décrit les spécifications concrètes concernant la manière dont le service sera implémenté: protocole de communication et format des données pour les opérations et messages définis par un type de port particulier. WSDL possède des extensions internes pour définir des services SOAP; de fait, les informations spécifiques à SOAP se retrouvent dans cet élément (Définition concrète).
- **service:** défini les adresses permettant d'invoquer le service donné, ce qui sert à regrouper un ensemble de ports reliés. La plupart du temps, c'est une URL invoquant un service SOAP (Définition concrète).

De nombreuses implémentations de Services Web se sont juste limitées à l'utilisation de la couche de transport SOAP pour faire interagir un client et un prestataire sans déclarer le service au travers d'un document WSDL. Cela est suffisant dans une situation où le service en question ne présente qu'un intérêt limité, propre aux deux acteurs qui contrôlent les noeuds de communication concernés. En revanche, si le Service Web est destiné à une utilisation dans un cadre plus élargi, il va rapidement devenir fastidieux pour le fournisseur de ce service de décrire et d'informer chaque consommateur potentiel des caractéristiques fonctionnelles et techniques de ce service. Il sera certainement préférable que ce fournisseur concentre ses ressources sur les aspects commerciaux et contractuels de son offre par exemple.

Ainsi, la spécification WSDL joue un rôle pivot dans une architecture de Services Web. C'est la présence d'un contrat WSDL qui permet d'affirmer que l'on met en œuvre un Service Web. Le seul usage de SOAP dans la communication entre applications réparties ne suffit pas pour qualifier ces applications de Services Web.

```
<definitions name="GoogleSearch" targetNamespace="urn:GoogleSearch"  xmlns:typens="urn:GoogleSearch"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"   xmlns="http://schemas.xmlsoap.org/wsdl/">
   <types><xsd:schema>…</xsd:schema></types>
   <message name="doGoogleSearch">
     <part name="key"          type="xsd:string"/>
     <part name="q"            type="xsd:string"/>
     <part name="start"        type="xsd:int"/>
     <part name="maxResults"   type="xsd:int"/>
     <part name="filter"       type="xsd:boolean"/>
     <part name="restrict"     type="xsd:string"/>
     <part name="safeSearch"   type="xsd:boolean"/>
     <part name="lr"           type="xsd:string"/>
     <part name="ie"           type="xsd:string"/>
     <part name="oe"           type="xsd:string"/>
   </message>
   <message name="doGoogleSearchResponse">
     <part name="return"       type="typens:GoogleSearchResult"/>
   </message>
   <portType name="GoogleSearchPort">
     <operation name="doGoogleSearch">
       <input message="typens:doGoogleSearch"/>
       <output message="typens:doGoogleSearchResponse"/>
     </operation>
   </portType>
   <binding name="GoogleSearchBinding" type="typens:GoogleSearchPort">
     <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
     <operation name="doGoogleSearch">
       <soap:operation soapAction="urn:GoogleSearchAction"/>
       <input>
         <soap:body use="encoded" namespace="urn:GoogleSearch"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
       </input>
       <output>
         <soap:body use="encoded" namespace="urn:GoogleSearch"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
       </output>
     </operation>
   </binding>
   <service name="GoogleSearchService">
     <port name="GoogleSearchPort" binding="typens:GoogleSearchBinding">
       <soap:address location="http://api.google.com/search/beta2"/>
     </port>
   </service>
</definitions>
```

Figure 2.8. Contrat WSDL de Google

1.2.1.3 Les Schémas XML

Un schéma XML définit les éléments possibles dans un document et les attributs associés à ces éléments avec leur type de données. Les schémas sont standardisés sous forme de recommandation du W3C (XML Schema 1.0) depuis mai 2001. Les schémas XML [83] ont un rôle essentiel dans la création des Services Web. Ils permettent de créer des types propres aux services. Les types de données créés sont soit des types simples, soit des types complexes.

Il existe une large variété de types simples intégrés à XML (*Built-in Simple Type*), dont les entiers, les réels, les chaînes de caractères et les dates sous des formes variées. De plus, les types simples intégrés peuvent être également spécialisés.

La figure 2.9 représente la hiérarchie des types définis dans le schéma des schémas XML (http://www.w3.org/2001/XMLSchema.xsd).

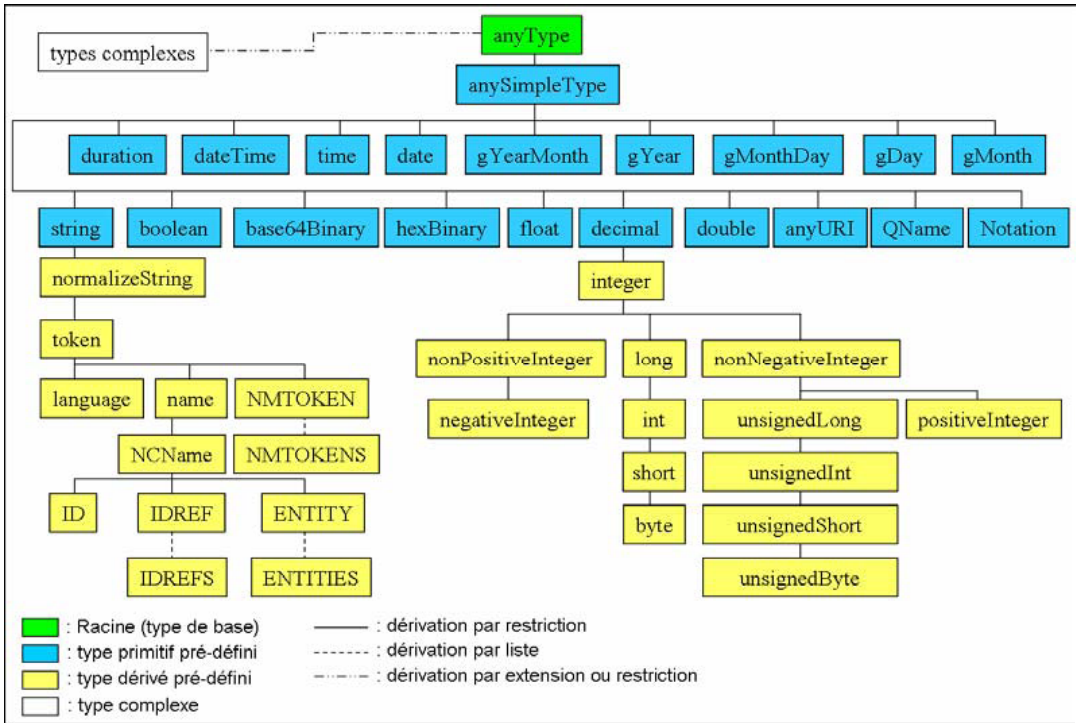


Figure 2.9. La hiérarchie des définitions de type d'XML Schema

Les types complexes sont des types composés d'éléments assemblés à l'aide de constructeurs: séquence (sequence), choix (choice) et tas (all). Ils permettent de définir la structure de documents complexes, de manière quasi similaire aux constructeurs du monde objet. Ils sont cependant, plus limités que ceux du monde objet et plus appropriés à la modélisation d'éléments pour les documents XML. La figure 2.10 présente ainsi le type complexe retourné par l'opération « *doGoogleSearch* » de Google. Ce type est constitué d'une séquence d'éléments simples et complexes.

```
<xsd:complexType name="GoogleSearchResult">
    <xsd:all>
        <xsd:element name="documentFiltering"          type="xsd:boolean"/>
        <xsd:element name="searchComments"             type="xsd:string"/>
        <xsd:element name="estimatedTotalResultsCount" type="xsd:int"/>
        <xsd:element name="estimateIsExact"            type="xsd:boolean"/>
        <xsd:element name="resultElements"             type="typens:ResultElementArray"/>
        <xsd:element name="searchQuery"                type="xsd:string"/>
        <xsd:element name="startIndex"                 type="xsd:int"/>
        <xsd:element name="endIndex"                   type="xsd:int"/>
        <xsd:element name="searchTips"                 type="xsd:string"/>
        <xsd:element name="directoryCategories"        type="typens:DirectoryCategoryArray"/>
        <xsd:element name="searchTime"                 type="xsd:double"/>
    </xsd:all>
</xsd:complexType>
```

Figure 2.10. XML Schema - Le type complexe GoogleSearchResult

Les schémas XML possèdent des instructions qui donnent une grande puissance d'expressivité:

- ***Spécialisation de types par restriction :*** Comme avec les sous-classes des langages orientés objet, il est possible de définir des sous-types par héritage. De manière classique, la spécialisation peut s'effectuer sur un type simple ou complexe par ajout de contraintes, appelées restrictions. Ces contraintes peuvent porter sur une propriété de type comme dans l'exemple présenté dans la figure 2.11. De manière générale, tous les types simples mais aussi les types complexes peuvent être réduits par des contraintes de restriction, appelées facettes (ex : facette « pattern » du type string dans la figure 2.11). Les facettes définissent les différents types de restrictions applicables sur un type de données.

```
<xsd:simpleType name=« typeCodePostalFR »>
   <xsd:restriction base=«xsd:string »>
      <xsd:pattern value=« \d{5} »>
   </xsd:restriction>
</xsd:restriction>
```

Figure 2.11. Spécialisation de type par restriction

- ***Dérivation de types par extension :*** L'ajout d'information à un type complexe est appelé une extension. Les types ainsi spécialisés sont marqués par une balise <complexContent>. La clause <xsd :extension> introduit cette catégorie de soustypage avec en attribut le nom du type de base.

En résumé, les schémas offrent une grande variété de types simples surchargeables et la possibilité de construire des types complexes extensibles. Les schémas apportent une définition des types de données de base riche et extensible, un peu comme l'objet. Sans schéma, XML reste un langage de représentation de documents textuels et n'est pas soustendu par un véritable modèle de données. Les schémas sont complexes et relativement difficiles à mettre en œuvre. Ils sont toute fois indispensables à la création des services Web et ont une part importante dans la définition du contrat WSDL. Les schémas XML sont spécifiés d'emblée comme le seul outil de définition de format XML dans les Services Web.

1.2.1.4 SOAP

SOAP [76], [77], [78] fournit un mécanisme qui permet d'échanger de l'information structurée et typée entre applications dans un environnement réparti et décentralisé. Il ne véhicule pas de modèle de programmation ou d'implémentation, mais fournit les outils nécessaires pour définir des modèles opérationnels d'échange (styles d'échange) aussi diversifiés que les systèmes de messagerie asynchrone et l'appel de procédure distante (RPC).

Le message SOAP est un document XML. Un message SOAP présente une structure normalisée (cf. Figure 2.12). Il est toujours constitué d'une enveloppe (SOAPENV: Enveloppe) munie d'une en-tête (SOAP-ENV:Header) optionnelle et d'un élément (SOAP-ENV:Body) obligatoire contenant le corps du message, suivis d'éventuels éléments applicatifs spécifiques.

La spécification considère explicitement que les en-têtes SOAP sont destinés à la mise en œuvre de couches supérieures et transversales de la technologie des services Web, comme la gestion des transactions, la gestion de la sécurité, etc.

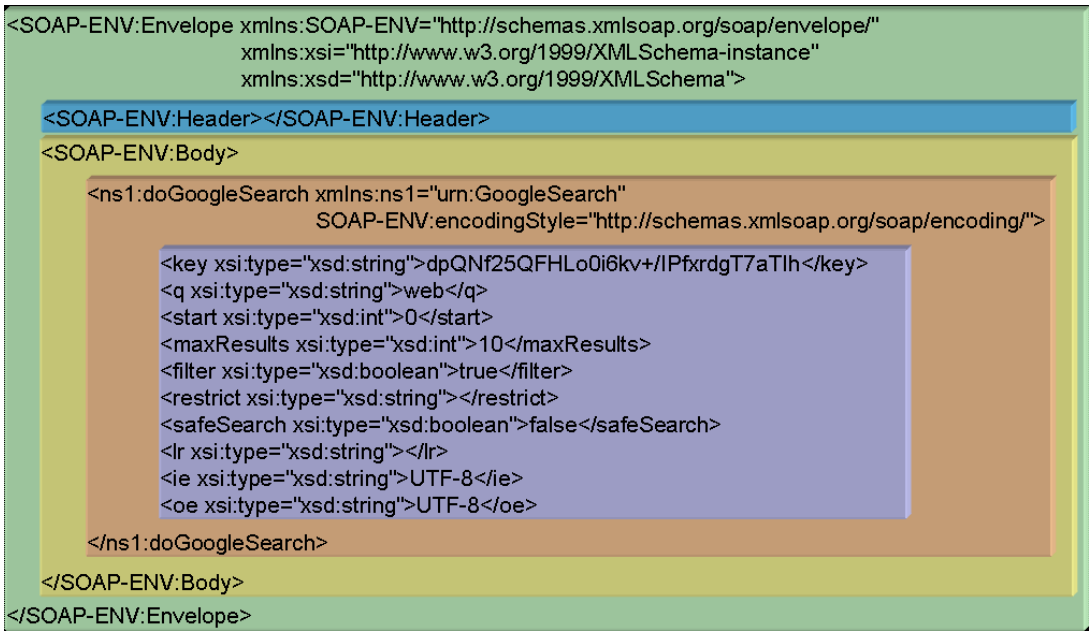


Figure 2.12. Requête SOAP de Google

1.2.1.5 UDDI

UDDI [50] est le support d'un système réparti d'annuaires répliqués qui permettent la publication et la découverte de services sur Internet. Un annuaire UDDI est accessible par l'intermédiaire du protocole SOAP. L'API UDDI est un service Web décrit au format WSDL qui permet d'accéder à un annuaire via l'utilisation du protocole SOAP. On peut également faire une analogie entre l'annuaire UDDI et le service de désignation de CORBA.

L'annuaire UDDI est accessible soit via un navigateur Web qui dialogue avec une application Web dédiée, interface spécifique à l'annuaire accédé, soit par programme, en utilisant l'API (*Application Programming Interface*) définie par la spécification.

L'API comporte deux groupes de fonctions :

- ***les fonctions de recherche (Inquiry API) :*** navigation, recherche et consultation des informations de l'annuaire;
- ***les fonctions de publication (Publishing API) :*** publication, création, modification ou suppression des informations de l'annuaire.

Un annuaire UDDI peut être public ou privé. De manière standard, la spécification prévoit qu'un annuaire est distribué sur plusieurs noeuds. Ces noeuds sont synchronisés au moyen du mécanisme de réplication interne. La réplication n'est pas obligatoire, notamment dans un cadre privé, mais est fortement recommandée pour des raisons évidentes de disponibilité. Cette fonctionnalité est, bien sûr, mise en œuvre par l'annuaire public UDDI, dont les implémentations des opérateurs (IBM, Microsoft, NTT Communications et SAP) se répliquent entre elles.

2. Du Web au Grid

Alors que le Web a montré toute la puissance d'une infrastructure pour le partage d'informations distribuées, Grid ouvre un nouveau champ de possibilités avec une infrastructure mutualisant des ressources informatiques réparties géographiquement. De ce fait, Grid apporte de nouvelles réponses à certaines limites intrinsèques du Web, telle que la coordination des ressources pour délivrer une qualité de service à la demande [23]. Bien qu'il existe des points de convergence entre ces deux architectures [59], ce qui distingue foncièrement Grid du Web sont des différences d'ordre topologique entre les noeuds de provision des ressources et les noeuds de consommation de ces ressources. Pour expliquer cette évolution, la figure 2.13 nous donne une représentation schématique des trois familles de topologies qui sont apparues depuis l'existence des réseaux informatiques. Les cercles indiquent les terminaisons côté utilisateur tandis que les carrés décrivent les terminaisons du côté des ressources informatiques. Le premier cas correspond à la période qui précéde l'ère internet ; les ressources sont fortement concentrées en un point et découplées des terminaisons utilisateurs. Le second cas correspond à la période qui suit l'apparition des architectures client-serveur et du Web ; les ressources commencent à être réparties mais restent fortement couplées physiquement aux terminaisons des utilisateurs. C'est la principale raison pour laquelle il n'est pas possible de coordonner les ressources réparties sur une infrastructure Web. Le troisième cas correspond à une utilisation généralisée de Grid avec des ressources réparties et virtualisées. Les terminaisons des utilisateurs ne sont plus couplées physiquement à des ressources informatiques mais elles sont en interaction avec des services à état, un concept clé de Grid dont nous détaillerons les principes dans ce chapitre.

HTTP ET DNS

Le fonctionnement du Web repose sur une infrastructure de serveurs interconnectés qui communiquent grâce aux protocoles Domain Name Service (DNS) et Hyper Text Transfer Protocol (HTTP). Le protocole DNS permet d'établir une correspondance entre une adresse Internet numérique de la forme 193.49.113.146 et un nom de domaine (ex. agora.lirmm.fr). Cette association réside de façon entièrement statique dans des tables de correspondance des serveurs DNS. Les modifications sur ces tables peuvent être effectuées à titre exceptionnel. Ce nom de domaine permet d'identifier les différents serveurs HTTP. Ces serveurs fournissent un service d'échange d'information. L'indication dans un navigateur de l'adresse complète http ://www.lirmm.fr déclenche automatiquement l'appel du protocole HTTP. L'inconvénient de cette architecture est son manque de fiabilité : un serveur HTTP qui s'écroule est autant d'information inaccessible.

Grid permet grace au service à état de virtualiser toute forme d'échange d'information et de minimiser les risques de pannes. De cette façon la fonction serveur HTTP est réalisée de façon logique par un service à état accessible par un identifiant, au lieu d'être matérialisé par un hôte (ie. une machine réelle ou virtuelle) avec une adresse TCP/IP comme c'est le cas pour le Web. Ce principe permet de déployer de façon simple et efficace des réseaux de services en interaction avec des règles globales pour une meilleure tolérance aux pannes et pour décentraliser les politiques de sécurité.

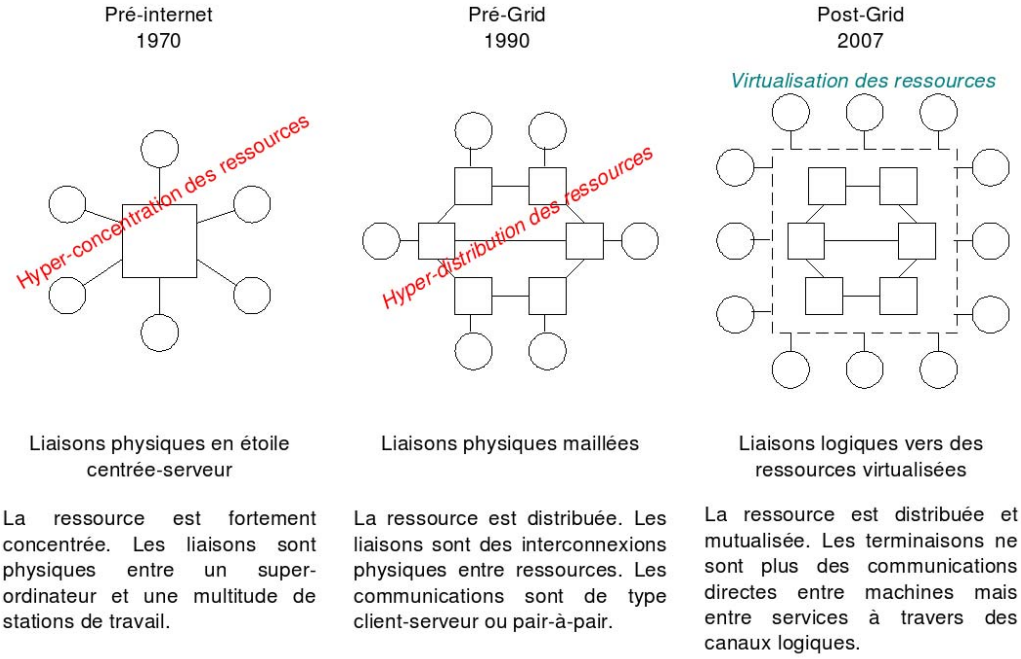


Figure 2.13. Trois familles de topologies.

2.1 Aspects normatifs

L'architecture Grid est principalement décrite par deux normes complémentaires.

– La norme OGSA a été adoptée par le Global Grid Forum (GGF) 10 lors de la conférence de Toronto en 2002. OGSA définit un modèle d'architecture orientée-services qui vise à virtualiser les ressources et à les restituer sous forme de services afin de pouvoir les assembler et les désassembler en fonction des besoins [24]. Le principe fondamental du modèle OGSA provient du processus de génération dynamique des services au moyen du couple générateur-instance de service. Ceci implique l'existence de mécanismes de réservation et de libération des ressources. L'autre point fort de cette norme concerne la sécurité. Ce sont les principaux aspects novateurs d'OGSA par rapport aux services Web.

– La norme WSRF est apparue en 2004 en remplacement de la norme Open Grid Service Infrastructure (OGSI) en vue de rapprocher les concepts Grid avec les servicesWeb [11]. Cela a nécessité une harmonisation dans la définition des interfaces et des protocoles d'échanges entre services. La base des services Web a finalement été adoptée [25]. Ceci a débouché sur une série de spécifications qui reprend les concepts OGSI [12] (voir table 2.1). Ces spécifications définissent les interfaces en Web Services Description Language (WSDL) et les protocoles en Service Oriented Application Protocol (SOAP) [9] permettant la création et la libération de ressources et la gestion des notifications. Ces normes sont stabilisées en vue de maintenir un bon niveau d'inter-opérabilité entre services et intergiciels. Nous nous intéressons plus particulièrement, dans le cadre de cette thèse, aux principes généraux décrits par ces normes. Ces principes sont développés dans la section 3.2.

Concept	Définition OGSI	Définition WSRF
Représentation du service Propriété du service Durée de vie du service Instantiation du service	Grid service FindServiceData SetServiceData RequestTerminationAfter RequestTerminationBefore Destroy Factory createService	WS-Resource WS-Resource Properties WS-Resource Lifetime WS-Resource factory
Identification du service Adresse du service	HandleResolver FindByHandle GSH (Grid Service Handler) GSR (Grid Service Reference) WS-Addressing	WS-Renewable References endPoint reference
Service de notificationssubscribe	NotificationSource, NotificationSink deliverNotification	WS-Notification
Gestion des groupes	ServiceGroup	WS-Service Group
Gestion des exceptions	Base fault type	WS-Base Faults

Tableau 2.1. Correspondance entre les concepts OGSI et WSRF

2.2 Principes

La norme OGSA est bâtie sur des principes liés aux architectures distribuées (voir section 1.1). Cette section décrit la taxinomie des services qui constituent l'entité opérationnelle de base dans une architecture orientée services et comment s'organise l'accès aux ressources réparties de façon sécurisée par des communautés d'utilisateurs.

2.2.1 Taxinomie des services

Les services se distinguent par leur fonction et leurs propriétés. Afin de les classifier, cette taxinomie s'inspire de celle proposée par IAN FOSTER:

– Une instance de service est une entité logicielle identifiable par une adresse unique, dont le rôle est d'exécuter des opérations dans un environnement d'exécution propre et de communiquer avec d'autres instances de service à travers une interface. Idéalement, cette interface et le protocole de communication entre services sont normalisés. Les services Grid se basent sur les normes du W3C, WSDL comme format d'interface et SOAP comme protocole.

– Un service sans état est une instance qui accomplit un traitement séquentiel de messages sans utiliser des informations qui ne sont pas contenues dans le message lui-même. L'état des résultats produits par ces opérations n'est pas conservé. Le contexte d'exécution de ces services ne nécessite pas de ressource de mémorisation. Ces services peuvent être assimilés à des fonctions. Un service de compression-décompression de données est un exemple typique de service sans état.

– Un service conversationnel est une instance qui accomplit un traitement séquentiel de tel sorte que le résultat d'une opération agit sur une opération suivante. Le comportement d'une opération est produit par une séquence logique de messages. De nombreux sites sur le Web utilisent cette technique à travers l'utilisation de cookies.

– Un service à état est une instance de service qui peut manipuler des données extérieures au service en réponse à des sollicitations, et qui conserve l'état des résultats produits par ces opérations en vue d'utilisations ultérieures. Le contexte d'exécution de ces services nécessite de la ressource de mémorisation propre.

– Un générateur de services ou service factory est une instance de service d'un type particulier car sa seule fonction est de générer d'autres instances de service. Cette forme de génération de service constitue l'un des principes fondamentaux de Service Oriented Computing (SOC) [58] qui est à la base du modèle OGSA.

– Un service persistant est instancié lors de l'initialisation de son environnement d'exécution (conteneur de services) et cette instance perdure indéfiniment. Ce service peut être éventuellement détruit par un processus externe (out-of-band process) mais reste normalement actif aussi longtemps que son environnement d'exécution le lui permet.

– Un service éphémère est instancié par générateur de service. Cette instance de service a une durée de vie limitée dans le temps ou plus précisément un état final naturel (soft-state). Ces services sont d'une importance capitale dans l'architecture Grid car ils permettent d'allouer ou de libérer de la ressource de façon dynamique et d'être composable de différentes façon. Ainsi plusieurs services élémentaires peuvent être orchestrés pour composer un service de plus haut niveau.

SERVICE À ÉTAT

Le terme état est vague dans la mesure où il peut couvrir de nombreux d'aspects d'un système informatisé. Le sens admis par la communauté Grid est qu'un service à état dispose d'une ressource dédiée de telle sorte que l'état de cette ressource peut s'exprimer selon une syntaxe univoque (i.e document XML), a un cycle de vie bien défini dans le temps et peut interagir avec d'autres services (avec ou sans état). De nombreux exemples, tels qu'une structure de fichiers dans un arbre ou des entrées dans une base de données, ou encore une combinaison de plusieurs de ces objets, peuvent être modélisés par des services à état.

2.2.2 Gestion des ressources réparties

Le principal défi de Grid est de coordonner de façon dynamique des ressources partagées entre diverses institutions. Le partage ne concerne pas seulement des fichiers mais aussi des accès à toutes sortes d'équipements informatisés. Une infrastructure Grid est constituée d'un

ensemble de ressources informatiques réparties géographiquement. L'usage de ces ressources est dynamique, ce qui implique une gestion qui prévoit des mécanismes de réservation de ressources pour des tâches planifiées et des mécanismes de libération de ces ressources une fois ces tâches accomplies. Ce principe se décompose en trois étapes élémentaires de la façon suivante :

1. Mutualisation de ressources informatiques hétérogènes distribuées ;

2. Virtualisation de cet ensemble de ressources en un unique super-ordinateur ;

3. Réification des parties de cet ensemble ainsi virtualisé. Cette étape permet d'allouer dynamiquement de la ressource aux services selon leur besoin.

Cette ressource ainsi rationalisée se traduit par de la capacité informatique quantifiable qui se présente sous trois formes :

– capacité de traitement

– capacité de stockage

– capacité d'échange

Nous appelons hôte une unité élémentaire de ressource comprenant ces trois formes de capacité. La figure 2.14 est une représentation schématique d'un hôte selon l'architecture VON NEUMANN avec les trois formes de capacité décrites ci-dessus. Ces capacités sont toujours présentes dans des proportions extrêmement variées selon les hôtes. Elles sont couplées en interne par un bus.

Pour qu'un hôte devienne partie intégrante d'une infrastructure Grid, il doit satisfaire les deux conditions suivantes :

– Avoir un intergiciel Grid installé et convenablement configuré (par exemple Globus).

– Être reconnu par l'infrastructure (par exemple un certificat d'hôte valide l'appartenance de l'hôte à l'infrastructure, comme décrit dans la section 3.2.4 traitant de la sécurité).

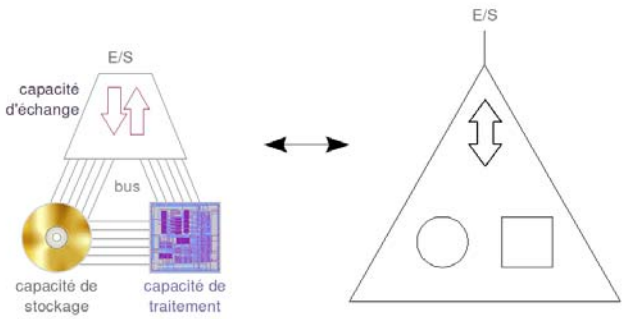


Figure 2.14. Représentation schématique d'un hôte selon l'architecture VON NEUMANN

Les hôtes sont interconnectés par un réseau. Ce réseau est généralement internet mais ce peut être un réseau spécialisé dans le cas notamment des interconnexions locales. Pour simplifier, nous considérons qu'un hôte est identifiable par une adresse fixe et unique. Cette adresse est, dans la vaste majorité des cas, une adresse Internet Protocol version 4 (IPv4).

Dans notre formalisme graphique initié dans [18] puis spécifié en détail dans [35], nous avons prévu de décrire comment s'organise la gestion des ressources depuis les éléments de capacité élémentaires jusqu'à l'étape de réification des ressources virtualisées. La figure 2.15 décrit ces différentes étapes de gestion des ressources.

Les étapes de mutualisation puis de virtualisation des hôtes Grid mènent à la constitution d'une large capacité de ressources élémentaires qui se mesure en PetaOctets (PO) pour le stockage et en TeraFlops (TF) pour la capacité de traitement. Par contre la capacité d'échanges va dépendre de la localisation des ressources et du couplage entre ces ressources. Si toutes ces ressources sont interdépendantes entres elles, nous voyons qu'il existe des solutions optimales pour les disposer dans une topologie adéquate. L'optimisation des flux et la localisation des ressources sont des problèmes du ressort de l'ingénierie des systèmes en réseaux et de la recherche opérationnelle. Ces questions ne sont pas abordées dans le cadre de cette thèse.

FLOPS

FLoating point Operations Per Second (FLOPS) est l'unité de référence servant à mesurer la puissance de traitement informatique. Un ordinateur de bureau de base comme un Pentium 4 opérant à une fréquence d'horloge de 2 GHz fournit une puissance de l'ordre de quelques GigaFlops (GF). Pour se fixer un ordre de grandeur, la puissance du premier superordinateur le Cray 1 était en mesure de fournir une puissance bien inférieure à 1 GF en 1976. Aujourd'hui, Blue Gene/L (voir photo 3.2), le calculateur le plus puissant du monde, délivre une puissance de 280,6 TF. Il est composé de 64 groupes de 1024 noeuds équivalents chacun à deux PowerPC 440 à 770 MHz (2,8 GF chacun). Les concepteurs de Blue Gene/P, la nouvelle génération de ce calculateur, annoncent avoir franchi le mur théorique du PetaFlop en 2007. Dans le cas des systèmes distribués, il faut distinguer la puissance maximum théorique (somme des puissances des différents éléments distribués) et la puissance effective qui dépend du couplage entre les différents éléments. Un exemple de système distribué de 40 TF est développé sous Linux avec 4 500 processeurs et de 128 000 Go de stockage [74].

2.2.3 Sécurité

La sécurité est un élément fondamental de Grid car elle détermine une topologie de reconnaissance mutuelle sur un ensemble d'entités que sont les utilisateurs et les ressources informatiques. La sécurité intègre les principes suivants :

1. La confidentialité qui repose sur un chiffrage à clé asymétrique.

2. L'intégrité qui repose sur un algorithme de hachage.

3. L'authenticité mutuelle qui repose sur l'échange de certificats X509 comportant des signatures électroniques.

4. La délégation qui assure la délivrance de certificats X509 valides sur une courte période.

5. L'autorisation qui détermine les conditions d'accès aux services d'une organisation virtuelle.

Les quatre premiers principes sont décrits par la spécification Grid Security Infrastructure (GSI) [26]. Le cinquième principe relève d'une autre spécification décrite par le Community Authorization Service (CAS) [53].

2.2.3.1 Sécurité 1/5 : confidentialité

La confidentialité de l'information est assurée dans GSI par un chiffrage à clé asymétrique de type RSA14. Cet algorithme consiste à générer deux clés différentes, l'une privée, l'autre publique, pour chiffrer puis déchiffrer les messages. L'émetteur d'une information chiffre ses messages avec la clé publique du destinataire. Ce dernier peut, à son tour, déchiffrer le message avec sa clé privée.

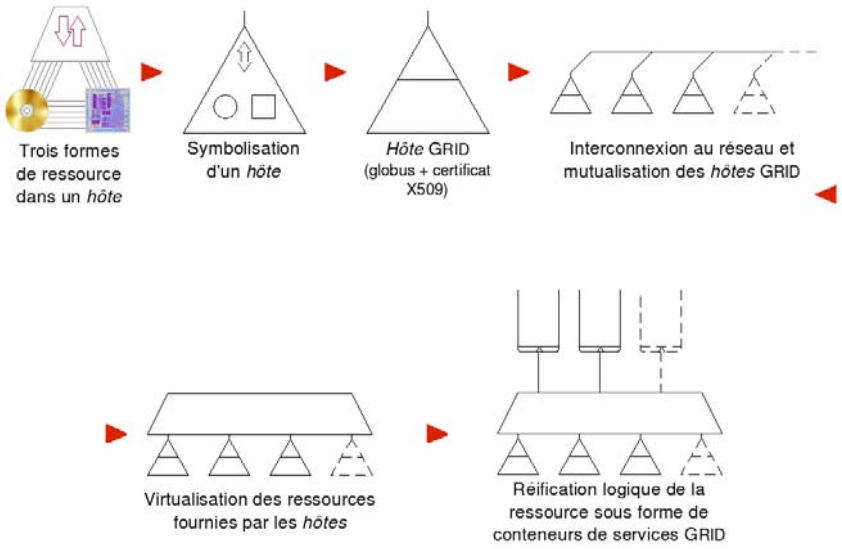


Figure 2.15. Différentes étapes pour la gestion des ressources dans Grid

2.2.3.2 Sécurité 2/5 : intégrité

L'intégrité signifie que le destinataire d'une information est assuré de la conformité de cette information avec l'original. Pour ce faire, l'information est transmise avec une empreinte qui a été générée à partir de cette information par une fonction de hachage. Ce principe consiste à associer une série d'octets de taille déterminée à une chaîne de caractères de taille quelconque. Si l'information est modifiée durant son transfert, celle-ci n'est plus conforme avec son empreinte et le test d'intégrité échoue. Cette transformation n'a pas de fonction réciproque, autrement dit, il n'est pas possible de déduire la chaîne de caractères originale à partir de l'empreinte. L'algorithme de hachage utilisé par GSI est le Message Digest 5 (MD5) [75]. Par exemple, l'expression « Bonjour de Montpellier » a pour empreinte MD5 la chaîne b41162f945e36624b357a2ffc1290f8a. Un autre intérêt des fonctions de hachage est de pouvoir mémoriser des mots de passe chiffrés dans une base de données.

2.2.3.3 Sécurité 3/5 : authenticité

Pour qu'une conversation soit totalement sécurisée, la confidentialité et l'intégrité des messages ne sont pas suffisantes. En effet, un agent mal intentionné pourrait transmettre une information en usurpant l'identité d'un autre agent puisqu'il chiffre ses messages avec la clé publique de son destinataire. Pour éviter ce type d'abus, GSI prévoit un mécanisme de reconnaissance mutuelle d'authenticité sur la base de certificats X509 [64] qui contiennent un certain nombre d'informations dont une signature chiffrée de l'émetteur. Cette signature est produite à partir de l'empreinte des informations sur l'identité du porteur et chiffrée avec la clé privée de l'émetteur. Ainsi, en comparant l'empreinte du certificat et la signature déchiffrée avec la clé publique de l'émetteur, le destinataire peut valider l'authenticité du message. Ce mécanisme est basé sur Transport Layer Security (TLS) [80], un dérivé de la norme Secure Sockets Layer (SSL) [79]. GSI permet l'assignation de certificats aux utilisateurs et aux hôtes.

La confiance au sein d'une communauté peut ensuite être instaurée au moyen d'un système de validation des certificats. Un certificat est valide s'il a été signé par une autorité habilitée à le signer ou Certification Authority (CA) [69]. Cette autorité est elle-même porteuse d'un certificat valide. Il apparaît ici une structure hiérarchique qui implique l'existence d'une autorité originale. Cette autorité, qui a elle-même signé sont propre certificat, se place à la racine de cette structure. La figure 2.16 décrit comment s'organise une telle structure avec deux autorités, dont l'une est à la racine. Les flèches indiquent la relation « est signataire du certificat X509 de... » . Dans la terminologie GSI, cette structure arborescente constitue un trust (voir encadré) que l'on peut décrire comme un espace à l'intérieur duquel la confiance mutuelle est garantie. En pratique, lors des échanges, les parties disposent d'une copie du certificat de l'autorité dont la clé publique est contenue dans ce certificat. Ceci évite la multiplicité des requêtes pour l'obtention de la clé publique de l'autorité de certification.

TRUST

Terme anglais de la terminologie Grid qui n'a pas d'équivalent simple en français. Un trust est un ensemble d'entités correspondant aux ressources et aux utilisateurs formant un ensemble Grid cohérent. Ces éléments se reconnaissent mutuellement comme faisant partie d'un même trust. En pratique, un trust est circonscrit par l'origine des signatures des certificats X509 (cf. Figure 2.16). En général, un environnement Grid couvre plusieurs domaines ayant leurs propres politiques de sécurité. Le trust permet de contrôler les interactions entre domaines en s'adaptant aux différentes politiques de sécurité locales des domaines.

2.2.3.4 Sécurité 4/5 : délégation

La délégation consiste à permettre à un agent de donner temporairement un pouvoir à un autre agent. Une situation typique se présente lorsqu'un service doit accéder à une ressource indépendamment de l'utilisateur. Pour ce faire, ce service doit disposer des privilèges pour accéder à cette ressource. Ces privilèges sont accordés temporairement par l'utilisateur sous forme d'un certificat signé par cet utilisateur. GSI prévoit la création d'une procuration appelée Grid Proxy Certificate (GPC) dans la terminologie GSI. GPC est une extension du protocole TLS. Ce certificat est une forme de mandat pour réaliser une séquence de tâches bien définie dans le temps afin de réduire le nombre de requêtes d'autorisation.

2.2.3.5 Sécurité 5/5 : autorisation

L'autorisation est un principe de sécurité qui spécifie la relation entre les membres et les services d'une organisation virtuelle. Ce mécanisme est décrit dans la section 3.2.4.

2.2.4 Organisations virtuelles

2.2.4.1 Généralités

Dans sa définition originale, une organisation virtuelle est une collection dynamique d'individus, d'institutions et de ressources [27]. Cette définition a évolué au fil du temps. Elle peut être de taille extrêmement variable en termes d'individus, de ressources et de durée indéfinie allant de quelques heures à quelques années.

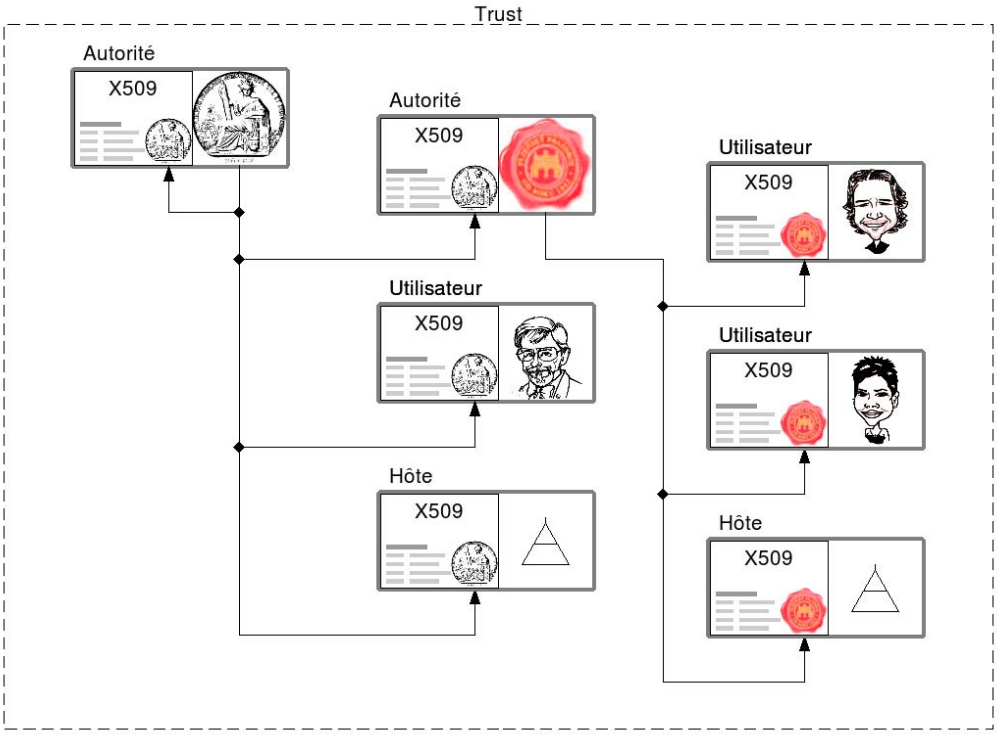


Figure 2.16. Structure hiérarchique de validation des certificats X509

Bien que cette définition ne le précise pas de façon explicite, l'article dont elle est extraite sous-entend que chaque organisation virtuelle dispose d'un contrôle total sur les règles de sécurité vis-à-vis des ressources qui lui sont allouées. Cet aspect est un apport particulièrement novateur de Grid. Selon IAN FOSTER, les solutions d'architectures distribuées orientées-systèmes ne sont pas adaptées à la gestion d'organisations virtuelles multiples [27] : il cite notamment CORBA qui ne gère que l'équivalent d'une simple organisation virtuelle ; Distributed Computing Environment (DCE), basé sur le principe des RPC, qui supporte de la ressource sécurisée mais est trop peu flexible ou encore Storage Service Provider (SSP) qui ne fonctionne qu'à travers un Virtual Private Network (VPN). De manière générale, les exemples cités sont composées d'un groupe de serveurs dont les utilisateurs se reconnaissent par le fait qu'ils appartiennent au même domaine. Or, le gros inconvénient du domaine est la rigidité de la

relation entre adresses IP et domaines DNS. De fait, ceci crée une forte dépendance entre la structure des organisations virtuelles et la topologie de l'infrastructure. Le VPN est une solution qui permet de contourner en partie ce problème d'un point de vue de la répartition géographique mais n'est pas adapté à des situations très dynamiques lorsque les membres et les ressources des organisations virtuelles doivent être souvent réalloués.

Parmi les organisations virtuelles existantes, nombreux sont les projets basés aux États-Unis avec des vocations scientifiques diverses : Sloan Digital Sky Survey en astronomie ; Collaboratory for the Multi-scale Chemical Science (CMCS) en chimie ; Network for Earthquake Engineering Simulation (NEES) et Southern California Earthquake Center (SCEC) en géologie ; Earth System Grid et Linked Environments for Atmospheric Discovery (LEAD) en climatologie ; Long Term Ecological Research (LTER) en écologie ; Biomedical Informatics Research Network (BIRN) en médecine ; Laboratory for the Ocean Observatory Knowledge INtegration Grid (LOOKING) en océanographie ou encore Laser Interferometer Gravitational-Wave Observatory (LIGO) en physique.

2.2.4.2 Gestion des autorisations d'une organisation virtuelle

Une solution rudimentaire, initialement proposée par Globus pour gérer les autorisations d'une organisation virtuelle, consiste à utiliser une table de correspondance entre utilisateurs et services à partir d'un simple fichier (connu sous le nom de fichier gridmap-file). Lorsque le nombre d'utilisateurs augmente dans une large mesure, l'utilisation d'un simple fichier devient une solution complexe à administrer. Ainsi, plusieurs solutions sur la base d'un annuaire Lightweight Directory Access Protocol (LDAP) ont été proposées, comme celle des projets EDG [70] et DataTag [71]. Une solution de ce type plus élaborée est apparue ensuite avec Community Authorization Service (CAS). CAS permet de déléguer la gestion d'un ensemble de ressources aux membres d'une organisation virtuelle, en laissant aux membres qui en ont le droit la possibilité d'ajuster les permissions des autres membres vers les services [53]. Virtual Organization Membership Service (VOMS) est une solution alternative au CAS [3] qui gère des comptes utilisateurs dans une base de données relationnelle de type MySQL. Ce service exploite davantage les concepts GSI, tel que la délégation à base de GPC ou encore la simple authentification en remplaçant l'utilisation de grid-proxy-init par une procédure plus évoluée, voms-proxy-init et présente quelques avantages sur CAS en agrégeant l'accès à de multiples organisations virtuelles simultanément. Toutefois, VOMS propose une solution d'administration centralisée des droits d'une organisation virtuelle avec l'idée qu'une organisation virtuelle est une structure hiérarchique composée de groupes et de sous-groupes. Dans la terminologie VOMS un groupe est un sous ensemble d'une organisation virtuelle composé de membres. Les groupes sont organisés de façon hiérarchique sous forme arborescente. avec un groupe racine à la base. VOMS introduit aussi la notion de groupe primaire pour chaque utilisateur. La notion de rôle permet d'attribuer des droits à un utilisateur pour exécuter des tâches. La distinction entre rôle et privilèges au sein d'un groupe est parfois un peu floue. Un utilisateur occupe un rôle de façon passagère alors que les privilèges sont déterminés au moment où un utilisateur devient membre d'un groupe.

VOMS apporte une dimension relationnelle supplémentaire avec la relation Groupe-Rôle dont nous devons tenir compte dans notre démarche de modélisation.

2.2.4.3 Caractérisation d'une organisation virtuelle

En somme, une organisation virtuelle se présente comme un système mutli-utilisateurs virtuel dont la taille peut varier dans le temps. C'est une couche d'abstraction très dynamique et indépendante de la structure physique des domaines.

La structure logique d'une organisation virtuelle doit être réduite à sa plus simple expression : d'une part ses membres, d'autre part ses services (le service est une couche d'abstraction vers des ressources). Une organisation virtuelle dispose d'un conteneur de services unique qui fourni les ressources aux services et dans lequel sont définies les autorisations d'accès aux services pour les membres. D'un point de vue de la sécurité, l'authentification est gérée au niveau du trust qui, d'un point de vue hiérarchique est placé au dessus de l'organisation virtuelle.

2.3 Synthèse : simplification du modèle

Les sections précédentes montrent, d'un côté, l'énorme potentiel de Grid en termes de gestion des ressources et en termes de sécurité, et, de l'autre côté, font apparaître une relative complexité du modèle d'architecture. Certes, ceci tient du fait de la richesse des fonctionnalités mais aussi parce que Grid est un objet très convoité qui attise déjà les intérêts industriels alors qu'il n'a pas encore atteint une pleine maturité conceptuelle. Pour cela nous avons jugé pertinent de faire abstraction des évolutions industrielles propres à une vision particulière en nous focalisant uniquement sur les concepts fondamentaux afin d'en dégager une représentation simplifiée. En nous appuyant sur l'ouvrage de référence [28] et les publications originales [27] [29], une représentation des concepts clés de Grid, sous forme d'un formalisme graphique, a été présentée pour la première fois en 2005 dans [18]. Une synthèse de ces concepts est représentée sur la figure 2.17. La gauche de cette figure représente les différents concepts et leurs variantes (service, organisation virtuelle et ressource). La droite de cette figure représente les relations et les opérations dynamiques telle que la relation de membre, l'accès à un service par un membre d'une organisation virtuelle, l'instanciation d'un service par un générateur de service, deux services en interaction ou encore l'exécution d'un service dans son conteneur.

Ces concepts, une fois établis, ont été réassemblés pour former un modèle d'architecture Grid simplifié (cf. Figure 2.18). Le bas de la figure représente l'espace des ressources composé d'hôtes ou de grappes d'hôtes. Des certificats X509 sont attribués aux hôtes et aux utilisateurs pour former un trust. Les deux opérations de virtualisation puis de réification de ces ressources permettent de constituer des conteneurs de services pour chaque organisation virtuelle. Le haut de la figure représente la formation des groupes composés de membres qui disposent de droits d'accès aux services de la communauté virtuelle. Ces droits sont définis dans une matrice M×S qui établit la relation entre chaque membre M avec chaque service S. Bien que ce ne soit pas visible sur cette figure dans un souci de clarté, cette matrice peut être extensible à une matrice (M + S) × S si l'on considère qu'au sein d'un même conteneur, tout service peut aussi avoir des droits sur les autres services.

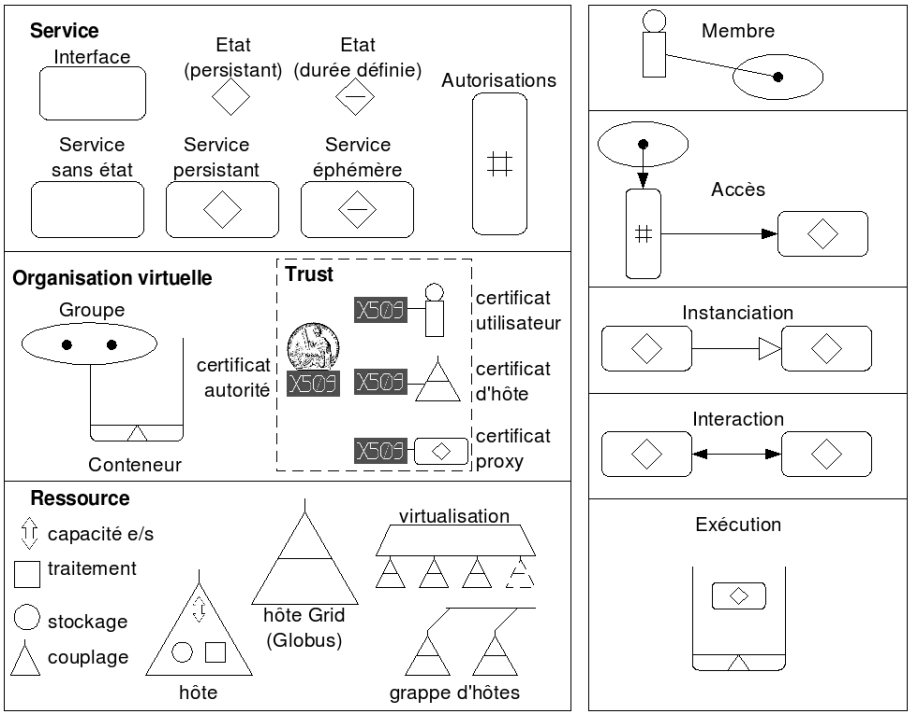


Figure 2.17. Synthèse des concepts Grid

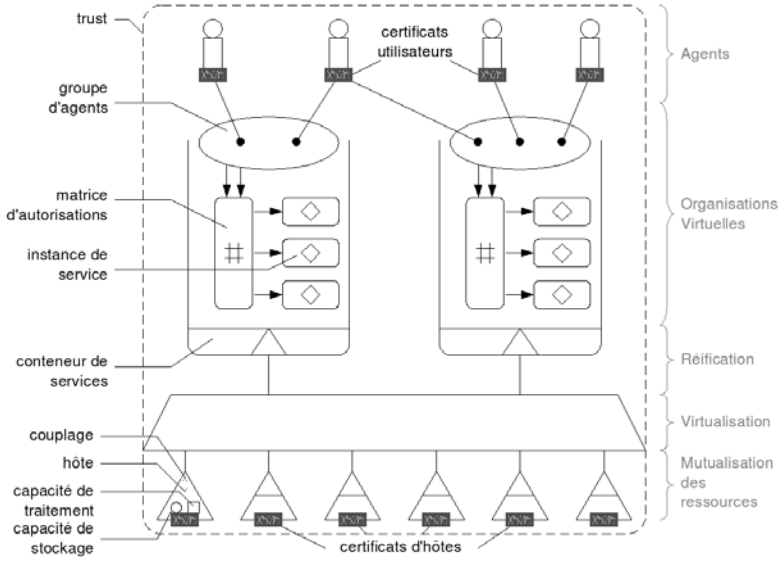


Figure 2.18. Architecture Grid simplifiée

Ce formalisme a été appliqué dans la recherche des synergies potentielles entre Grid et les SMA. Ce formalisme a d'abord permis d'effectuer une analyse conjointe des deux domaines pour en déduire une ontologie unifiée [19]. Ensuite, cette ontologie a permis de développer un langage de description [35] [36].

Conclusion

OGSA (Open Grid Services Architecture) spécifie rigoureusement les services Grid. Là où les services Web sont sans état et persistants, les instances de service Grid peuvent être soit avec ou sans état et éphémères ou persistants. Les services sans état sont synchrones (i.e., les messages ne peuvent être "bufferisés"), point-à-points (i.e., utilisables par un seul utilisateur), et interagissent par de simple requête/réponse. Ils retournent simplement une réponse à une invocation précise. Les services à état peuvent être asynchrones, multi-points et interagissent par des conversations. Ils permettent plus d'adaptation dans le service fourni. De plus, OGSA prévoit des mécanismes de supervision des défaillances pour accroître la disponibilité des services. Le service à état donc permet de déployer des services ubiquitaires qui se chargent de tous les aspects d'interopérabilité. Ceci garantit une quasi indépendance technologique au niveau du client. Plus récemment, WSRF (Web Service Resource Framework) a uniformisé les mécanismes pour accéder à des ressources à état avec des services Web/Grid.

On va voir dans les deux prochains chapitres (partie contribution), comment on peut bénéficier de ces technologies dans un environnement de formation où les participants doivent fortement collaborer, en adoptant en premier lieu les plateformes qui peuvent leur apporter plus de privilège et en deuxième lieu en intégrant ces deux plateformes.

Chapitre III

Le choix des plateformes

Introduction

Ce chapitre est divisé en deux parties : la première présentera une étude de quatre plateformes qui sont célèbres mondialement afin de sélectionner celle qui peut nous apporter plus de bénéfices en termes de fonctionnalités, de technologies en matière d'applications d'entreprise et des évolutions à venir. La deuxième va aborder tout d'abord le concept de grille ainsi que ses domaines d'application pour bien savoir que l'informatique collaborative inclut ce concept. Nous verrons ensuite comment les principes de base des grilles sont en partie implémentés dans Globus en étudiant son architecture. Enfin nous terminerons par une classification des différents types d'intergiciels par rapport à leurs fonctionnalités afin de sélectionner celui qui constitue la base de nombreux travaux.

1. Plateformes collaboratives

Une plateforme collaborative ou plateforme d'apprentissage en ligne, appelée parfois LMS (Learning Management System), est un site web qui héberge du contenu didactique et facilite la mise en œuvre de stratégies pédagogiques. On trouve aussi les appellations de centre de formation virtuel ou de plateforme e-learning (FOAD).

Il existe environ plus de 200 plateformes d'apprentissage en ligne dont une trentaine sous licences libres. Dans ce qui vient nous allons se baser uniquement sur ces dernières.

1.1 Étude des plateformes Collaboratives les plus célèbres

1.1.1 Moodle

Moodle fut créé par Martin Dougiamas, auparavant administrateur de la plateforme WebCT (maintenant Blackboard) à l'Université de Curtin en Australie. Dans le cadre de ses recherches doctorales, Martin a étudié les apports du constructivisme social dans la pédagogie en ligne. Ses travaux ont fortement influencé la conception de la plateforme Moodle.

1.1.1.1 Principales caractéristiques

Moodle tourne sans modification sur Unix, Linux, FreeBSD, Windows, Mac OS X, NetWare et autres systèmes qui supportent un serveur web, PHP et un Système de gestion de base de données (Mysql, PostgreSQL, ...). Il est également intégré au serveur de service Free-EOS

Elle présente de nombreuses caractéristiques partagées avec les autres plates-formes de formation en ligne : forums, gestionnaire de ressources, tests et neuf modules clé en main (Devoirs, Chat, Sondage, Glossaires, Journal, Étiquettes, Leçons, Wiki, Tests). Elle intègre aussi un module de création de tests d'entraînement. Les questions créées avec ce module peuvent être mutualisées et réutilisées dans différents contextes d'épreuve.

Moodle est très modulable car dès le départ, elle a été créée de manière modulaire : elle permet de répondre aux besoins d'un formateur isolé comme d'une institution académique. Aujourd'hui, le développement de Moodle est fortement influencé par les demandes de la communauté d'administrateurs et d'utilisateurs (enseignants, pédagogues) de Moodle. On peut développer de nouveaux modules facilement puisque Moodle s'appuie sur le PHP, le langage de script le plus utilisé dans le monde de l'Internet.

Elle présente une interface très conviviale avec les formateurs, apprenants et administrateurs. Sa prise en main s'effectue en moins d'une heure. Elle est très facile à

installer. Son coût d'hébergement est très faible (100 à 200 euros). Moodle est supportée par Fantastico un installeur de logiciels sous licence open source chez certains hébergeurs.

La plate forme Moodle a été créée dès le début dans une perspective mondiale et donc multilingue. Chaque installation comporte un pack de 40 langues. On peut changer à loisir la langue de la plateforme.

Des filtres permettent d'utiliser facilement des fichiers multimédia ou des expressions mathématiques au sein des pages Moodle.

Des rapports d'usage détaillés pour chaque apprenant permettent de superviser les efforts d'apprentissage.

La communauté Moodle (4900 individus au mois de février 2005) est fortement structurée à travers de nombreux forums présents sur le site moodle.org : il s'agit de communautés de pratiques centrées sur des problématiques précises.

S'inspirant fortement du courant pédagogique du constructivisme social, Moodle propose des outils de gestion du savoir (knowledge management) : wiki, fil RSS, forums et blog. Ces outils favorisent le travail collaboratif d'une communauté centrée autour d'un projet d'apprentissage.

Moodle facilite la gestion dynamique d'un cours avec le calendrier. Chaque cours se présente comme un portail composé de blocs que le tuteur peut afficher à sa guise tout au long du déroulement du cours, évitant ainsi une surcharge informationnelle.

1.1.2 Claroline

Claroline est un environnement collaboratif d'apprentissage gratuit, réputé pour sa facilité d'utilisation et sa simplicité. La gestion de cette plateforme ne requiert pas de compétences techniques particulières. Les coûts associés à la formation et au support seraient donc grandement diminués.

1.1.2.1 Principales caractéristiques

a. Technologies utilisées

Claroline utilise les composantes usuelles de la pile technologique LAMP (Linux – Apache – MySQL – PHP – Pear. La plateforme intègre d'autres applications issues du logiciel libre en PHP, comme l'accès aux sources de données par PEAR ou l'indexation du contenu par PHPDig. La traduction est quant à elle facilitée par PhpLangEditor, une extension de Firefox consacrée à l'édition des fichiers langues d'applications php.

b. Normes supportées

Le support récent du standard IMS-QTI 2.0 donne la possibilité d'exporter et d'importer des exercices standardisés, ce qui est d'un grand secours pour archiver les exercices ou pour les réutiliser dans d'autres cours. L'interopérabilité des services est aussi présente dans la plateforme. L'utilisation de services Web standards, entre autres, le *Single Sign-On*.

L'utilisation des objets d'apprentissages de la norme SCORM sont implémentés. Pour la création des SCO, les développeurs de Claroline suggèrent l'emploi de l'extension SCORM RTI de Dreamweaver, tout en documentant de façon complète leur écriture manuelle.

c. Architecture

La documentation spécifique sur l'architecture de Claroline est, à toute fin pratique, absente. Les utilisateurs sont même invités à parcourir le code pour saisir les particularités de l'API. Les fonctions, classes et événements qu'on y trouve forment un ensemble minimalement commenté.

Composants : En plus des composantes de base, soit les outils et l'API, certaines améliorations ont été soumises par la communauté de développeurs. « Claroline UTC » est une modification et adaptation de la version 1.3 du système fournie. L'UQAR a elle aussi fournie un déploiement d'outils personnalisés, tels une FAQ, un glossaire, un outil de résultats et un module de rétroactions.

Possibilités d'expansion : Il est possible de développer de nouveaux outils pour Claroline via son API. Les fonctionnalités qu'elle couvre sont suffisantes pour envisager un grand nombre de nouveaux modules.

Mise à l'échelle : Comme l'architecture formelle de Claroline n'est pas documentée, il est difficile d'évaluer les capacités de mise à l'échelle de la plateforme. Selon les concepteurs, le système pourrait accommoder des dizaines de milliers de cours et d'utilisateurs et aurait été conçu pour une université de 20 000 étudiants. La technologie PHP n'a pas la réputation d'être au niveau de la mise à l'échelle, si on la compare à Java. Cependant, l'étude des autres LMS Open Source tournant sous PHP démontre que le langage se comporte bien pour un grand nombre d'usagers.

Internationalisation : Le produit est actuellement traduit dans 34 langues différentes, dont le français, l'anglais, l'arabe et l'espagnol. C'est près de 1 500 termes à traduire en tout. L'effort d'internationalisation s'effectue au moyen de deux modes d'exécution de Claroline, soit le mode Traduction et le mode Production. Dans le premier, tous les termes sont stockés et lus dans un seul et même fichier, ce qui facilite l'édition, mais diminue les performances. Dans le second mode, tous les termes sont répartis dans des

fichiers plus petits et propres à chaque outil, ce qui augmente les performances, mais accroît la redondance.

d. Outils disponibles

Claroline offre onze outils, la plupart s'avérant des outils usuels d'un LMS. L'apprenant peut organiser son temps au moyen de l'*agenda* et des *annonces*, tous deux mis à jour par le formateur. Les apprenants peuvent être réunis en *groupes* possédant des outils collaboratifs privés qui leur sont réservés. De ces outils, on retrouve les *forums*, les *espaces de discussion*, une zone de dépôt de *documents* divers, ainsi qu'un d'édition de type *Wiki*.

Pour évaluer et suivre l'apprentissage des participants, un outil permet de créer des *exercices* en ligne sous la forme de listes de questions. Les apprenants postent aussi leurs travaux et fichiers pertinents via l'outil *travail*. Ces sessions de travail, gérées individuellement ou par groupe, offrent des objectifs et contraintes différentes. Les exercices et les documents peuvent être composés et séquencés via l'outil SCORM de *parcours pédagogique*.

e. Volet pédagogique

L'approche ayant menée à l'élaboration de Claroline était avant tout de doter les formateurs d'une autonomie d'action. Les enseignants devaient donc pouvoir expérimenter aisément un jeu de fonctionnalités réduites. Claroline, conçue par des enseignants, se caractérise par une certaine simplicité en comparaison aux LMS poids lourds.

Le modèle pédagogique employé ici dérive de l'approche constructiviste. L'information est transformée en connaissances par les activités de l'apprenant, pour ensuite être réinjectée dans la boucle systémique. Les outils permettant cette acquisition de connaissance doivent promouvoir l'apprentissage.

1.1.3 KEWL

Le système de e-Learning et *framework* KEWL est le résultat du travail du *African Virtual Open Initiatives and Resources*, ou AVOIR. Ce groupe de travail, initié par l'Université de Western Cape, vise la collaboration d'intervenants dans le domaine du logiciel et de l'éducation afin de concevoir, développer et supporter des logiciels éducatifs en éducation supérieure. AVOIR cherche à créer des opportunités éducationnelles et d'affaires qui bénéficieront au continent africain et à ses ressources humaines.

1.1.3.1 Principales caractéristiques

a. Technologies utilisées

KEWL utilise les technologies LAMP (Linux – Apache – MySQL - PostgreSQL – PHP – Pear) dans sa nouvelle mouture. Il est à noter que seule la version 4.x de MySQL est supportée. Bien que le paradigme orienté objet de PHP n'est apparu que dans la version 5, et que KEWL ne supporte actuellement que PHP 4, le système adopte une approche OO. Une douzaine de stagiaires travaillent présentement à la mise à jour vers PHP 5.

L'abstraction des accès à la base de données est assurée par librairie DB du *framework* PEAR. Certaines autres fonctionnalités de ce *framework* sont utilisées, dépendamment du contexte. Par exemple, pour ce qui est de l'authentification, les concepteurs ont utilisé la connectivité à LDAP.

KEWL intègre, encourage et documente le développement de services Web. Ces serveurs peuvent être développés dans n'importe quel langage et accédés via SOAP, le protocole le plus répandu actuellement. La description du service est faite via la librairie PHP Nusoap.

b. Normes supportées

L'application tente de suivre les diverses normes d'échanges XML dans le domaine du e-Learning. KEWL supporte donc les schémas SCORM 1.2 et IMS, en plus du contenu Dublin Core Metadata. Il est aussi conforme au protocole du *Open Archive Initiative*. Pour remplir la vision du Web 2.0, KEWL bénéficie du support des contenus xHTML, RSS, Atom et Podcast.

c. Architecture

Composants : KEWL.NextGen est bâti autour du framework applicatif KINKY (Kinky is not Kewl Yet), développé au Free Software Innovation Unit de l'Université de Western Cape. Le *framework* est avant tout un port de la version antérieure de KEWL, qui était alors en ASP / SQL Server, vers PHP orienté objet. Il utilise le pattern Model–View–Controller, ou MVC. Afin de préserver la consistance de l'architecture, le langage de modélisation UML fait partie intégrante des processus de développement entourant KEWL. Les modifications majeures ou les nouveaux modules doivent tous être proposés via un schéma explicatif UML, qui sera discuté et approuvé par la communauté de développeurs.

Possibilités d'expansion : L'approche MVC modulaire permet, selon ses concepteurs, une meilleure automatisation de l'application des permissions et de la gestion des modules. Le *framework* est de type « frontcontroller », où chaque requête est traitée par un script unique, puis est acheminée au module approprié dépendamment des paramètres de l'appel. Les capacités d'expansion seraient aussi virtuellement illimitées.

Mise à l'échelle : Afin de pallier à une qualité de service variable de la part de certains fournisseurs de services Internet, une fonctionnalité de sites miroirs est disponible pour KEWL. ADM (Active Dynamic Mirroring) permet de répartir et de synchroniser en temps réel les données et objets sur un regroupement de serveurs.

Internationalisation : L'interface multilingue est une des préoccupations majeures du logiciel KEWL. Les développeurs contribuant au projet sont encouragés à ne jamais utiliser de chaînes de caractères codées en dur. En fait, c'est un des critères d'acceptation du code contribué. Un utilitaire externe, le Translation Helper, programme en C#, permettra de traduire hors ligne le contenu dans un autre langage. Cet outil suit l'architecture n-tier de Microsoft.

d. Outils disponibles

Présentement, environ une centaine de modules sont disponibles, et ce chiffre est supposé passer à 250 d'ici la fin de l'année. On retrouve des modules d'édition de contenu, de quiz, de devoirs, de blogues, de calendriers, de recherche, d'images, et plusieurs autres.

Les deux implémentations majeures réalisées à partir du *framework* KINKY sont kGroups, qui fournit plusieurs outils de communication et de collaboration, ainsi que kForums, qui offre des discussions multilingues et sous forme de *mind map*. Le module de PodCasting mérite aussi une attention particulière.

e. Volet pédagogique

En plus du produit livrable lui-même, la maîtrise d'un environnement de développement Open Source fait partie des attentes du projet. Ces capacités seront transférées aux divers intervenants du projet lors de séminaires. Aussi, le projet se veut rassembleur : plusieurs étudiants en ont fait leur thèse de maîtrise.

Un module spécifique est consacré aux études supérieures afin de supporter la supervision de la recherche. Un des objectifs est de rapprocher le superviseur et l'étudiant, en offrant des outils de discussion, un journal des activités, un *gantt chart*.

De façon générale, les processus de design de cours sont en cours d'incorporation. Plus particulièrement, l'utilisation des technologies de e-Learning réduira les problématiques entourant un trop grand nombre d'élèves dans certaines classes, par exemple les Maths 101 ou 103. Pour ce faire, l'Université de Jos veut donc convertir en priorité le contenu mathématique.

1.1.4 Sakai

Sakai (www.sakaiproject.org) désigne à la fois :

- un projet auquel sont rattaché une fondation, un comité de direction et des partenaires institutionnels et commerciaux;
- une communauté qui regroupe plusieurs institutions qui coopèrent et maintiennent Sakai. Elle regroupe actuellement plus de 80 universités dans plusieurs pays au monde ainsi qu'une dizaine de partenaires commerciaux;
- un framework, ouvert et extensible, pour la construction d'outils de collaboration;
- un produit basé sur le framework qui inclut une suite d'outils pour le support de l'apprentissage, de la collaboration et de la recherche.

L'objectif de Sakai est de produire une plateforme complète de gestion de cours libres de qualité équivalente ou supérieure aux autres produits retrouvés sur le marché. Sakai (Cf. figure 3.1) est publié sous licence « Open Source ».

Le produit s'adresse, dans un premier temps, aux institutions universitaires. Ce sont ces dernières qui définissent les orientations, les spécifications et les priorités quant au développement du projet.

Le *framework* ainsi que les outils développés sont génériques et peuvent être réutilisés comme base pour la construction d'outils de collaboration ou d'apprentissage dans n'importe quel domaine.

Le projet Sakai a été lancé en 2004 par quatre universités américaines avec pour objectif de consolider leurs développements en matière de plateforme d'apprentissage. Chacune de ces universités, soit Indiana University, Massachusetts Institute of Technology (MIT), Stanford University et University of Michigan utilisaient des systèmes de gestion de cours différents, souvent développés à l'interne. Au groupe se sont joint des membres de uPortal ainsi que du projet OKI (Open Knowledge Initiative).

Le projet a initialement été financé par la Fondation Andrew W. Mellon, à plus de 2.4M $ US, ainsi que par les universités instigatrices du projet sous forme de contribution en ressources évaluée à 4.4M $ US.

La version 1 est parue en octobre 2004. Depuis, le projet n'a cessé de prendre de l'ampleur, alimenté par la participation d'une communauté de développement toujours plus importante.

Le projet Sakai est soutenu par une fondation dont l'objectif est de coordonner et d'assurer l'évolution du produit. La fondation regroupe des partenaires commerciaux et institutionnels.

Seuls les partenaires institutionnels ont le droit de décision sur les orientations du projet. Les partenaires commerciaux regroupent, entre autres, Apple, IBM, Oracle, Sun Microsystems, Pearson Education, Unicon et Unisys.

Le modèle de développement utilisé par la Fondation Sakai est le « Community source model ».

Il s'agit d'une variante du modèle de développement Open Source traditionnel à l'intérieur duquel un comité de direction, chargé de coordonner les activités de développement, est impliqué.

Ce dernier recueille les exigences et les contributions financières des partenaires lui permettant de réaliser une part du développement. Comme pour tout projet Open Source, des contributeurs individuels et institutionnels prennent part au développement. Ce modèle permet au produit une évolution plus rapide et plus structurée tout en assurant le respect des besoins des institutions qui le financent.

La Fondation Sakai regroupe actuellement plus de 80 partenaires institutionnels qui sont à différents stades quant au déploiement de Sakai. Autres que les partenaires officiels de Sakai, plusieurs institutions dans le monde l'ont adopté comme environnement collaboratif d'apprentissage.

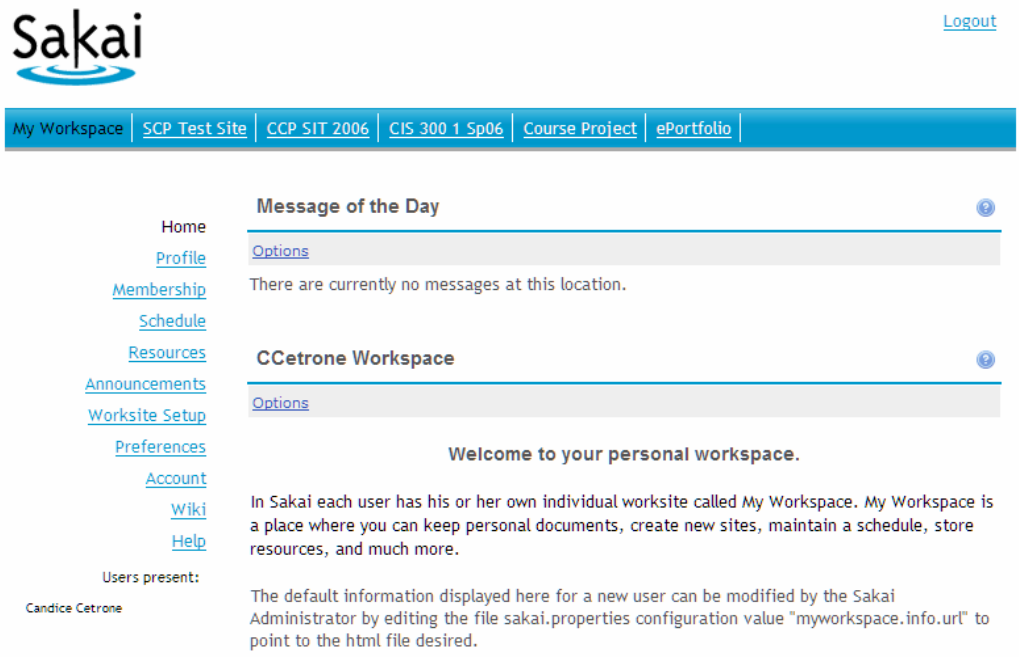


Figure 3.1. Page d'accueil de l'étudiant dans Sakai

1.1.4.1 Principales caractéristiques

a. Technologies utilisées

Sakai est développé en Java et utilise plusieurs technologies issues de J2EE. Il est construit sur des *frameworks* et librairies libres tels que Spring, Hibernate, log4j et

Jakarta Common. Différents mécanismes de présentation dont JSF, Velocity et XSL sont utilisés pour la présentation.

Le déploiement s'effectue généralement à l'aide du serveur Web Apache et du serveur Java Apache Tomcat.

Sakai permet la mise en place d'une architecture orientée service (SOA) en offrant la possibilité d'interconnecter les différents composants à l'aide des services Web. Pour la distribution et l'accès à des services distants, la norme WSRP est aussi supportée. Ces mécanismes permettent l'intégration de développements effectués sur la plupart des technologies et langages disponibles (PHP, .NET, RubyOnRail, Python, …)

J2EE est principalement utilisé pour le développement des applications Web. Les Servlets et les Portlets sont utilisés respectivement pour gérer l'accès à l'application et l'accès aux services.

Les pages JSP sont construites à l'aide de JSTL et exploitent le *framework* JSF comme gestionnaire de présentation. Ce *framework* favorise la construction d'interfaces à l'aide de composants.

Il se base sur le patron de conception MVC (Modèle Vue Contrôleur).

Spring est un *framework* J2EE léger qui vise à réduire l'effort de développement tout en favorisant l'augmentation de la qualité du produit. Il exploite de façon simple, non intrusive, les objets (beans) et facilite la gestion des différents composants d'une application telle que la persistance, les services et la présentation. L'utilisation de Spring encourage un bon découpage de l'application et une architecture modulaire.

Hibernate est un *framework* pour le « mapping » relationnel/objet et la gestion de la persistance.

Il agit comme une couche d'abstraction pour l'accès à la base de données en prenant en charge toutes les interactions avec celle-ci. Les opérations reliées à la persistance peuvent donc être réalisées par simple manipulation d'objets. Hibernate permet un support transparent pour la plupart des fournisseurs de bases de données (Oracle, MSSQL, SyBase, MySql, Postgres, HSQL, …).

Chacun des outils intégrés à Sakai est libre d'exploiter ses propres technologies et méthodes de développement. Les outils et services de Sakai sont déployés dans des contextes d'applications Web différents évitant ainsi d'éventuels conflits entre les librairies utilisées.

b. Normes supportées

Sakai se conforme à plusieurs normes et standards, tant au niveau technologique que pédagogique.

Les technologies Java utilisées par Sakai couvrent plusieurs des spécifications du Java Community Process. Que ce soit pour les Servlets, les Portlets, JSF ou les dépôts de contenu, le respect des normes JSR permet l'interopérabilité avec les autres systèmes, libres ou propriétaires, issus du monde Java. Par exemple, un Portlet implémentant les interfaces de la norme JSR-168 peut être exécuté sur Sakai, uPortal, Apache JetSpeed, JBoss Portal, IBM WebSphere, Oracle Portal Service, Novell eXtend, Sun One Portal et plusieurs autres.

Sakai supporte aussi rigoureusement les normes XHTML et CSS du W3C. Le style d'affichage est concentré au niveau des fichiers CSS et la refonte de l'aspect visuel de l'application Web peut se faire en modifiant uniquement ces fichiers. Les dernières normes en matière d'accessibilité sont aussi supportées. Plusieurs outils permettent l'utilisation de format d'échanges tel que RSS.

Les fonctionnalités reliées aux services Web se conforment aussi à plusieurs normes. Les interfaces sont définies à l'aide du format WSDL et le format d'échange WSRP est utilisé pour le partage de services. WSRP (pour Web Service Remote Portlet) permet d'accéder à un Portlet exécuté sur un serveur distant. Il permet donc à la plateforme de présenter des outils provenant de différentes sources.

Les normes de e-Learning actuellement supportés sont IMS, OKI et IEEE LOM. Une implémentation SCORM est aussi en cours de développement. Le respect de ces standards permet, entre autres, l'exploitation d'une taxonomie universelle pour la description des objets d'apprentissage.

Le support de formats d'échanges tel que SCORM permet l'interopérabilité entre différentes plateformes d'apprentissage.

c. Architecture

Sakai offre un *framework* ouvert et extensible permettant plusieurs niveaux de personnalisation.

Il rend possible l'intégration d'outils et de composants ainsi que des mécanismes permettant de modifier le comportement de la plateforme sans en modifier la structure.

Composants : Sakai se décompose en trois composants principaux : les outils, les services et le *framework*. Le *framework* agit comme le point central de l'application. Son rôle est de gérer et de référencer les services et outils. Les services regroupent l'ensemble des opérations relatives à la plateforme. Il s'agit, entre autres, de l'authentification, de la gestion des utilisateurs, de la gestion des contenus, de l'accès aux données, de la gestion des groupes, etc. Les outils correspondent aux fonctionnalités présentées aux utilisateurs (barres de navigation, texte de présentation,

forum de discussion, calendrier…). Seuls les outils génèrent du contenu graphique. Les contenus qu'ils présentent sont accédés via les services.

Possibilités d'expansion : La plateforme offre d'impressionnantes possibilités d'expansions liées au fait que tous les services et outils peuvent être redéfinis afin d'en modifier le comportement. Pour chacun des types de service, il existe un API définissant les méthodes devant être supportées. Pour redéfinir un comportement de la plateforme, il suffit d'écrire un nouveau service conformément à l'API concerné et de spécifier ce nouveau service au niveau de la configuration. Le tout peut donc être effectué sans altérer la structure de la plateforme. Les services peuvent aussi être redéfinit à l'aide de service Web, permettant ainsi leur développement à l'aide d'un langage tiers (PHP, .NET, …). Un exemple concret de la flexibilité de la plateforme pourrait être l'intégration de Sakai avec des systèmes existants. Il serait alors possible de redéfinir les services d'accès aux utilisateurs de gestion des autorisations et de création des cours afin qu'ils puisent des données à l'intérieur d'un système tiers. Le respect de nombreux standards en matière d'intégration de composants augmente aussi les possibilités d'expansion de la plateforme. Par exemple, un outil développé selon la norme JSR-168 pourrait être intégré tel quel à un environnement Sakai sans nécessiter d'effort d'adaptation. La même situation s'applique aux services distribués à l'aide de la norme WSRP.

Mise à l'échelle : Sakai bénéficie des possibilités de mise à l'échelle issue de J2EE. Il est possible d'exploiter une grappe de plusieurs serveurs et d'appliquer des mécanismes de répartition de la charge et de la reprise en cas de panne.

Internationalisation : La plateforme Sakai ainsi que les outils qu'ils l'accompagnent sont entièrement internationalisés et permettent le support multilingue. Les langues actuellement supportées sont l'allemand, le coréen, le danois, le portugais, le chinois, le japonais, le slovaque, le catalan, le français, l'espagnol et l'anglais.

d. Outils disponibles

Sakai offre plus d'une vingtaine d'outils différents couvrant, entre autres, les besoins en communication, en gestion de cours et en évaluation. La grande majorité des outils conventionnels (forums, fichiers, collecte de travaux, listes de diffusions, clavardage, calendrier, recherche, quiz, aide en ligne…) s'y retrouvent.

Parmi les outils plus spécifiques à Sakai, notons la présence d'un wiki, d'un lecteur RSS, d'un outil de gestion des résultats, d'un mécanisme d'archivage des courriels ainsi que la compatibilité WebDAV.

e. Volet pédagogique

La grande majorité des outils d'apprentissage en ligne étant couverts par Sakai, cette plateforme permet de s'adapter à une vaste gamme de besoins en matière d'enseignement et d'apprentissage en ligne.

Cette plateforme ainsi que les outils qui y sont rattachés ont été principalement conçus pour une utilisation dans un contexte pédagogique, avec comme objectif de répondre aux besoins des universités membres du projet. Cela est reflété dans les fonctionnalités disponibles pour chacun des outils.

La plupart des outils offrent une approche maître / élève; c'est-à-dire que l'enseignant a le contrôle sur les ressources de chacun de ses cours. Ces outils offrent des volets de gestion permettant à l'enseignant de les configurer en fonction de ses besoins. La flexibilité des outils et de l'environnement combiné à ses possibilités d'extension permet l'exploitation de la plateforme pour des contextes pédagogiques variés.

La facilité d'utilisation est aussi importante pour les outils à saveur pédagogique. Sakai offre une interface légère et un guide de style commun pour l'ensemble des outils, facilitant ainsi l'utilisation de la plateforme.

1.2 Adoption de la plateforme collaborative Sakai

Les plateformes à l'étude diffèrent grandement en ce qui a trait aux technologies, à l'architecture, aux fonctionnalités, aux perspectives d'avenir qui les entourent et à l'effet de levier qu'elles peuvent offrir à notre projet. Après considération de ces éléments, on a décidé d'opter pour Sakai.

Les plateformes considérées adoptent soit PHP (Moodle, Claroline ou KEWL), ou Java (Sakai). Le premier langage est performant et simple, en plus de bénéficier d'un large bassin de développeurs. La maintenance d'une application PHP est relativement aisée, ce qui en fait un choix intéressant pour les établissements et les projets de faible taille. Parallèlement, les coûts de maintenance de projets Java de grande ampleur sont généralement plus faibles.

Sakai étant basé sur le paradigme Java, il hérite d'un modèle axé sur la réutilisation, la mise à l'échelle et l'interopérabilité. Dès l'installation initiale, la plateforme supporte des charges importantes. Ce n'est souvent pas le cas des applications PHP, telles Moodle, qui nécessitent alors des plans de déploiement minutieux. D'autre part, Sakai met en place une architecture orientée service modulaire permettant une intégration avec divers systèmes d'informations déployés dans les universités, notamment les systèmes de gestion scolaires, les annuaires ou les portails. Cette aise d'intégration est

aussi due à la conformité de Sakai envers plusieurs autres spécifications (portlet, JSP, servlets), normes et standards (XHTML, CSS, RSS, IMS, OKI, IEEE LOM).

Les fonctionnalités d'apprentissage disponibles (quiz, ressources, forums, évaluations, etc.) sont sensiblement les mêmes d'une plateforme à l'autre. Sakai se démarque ici par des outils de travail collaboratif génériques axés sur la recherche. Il est aussi le seul à intégrer de base un module de Portfolio, soit le projet OSP.

Sakai est un produit réalisé par et pour les universités. Une fondation universitaire a d'ailleurs été créée spécifiquement pour assurer à sa jeune communauté pérennité et croissance. Pour ce faire, elle organise deux conférences internationales par an afin d'amener les intervenants à partager leurs expériences et résultats de recherche. Ces conférences ont lieu en Amérique et en Europe, pour se transporter sous peu en Asie et en Afrique. Une participation du CRIM et des universités d'Afrique du Nord (y compris notre laboratoire LRI-Annaba) au sein de la communauté Sakai a donc toutes les chances d'être bien reçue. Les retombées envisagées sont une visibilité accrue ainsi qu'un maillage et un réseautage prospères et durables.

2. Intergiciels

2.1 Les grilles de calcul

2.1.1 Origine du terme « grille »

Foster et Kesselman ont introduit pour la première fois en 1998 la notion de grille dans le cadre du projet Globus. Le terme « grille de calcul » est la traduction de *grid computing*. Ce dernier vient de l'analogie avec le réseau de distribution d'électricité appelé *electrical power grid* en Anglais. En effet, on peut rapprocher la puissance de calcul actuelle avec ce qu'était l'électricité au début du XXe siècle. On maitrisait l'électricité et disposait de matériel pour l'exploiter mais chaque personne devait produire sa propre électricité et la consommer sur place. Ce n'est qu'avec le développement du réseau de distribution électrique que la vraie révolution a pu s'opérer en offrant, au plus grand nombre, la possibilité de recevoir et utiliser de l'électricité sans se soucier de la façon ou de l'endroit où elle a été produite.

Actuellement, chaque ordinateur ou ferme de calcul utilise la puissance de calcul qu'elle a généré localement. L'idée du *grid computing* est de mettre à disposition ses ressources en échange d'un accès aux ressources des autres utilisateurs.

2.1.2 Définition

Le concept de grille de calcule tant encore relativement récent, on peut en trouver de nombreuses définitions que ce soit dans la littérature scientifique ou sur Internet. Nous discuterons brièvement ici la définition ci-dessous avec les différentes notions abordées dans celle-ci.

Une grille de calcul est une infrastructure matérielle et logicielle qui fournit un accès fiable, uniformisé, répandu et bon marché à des ressources informatiques hautes performances.

Nous parlons d'une **infrastructure** car une grille de calcul vise à mutualiser un grand nombre de ressources (cycles CPU, données, espace de stockage). Pour ce faire, il est nécessaire de disposer d'une architecture matérielle permettant l'interconnexion de ces ressources, mais aussi d'une architecture logicielle afin de pouvoir faire cohabiter, surveiller et contrôler cet ensemble. Dans la suite de ce chapitre, nous décrirons quelques caractéristiques de cette infrastructure.

La nécessité d'un service **fiable** est fondamentale. Les utilisateurs doivent avoir l'assurance qu'ils auront accès à un service prévisible, constant et de haute performance de la part des différents composants qui constituent la grille. Selon le type de programme exécuté sur la grille, la signification que l'on donne à cette performance peut être de différentes natures telles que la bande passante du réseau, le temps de latence, la puissance de calcul brute, les services logiciels disponibles ou encore la sécurité.

Avoir un service **uniformisé** est le second objectif de la grille. Comme pour l'électricité, il est important d'avoir des services standardisés accessibles via des interfaces standards. Sans de telles normes, le développement d'applications ainsi que le déploiement de la grille sont fortement compromis voire impossibles. Un des grands défis du Grid est d'arriver à concevoir de tels standards devant gérer des matériels et logiciels très hétérogènes (alors que les fermes sont plutôt homogènes) tout en conservant de hautes performances.

Un accès **répandu** garantit de pouvoir retrouver les différents services quel que soit l'environnement vers lequel on pourrait migrer. Cela ne signifie pas que les ressources sont accessibles de n'importe où universellement. De même qu'une nouvelle maison n'a accès à l'électricité qu'une fois le câblage installé et après avoir souscrit chez un fournisseur, l'accès à la grille aura une circonscription et un contrôle d'accès similaire. On pourra toutefois compter sur un accès suffisamment universel pour pouvoir être utilisé quel que soit l'environnement d'exécution.

Enfin, l'accès à cette infrastructure doit être relativement **bon marché** au regard des bénéfices qu'elle peut apporter. Les industriels dépensent des dizaines de millions pour construire des centrales électriques car cet investissement leur est rentable. D'autre part, l'investissement financier pour pouvoir bénéficier du réseau électrique est, bien heureusement, infiniment plus restreint. Idéalement, la Grille de calcul devrait jouir des mêmes attraits économiques.

En plus des exigences abordées dans cette définition, nous pouvons également rajouter :

– La **confidentialité** et la **sécurité**. Mettre de telles ressources à disposition de différentes communautés d'utilisateurs ne peut évidemment se faire qu'avec une politique stricte de sécurité. N'importe qui ne doit pas pouvoir utiliser n'importe quelle ressource ou accéder aux données d'autres utilisateurs, et encore moins les modifier. La grille doit donc disposer d'outils souples et fiables pour gérer l'identification des utilisateurs ainsi que les droits qui leur sont associés.

– On peut imaginer vouloir utiliser des applications interactives sur la grille avec lesquelles un utilisateur voire même un groupe d'utilisateurs pourraient interagir directement. La grille doit permettre un niveau de performances tel qu'elle sera à même de satisfaire les contraintes temps réel inhérentes à ce type d'applications.

2.2 L'architecture de Globus Toolkit

Comme nous l'avons vu, les services Web sont particulièrement bien adaptés à une utilisation sur Internet. Il reste toutefois un problème de taille à régler avant de pouvoir les utiliser dans le cadre des grilles : la persistance des données. En effet, les services Web ne permettent pas de conserver un contexte d'exécution entre deux invocations successives d'une méthode. Si cela ne pose pas de problème dans le cadre de service simple comme celui présenté en exemple, cela devient beaucoup plus problématique si l'on désire construire une application pour une grille. En effet, imaginons que l'on soumet un job pour exécution via un service Web. Il serait légitime d'espérer pouvoir interroger le serveur afin de connaitre l'évolution du job au fil du temps. Pour ce faire, le service Web doit pouvoir conserver des informations au fil des appels successifs du client.

Pour remédier à ce problème, Globus a développé le *Web Services 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 stockent 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. Ainsi, lors de l'invocation d'une méthode, il suffit de passer au service Web l'identifiant de la ressource que l'on désire utiliser et celui-ci pourra retrouver les valeurs des variables qu'elle contient.

L'association d'un service Web avec une ressource est appelée une WS-Ressource. Le WSRF définit une série de spécifications permettant la manipulation aisée des WS Ressource évitant d'avoir à gérer manuellement l'identifiant de la ressource avec laquelle on travaille.

La figure 3.2 illustre un service possédant trois ressources de type fichier, chaque fichier possédant trois propriétés (appelées *WS-ResourceProperties*).

En plus des différents services que nous allons présenter dans la suite, le *toolkit* Globus offre, comme l'illustre la figure 3.3, des conteneurs permettant aux

développeurs d'implémenter aisément leurs propres services en langage C, Java ou Python. Ils peuvent ainsi bénéficier des facilités offertes par Globus en matière de sécurité, persistance de données (via les WS- ressources) ou découverte de services. Globus fournit également un ensemble de bibliothèques permettant aux clients d'invoquer facilement, toujours en C, Java ou Python, les fonctionnalités des services que ce soient ceux fournis avec Globus ou ceux développés par l'utilisateur. Enfin, le *toolkit* dispose d'une série d'interfaces en lignes de commande permettant d'interagir directement avec les différents services.

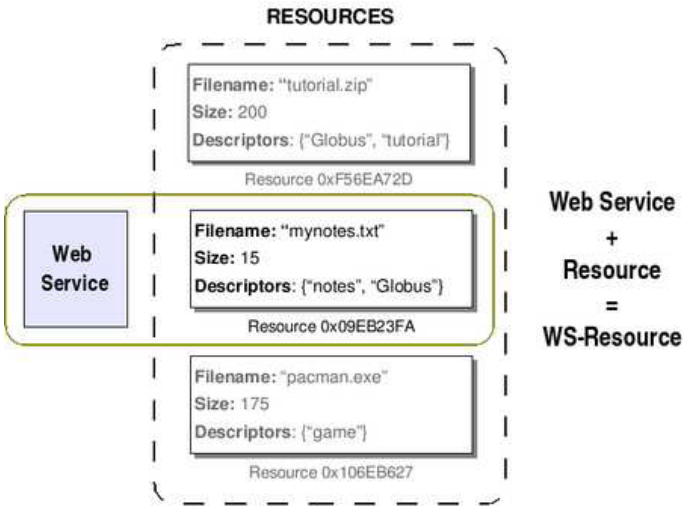


Figure 3.2. WS-Ressource

Voyons maintenant les quatre grandes classes de services offerts par le *toolkit* Globus.

2.2.1 Sécurité

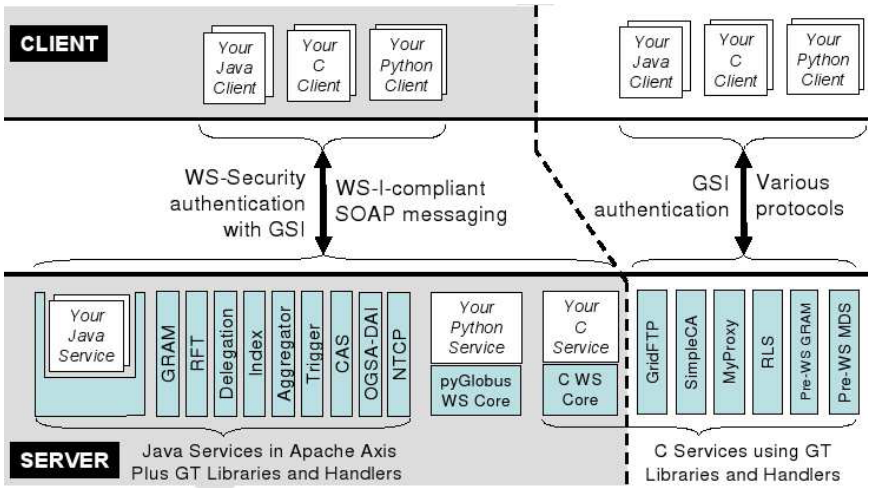


Figure 3.3. Vue Client / Serveur de l'architecture de Globus 4

Globus a développé un ensemble de protocoles nommés *Grid Security Infrastructure* (GSI) basés sur une architecture à chiffrement asymétrique (paire clé publique/clé privée) (cf. Figure 3.4). Ils sont utilisés pour l'authentification, la confidentialité des communications ainsi que pour l'identification.

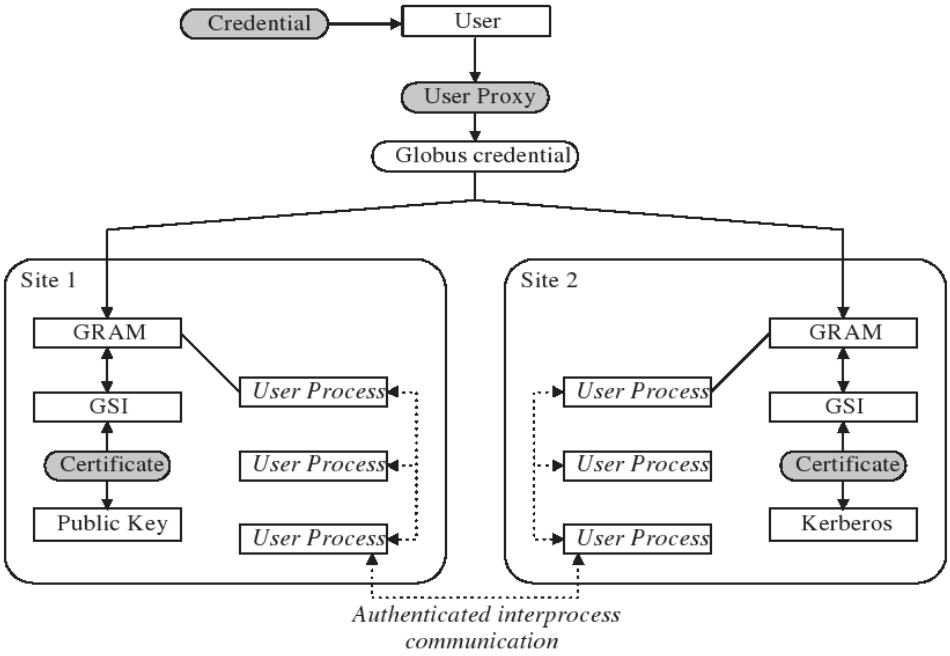


Figure 3.4. Architecture du GSI

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 Kerberos4 par exemple). . . Les utilisateurs et machines sont identifiés à l'aide d'un certificat au format X.509. Ce dernier contient :

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

Une clé privée, protégée par un mot de passe, est également associée au certificat.

Pour répondre aux problèmes de délégation de droits et d'identification unique, on utilise la notion de *proxy d'authentification*. Un *proxy* est très similaire à un certificat X.509 à quelques différences près.

- Il est associé à une nouvelle paire de clés (privée et publique).

- L'identité du propriétaire est quelque peu modifiée pour indiquer que le certificat est un *proxy* et non un certificat « classique ».

- Ce nouveau certificat est signé par l'utilisateur lui-même et non plus par l'autorité de certification.

- Il a une durée de validité assez courte, typiquement de l'ordre d'une dizaine d'heures.

La clé privée du *proxy* ne doit être connue que par les deux entités se l'échangeant mais, de par la durée de vie limitée de celui-ci, elle est moins sensible que celle du certificat et n'est donc pas protégée par mot de passe.

Imaginons maintenant que Mimi veuille s'identifier auprès de Sidali afin de, par exemple, lui demander d'effectuer une tâche. Mimi va alors générer un *proxy* associé à son certificat ; pour ce faire, elle aura besoin d'entrer son mot de passe afin de débloquer la clé privée de son certificat. La paire de clés associées à ce *proxy* sera choisie d'un commun accord entre Sidali et Mimi.

Ceci fait, Mimi envoie à Sidali son certificat ainsi que son *proxy* fraichement créé. Afin de s'assurer que Mimi est bien celle qu'elle prétend être, Sidali va d'abord vérifier, à l'aide de la clé publique du certificat de Mimi, que le *proxy* a bien été signé par cette dernière. Ensuite, il vérifie que la signature du certificat a bien été effectuée par l'autorité de confiance (Mimi et Sidali s'étant mis d'accord au préalable sur une autorité de confiance, ils possèdent tous les deux la clé publique de cette autorité).

Imaginons maintenant que la tâche que Sidali est en train d'effectuer pour Mimi est telle qu'il aimerait déléguer une sous-tâche de celle-ci à Rafik. Rafik ne connait pas Sidali et n'a donc aucune envie de travailler pour lui mais est, en revanche, un bon ami de Mimi et est prêt à l'aider avec plaisir. Sidali doit donc prouver à Rafik que la tâche qu'il lui demande d'exécuter est faite pour le compte de Mimi.

Pour ce faire, il suffit que Sidali transfère le *proxy* à Rafik. Celui-ci étant signé par Mimi et, Sidali possédant la clé privée associée au *proxy*, Rafik est assuré que Mimi a bien autorisé Sidali à travailler pour son compte.

Parmi les inconvénients de cette méthode, signalons le fait que Sidali peut potentiellement abuser du *proxy* et l'utiliser à d'autres fins que celles de servir Mimi. Pour limiter ce problème, on peut restreindre le champ d'action du *proxy* en spécifiant par exemple le type de ressource pour lequel il peut être utilisé. De plus, le *proxy* ayant une durée de validité relativement réduite, Sidali ne pourra pas en abuser bien longtemps.

Sans *proxy*, l'utilisateur devrait s'identifier auprès de chaque ressource utilisée alors qu'ici il suffit de le faire une fois lors de la génération du *proxy*.

Signalons enfin que le *proxy* ayant une durée de validité limitée, les conséquences au niveau sécurité sont moins problématiques si celui-ci est compromis que si l'on utilisait directement le certificat de l'utilisateur, qui possède généralement une durée de vie beaucoup plus longue (typiquement un an).

La GSI de Globus fournit également une interface de contrôle nommée *Generic Authorization and Access* (GAA) permettant au propriétaire d'une ressource de définir sa politique de sécurité locale.

2.2.2 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 Transfer Protocol* (FTP) nommée *GridFTP*. Elle y ajoute entre autres des fonctionnalités de la couche connectivité en terme de sécurité (authentification via la GSI) ou encore la gestion de transferts en parallèles sur plusieurs canaux pour accélérer les débits.

La gestion des données distribuées est assurée, entre autres, par le *Replica Location Service* (RLS) qui enregistre dans un catalogue, les différentes localisations d'un fichier afin d'en retrouver les réplicas. 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.

Enfin, il est important de noter que les fichiers stockés sur la grille suivent le principe du « write once, read many ».Ce qui signifie qu'une fois créé, un fichier est accédé uniquement en lecture et plus en écriture. Cela évite ainsi d'avoir à gérer la synchronisation d'un fichier avec ses différents réplicas et évite des problèmes en cas d'accès simultané à un même fichier. En pratique, cette limitation n'est pas aussi contraignante que ce que l'on pourrait être tenté de croire au premier abord. En effet, la majorité des données stockées sur une grille sont typiquement des relevés physiques ou des résultats expérimentaux (comme dans le cas du LHC par exemple). Dès lors, modifier ces données expérimentales n'a finalement pas beaucoup de sens.

2.2.3 Gestion des jobs

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. . . Elle permet également de recevoir des notifications lors des changements de statuts du processus.

GRAM peut s'interfacer avec plusieurs systèmes de gestion de processus allant des simples *fork* UNIX à des ordonnanceurs de jobs 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'ordonnanceurs existants. Le cas du *fork* Unix est un peu particulier, le job étant alors simplement exécuté en créant un processus sur la machine où il a été soumis.

Comme pour les fermes, les processus d'une grille peuvent être amenés à devoir communiquer entre eux lors de leur exécution (dans le cas de parallélisme fonctionnel). Si ceux-ci sont exécutés sur un même cluster, il n'y a, a priori, pas de problème, les processus pouvant réutiliser les fonctionnalités de la ferme pour ce faire.

Cela devient toutefois beaucoup plus complexe lorsqu'une même tâche possède des processus répartis sur différents sites de la grille. Bien que ce type d'applications est encore peu courant, il est indispensable de rendre, à terme, la communication entre ces processus possible si l'on désire pouvoir faire face à la complexité croissante des problèmes et profiter de toute la puissance de la grille pour les résoudre.

Le projet MPICH-G2 est une implémentation du standard MPI intégrant le support des grilles. Il utilise différents services de Globus et permet d'exécuter des applications MPI en convertissant automatiquement les communications entre processus en transferts TCP pour les communications inter-site ou en utilisant l'implémentation MPI de la ferme, s'il y en a une disponible, pour les communications intra-site.

Toutefois, si les processus sont situés sur des sites distincts, les temps de latences lors des communications sont bien évidemment relativement conséquents par rapport à une exécution sur une même ferme. Il peut alors devenir important d'en tenir compte lorsqu'on conçoit ses algorithmes ou applications afin de ne pas subir une trop grande dégradation des performances.

2.2.4 Découverte de services et ressources

La dernière grande catégorie de fonctionnalités que fournit le *toolkit* concerne la surveillance ainsi que 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. La finalité de cette collecte d'informations peut être soit la découverte des infrastructures qui sont à notre disposition, soit la surveillance de ces dernières.

Le composant Globus en charge de ces tâches est nommé *Monitoring and Discovery System* (MDS). C'est, par exemple, ce service qui sera utilisé par GRAM, que nous avons

présenté à la section précédente, pour trouver les ressources pouvant satisfaire les contraintes d'un job.

Que ce soit la surveillance ou la recherche de ressources, ces deux tâches nécessitent la collecte d'informations pouvant émaner de multiples sources. Pour ce faire, MDS dispose de services d'agrégation en charge de la collecte de données auprès des différentes ressources enregistrées. Ils fournissent également des interfaces et outils en ligne de commande permettant à l'utilisateur d'accéder et d'effectuer des recherches parmi ces informations.

2.3 Les différentes approches logicielles pour exploiter une grille

De par leur nature, les grilles de calcul sont plus complexes à utiliser qu'une station de travail ou qu'une grappe. Cela est dû à la distribution des nœuds qui peuvent être situés dans des domaines d'administration différents. De plus, de l'échelle des grilles découlent deux caractéristiques principales des nœuds qui sont la volatilité et l'hétérogénéité. De nombreux projets se sont intéressés à la gestion des ressources d'une grille et ont proposé des systèmes, appelés intergiciels, pour en simplifier l'utilisation.

Dans cette partie, nous présentons différents types d'intergiciels, classés par rapport aux fonctionnalités qu'ils offrent.

2.3.1 Intergiciels de communication

Les intergiciels de communication visent à simplifier l'accès à des ressources réparties sur une grille en fournissant un moyen aux ressources de communiquer entre elles.

JXTA de Sun Microsystems est un ensemble de protocoles pair-à-pair génériques qui permettent à n'importe quel nœud connecté au réseau de communiquer et de collaborer.

NaradaBrokering a pour objectif de fournir un environnement de communication unifié pour différents paradigmes de programmation tels que les services Internet (web services), les services de grille (grid services) et les modèles pair-à-pair. NaradaBrokering offre deux méthodes de communication qui sont l'échange de messages et la publication/souscription.

PadicoTM est un environnement de communication à hautes performances qui permet de multiplexer l'accès au réseau depuis plusieurs intergiciels. PadicoTM optimise les communications lorsque cela est possible en profitant d'un accès direct aux cartes de réseaux rapides et en ajustant les paramètres de la couche de transport. De plus, PadicoTM permet l'exécution d'applications réparties de part et d'autre de coupe-feu ou de réseaux isolés accessibles uniquement via un nœud frontal.

2.3.2 Intergiciels de gestion globale des ressources

Dans cette partie, nous présentons les approches logicielles pour gérer de façon globale les ressources d'une grille. Au minimum, ces intergiciels fournissent des mécanismes permettant de soumettre une application sur la grille, de trouver des ressources, de déployer une application et de récupérer les résultats produits.

a. Boîte à outils

Comme on a vu précédemment (section 2.2) le projet Globus a pour objectif de construire une boîte à outils permettant d'utiliser de façon sécurisée une grille dont les ressources sont réparties à travers plusieurs domaines d'administration. Dans sa version 4, Globus est la mise en œuvre de référence de la norme OGSA (Open Grid Services Architecture) qui est proposée par l'Open Grid Forum (OGF) et dont l'objectif est de définir une interface qui permet l'interopérabilité entre les ressources d'une grille. Voyons les principaux constituants de la boîte à outils Globus. GSI (Globus Security Infrastructure) est une infrastructure permettant d'authentifier les utilisateurs d'une organisation virtuelle qui est fondée sur l'utilisation de certificats X.509. GSI constitue la fondation de tous les autres services. MDS (Monitoring and Discovery Service) constitue le système d'information de Globus où toutes les ressources et leur état sont enregistrés. C'est en faisant une requête RSL (Resource Specification Language) que les utilisateurs peuvent interroger le MDS afin d'obtenir des ressources. GRAM (Globus Resource Allocation Manager) permet de soumettre des tâches sur des ressources distantes et d'en surveiller l'exécution. DUROC (Dynamically Updated Request Online Co-allocator) est un service au-dessus de GRAM qui permet la soumission simultanée de multiples tâches, c'est-à-dire, la co-allocation de ressources. GridFTP (Grid File Transport Protocol) et GASS (Global Access to Secondary Storage) permettent respectivement de gérer les transferts de fichiers entre les ressources et de gérer les répliques de données pour optimiser les temps d'accès. La nature de l'architecture de Globus en fait un système très polyvalent mais complexe à utiliser. Globus ne fournit que les briques de base et c'est souvent à l'utilisateur qu'est laissé le soin d'adaptation pour des utilisations spécifiques comme par exemple la mise en œuvre d'un ordonnanceur de haut niveau fondé sur un modèle économique. Toutefois, de par sa position privilégiée à l'OGF, Globus est presque considéré comme un standard de facto et constitue une fondation pour de nombreux travaux. Nous pouvons citer par exemple gLite, NorduGrid, GridLab ou encore GridBus qui sont des extensions de Globus visant à fournir des fonctionnalités de haut niveau pour l'exploitation des ressources d'une grille.

b. Systèmes intégrés

D'autres systèmes, plus simples à utiliser et disposant de fonctionnalités de plus haut niveau que Globus, permettent de prendre complètement en charge la gestion des ressources d'une grille. Ces systèmes ont pour objectif de masquer la distribution des

ressources à l'utilisateur en offrant une vue unique de la grille. Ainsi, les fonctionnalités d'allocation de ressources, de sécurité, d'interface utilisateur, de transfert de fichiers sont au moins partiellement intégrées au sein de ces systèmes.

Systèmes génériques : Condor est un des premiers intergiciels intégrés qui est né à la fin des années 1980. Condor vise les architectures de type réseau de stations de travail et a pour objectif de récupérer les cycles inutilisés des nœuds de calcul. Les stations de travail forment un Condor Pool et un gestionnaire central est chargé de distribuer les tâches sur les stations. Les utilisateurs peuvent spécifier la nature des ressources requises pour leurs tâches grâce au système ClassAdd (Classified Advertisements). Condor utilise un mécanisme de bac à sable pour l'exécution des applications afin d'éviter qu'un code malveillant ne compromette les nœuds utilisés. De plus, lorsqu'un nœud mobilisé par des tâches Condor est de nouveau utilisé par son propriétaire, les tâches qu'il exécutait sont déplacées sur un autre nœud via un mécanisme de point de reprise. Diverses évolutions de Condor ont été développées. Nous pouvons citer Flocking Condor qui permet de fédérer plusieurs Condor Pools. Dans la littérature, les auteurs proposent une évolution du Flocking Condor en organisant automatiquement les Condor Pools dans un réseau pair-à-pair structuré. Linger-Longer étend Condor pour supporter le vol de cycle avec un grain fin. Le problème avec le vol de cycle de Condor est que, dès qu'une utilisation même faible de la machine est détectée, les tâches sont suspendues. L'idée ici est de ne pas systématiquement suspendre ces tâches, mais plutôt de baisser leur priorité afin de ne pas gêner l'utilisateur.

Dans la lignée de Condor, Entropia est un système permettant de fédérer les nœuds de type poste de l'ingénieur d'une entreprise. Entropia permet d'exécuter des tâches sur les nœuds de manière sécurisée pour les consommateurs et pour les fournisseurs de ressources. Pour cela Entropia utilise un bac à sable pour protéger le fournisseur de ressources et un mécanisme de cryptage de fichier pour protéger le consommateur de ressources.

UNICORE est un projet européen dont l'objectif est la fédération de différents centres de calcul en Europe. UNICORE possède des fonctionnalités similaires à celles de Globus (sécurité, gestion des ressources, système d'information et gestion des fichiers) mais propose une interface d'utilisation conviviale qui intègre des fonctionnalités de haut niveau.

Legion a pour objectif d'offrir une vision unique des ressources de la grille. Pour cela, Legion propose un modèle objet où tout dans la grille (ressources de calcul, applications, fichiers) est représenté par un objet. Legion spécifie un certain nombre d'objets de bas niveau qui peuvent être modifiés ou étendus par les développeurs. Il est par exemple possible d'ajouter un support pour un nouveau gestionnaire de traitement par lots en dérivant la classe HostClass et en mettant en œuvre les nouvelles fonctionnalités souhaitées.

Vishwa est un intergiciel intégré pour grille fondé sur des réseaux pair-à-pair qui permettent de tolérer la dynamicité des nœuds d'une grille. Vishwa fournit un composant pour la communication entre plusieurs tâches qui est fondé sur Distributed Pipes.

PUNCH (Purdue University Network Computing Hub) est un projet de Purdue University dont l'objectif est d'utiliser la grille uniquement avec un navigateur internet. Via ce navigateur Internet, PUNCH fournit un moyen de déployer et d'utiliser des services sur la grille. PUNCH dispose d'un annuaire nommé ActYP (Active Yellow Pages) qui permet de recenser les services déployés et qui possède de bonnes capacités de passage à l'échelle grâce à l'utilisation d'une méthode de réplication active des données.

Adage (Automatic Deployment of Applications in a Grid Environment) est un projet développé au sein de l'équipe-projet PARIS de l'INRIA dont l'objectif est de fournir un mécanisme de déploiement pour les grilles de calcul. Adage propose un formalisme abstrait pour la description d'une application qui prend en compte la topologie du réseau et les applications fondées sur le modèle composant.

Zorilla est un projet de Vrije Universiteit fondé sur les réseaux logiques pair-à-pair. Zorilla est un intergiciel complètement distribué qui possède toutes les fonctionnalités requises, comme l'ordonnancement ou le transfert de fichiers, pour l'exécution d'applications sur une grille. Grâce à son mécanisme d'ordonnancement appelé flooding scheduling, Zorilla est capable de tenir compte de la localité des ressources dans l'allocation.

Systèmes spécifiques : Un certain nombre de projets sont spécifiques à un type d'application donné. Nous pouvons citer OptimalGrids qui est un intergiciel dédié au déploiement d'applications de type éléments finis. OptimalGrids permet de paralléliser automatiquement un problème en fonction des ressources disponibles, en fonction d'indications présentes dans une base de connaissance et en tenant compte du ratio calcul/communications. P3 (Personal Power Plant) est un intergiciel fondé sur JXTA dont l'objectif est de fournir une abstraction pour simplifier la programmation des applications parallèles fondées sur le passage de messages et les applications de type maître/travailleurs. Organic Grid est un intergiciel permettant d'exécuter des applications composées de tâches indépendantes. Organic Grid reprend la métaphore de la colonie de fourmis où chaque nœud, représenté par un agent, est capable de s'adapter à son environnement et ne possède qu'une faible connaissance de l'environnement global.

c. Architectures cibles des intergiciels de gestion globale des ressources

Le tableau 3.1 présente les architectures de grilles pour lesquels sont conçus les intergiciels de gestion globale des ressources qui ont été étudiés dans cette partie.

Nous pouvons voir que les intergiciels de gestion globale des ressources ne sont pas adaptés à toutes les architectures de grilles. Néanmoins, les projets Adage et Globus peuvent être utilisés dans tous les contextes de grille mais ce sont les projets qui offrent le moins de fonctionnalités de haut niveau. Nous voyons dans la partie 2.1.5.4 que de nombreuses extensions à Globus qui concernent en particulier l'ordonnancement permettent d'augmenter le niveau de fonctionnalités en conservant sa polyvalence initiale.

	Réseau de stations	**Fédération de grappes**	**Calcul P2P**	**Hétérogène**
Adage	Oui	Oui	Oui	Oui
Condor	Oui	Non	Non	Non
Condor+Flocking	Oui	Oui	Non	Non
Entropia	Oui	Non	Non	Non
Globus	Oui	Oui	Oui	Oui
Legion	Oui	Oui	Non	Oui
PUNCH	Oui	Non	Non	Non
UNICORE	Non	Oui	Non	Non
Vishwa	Oui	Non	Oui	Non
Zorilla	Oui	Non	Oui	Non

Tableau 3.1. Architectures cibles des intergiciels de gestion globale des ressources

2.3.3 Systèmes d'exploitation pour grille

Afin de donner une vision aussi parfaite que possible d'un système à image unique, certains projets s'autorisent à modifier directement un système d'exploitation afin d'en étendre les fonctionnalités. Des propriétés uniques peuvent ainsi être obtenues comme une transparence complète pour l'exécution des applications ou encore un support très avancé pour les opérations de points de reprise. Cette approche très puissante est toutefois plus complexe à déployer car les fournisseurs de ressources d'une grille ne sont pas forcément prêts à changer les systèmes d'exploitation afin de ne pas perdre les propriétés requises pour les exécutions locales comme le contrôle d'admission des tâches ou les politiques de priorités pour l'utilisation des ressources.

MOSIX est un système à image unique pour grappe de calculateurs. Il permet d'effectuer de l'équilibrage de charge entre les nœuds en effectuant de la migration de processus. Une extension pour la grille permet de profiter de la puissance de calcul d'une fédération de grappes mais le passage à l'échelle reste limité par le mécanisme de deputy utilisé dans MOSIX pour rediriger les entrées/sorties des processus déplacés.

9Grid est fondé sur le système à image unique Plan9 qui possède les propriétés de passage à l'échelle requises pour l'utilisation sur une grille de type fédération de

grappes. 9Grid traite des problèmes d'authentification, de sécurité, de découverte de ressources et de gestion des données. 9Grid propose une solution pour l'interopérabilité avec d'autres intergiciels comme Globus en utilisant le protocole 9P. Il faut noter toutefois que Plan9 ne supporte pas nativement les applications du monde Linux ou UNIX.

XtreemOS est un projet européen dont l'objectif est d'étendre le système Linux pour supporter les organisations virtuelles des grilles. XtreemOS vise les grilles dynamiques de très grande taille quelle que soit leur architecture.

2.3.4 Ordonnanceurs

De nombreux travaux se sont concentrés sur l'aspect ordonnancement de tâches et s'appuient sur des outils existants pour les autres fonctionnalités comme le système d'information, le transfert de fichiers ou encore la réservation des ressources.

SmartNet est un projet dont l'objectif est l'ordonnancement des applications sur des ressources hétérogènes. Pour cela, SmartNet tient compte de la charge des machines mais aussi de leurs performances selon le type de calcul utilisé. SmartNet peut reposer sur Condor ou IBM LoadLeveler.

MyGrid est un système permettant l'exécution d'applications de type sac de tâches. L'architecture de MyGrid comprend un ordonnanceur et une interface générique pour divers types de nœuds tels que ceux utilisant le GRAM de Globus. Contrairement à beaucoup d'approches, MyGrid est centré sur l'utilisateur (user centric en anglais). C'est-à-dire que lorsqu'un utilisateur soumet une application, c'est la machine utilisée pour la soumission qui exécute l'ordonnanceur. Ourgrid est une extension de MyGrid fondée sur un modèle pair-à-pair pour partager des cycles au sein d'une organisation virtuelle. OurGrid permet de protéger les ressources utilisées en utilisant un mécanisme de bac à sable.

AppLeS (Application Level Scheduling) est un projet définissant le processus de création d'un ordonnanceur adaptable en proposant six étapes qui sont : la découverte de ressources, la sélection de ressources, la génération d'ordonnancements possibles, la sélection d'un ordonnancement, l'exécution de l'application et le ré-ordonnancement. Afin de permettre aux développeurs de ne pas systématiquement réécrire un ordonnanceur, AppLeS propose trois types d'ordonnanceurs qui sont APST (AppLeS Parameter Sweep Template) qui permet d'ordonnancer les applications paramétriques, AMWAT (AppLeS Master Worker Application Template) qui permet d'ordonnancer les applications de type maître/travailleur et SA (Supercomputer AppLes) qui permet d'ordonnancer les applications sur un supercalculateur à espace partagé. AppLeS repose sur les services fournis par des gestionnaires de ressources tels que Globus, Legion ou NetSolve.

Nimrod/G est un ordonnanceur destiné à l'exécution de simulations paramétriques sur grille. Nimrod/G propose un ordonnancement fondé sur un modèle économique où les fournisseurs de ressources définissent un prix et les clients payent pour une exécution. Nimrod/G est construit sur Globus, notamment pour les services MDS, GRAM et GASS.

Condor/G est une évolution du projet Condor. C'est un ordonnanceur fonctionnant au dessus de la boîte à outils Globus. L'objectif est de fédérer plusieurs domaines d'administration en utilisant les mécanismes d'authentification de Globus. Les domaines fédérés peuvent être des Condor pools ou d'autres gestionnaires de tâches comme PBS et sont accédés via le GRAM de Globus.

SPHINX est un ordonnanceur pour les grilles composées de ressources dynamiques. SPHINX propose des mécanismes d'ordonnancement avancés en utilisant les informations récupérées sur les ressources grâce à un système de supervision et en les stockant pour de futurs ordonnancements. Par exemple, si un nœud possède un nombre élevé de tâches qui ont été supprimées, il ne sera pas choisi en priorité. SPHINX s'appuie sur le système d'information MDS de Globus mais une interface générique a été définie pour fonctionner avec d'autres systèmes tels que Ganglia, MonALISA ou Hawkeye.

KOALA est un ordonnanceur pour la co-allocation de tâches fondé sur Globus (RSL, MDS, RLS, GRAM, GridFTP). KOALA possède des politiques d'ordonnancement permettant de minimiser les transferts de fichiers. Il permet ainsi de placer les composants applicatifs au plus près des fichiers d'entrée ou des répliques.

2.3.5 Intergiciels pour la programmation d'applications de grilles

Certains intergiciels sont dédiés à la programmation d'applications pour les grilles. Ils fournissent une API permettant la programmation d'applications selon un paradigme particulier qui abstrait la localisation des ressources.

Paradigme du passage de messages pour la grille : MPICH-G2 est une implémentation de MPICH qui permet de construire des applications pour la grille. MPICH-G2 est fondé sur Globus pour la sécurité et la soumission des tâches. MPICH-V propose aussi une implémentation de MPICH, cette fois en prenant en compte la volatilité des ressources. MPICH-V propose plusieurs protocoles de tolérance aux défaillances qui sont : l'enregistrement de messages pessimiste (MPICH-V1 et MPICH-V2), l'enregistrement de messages causal (MPICH-V/causal) et la coordination globale non-bloquante (MPICH-V/CL) et bloquante (MPICH-P/CL) suivant l'algorithme de Chandy-Lamport.

Paradigme de la mémoire virtuellement partagée pour la grille : Des intergiciels permettent de créer des applications pour grille fondées sur le concept de la mémoire

virtuellement partagée. C'est le cas par exemple des projets JuxMem, Mome et Vigne, tous trois développés dans l'équipe-projet PARIS de l'INRIA.

Paradigme composant pour la grille : ProActive est un environnement développé dans l'équipe-projet OASIS de l'INRIA pour le développement d'applications fondées sur le paradigme composant Fractal. ProActive offre des mécanismes de communication de groupe, de support à la mobilité pour déplacer des composants afin de réaliser de l'équilibrage de charge ou encore d'abstraction des ressources avec le concept de Virtual Node. ProActive utilise un système d'information externe qui peut être RMIregistry, le MDS de Globus, LDAP ou UDDI.

GridCCM, développé au sein de l'équipe-projet PARIS de l'INRIA, définit un modèle qui étend CCM avec la notion de composants parallèles. GridCCM est tout particulièrement adapté au couplage de codes où des codes parallèles communiquent entre eux.

DG-ADAJ est un intergiciel développé dans l'équipe PALOMA au LIFL permettant d'exécuter des applications utilisant le paradigme composant CCA sur une grille de stations de travail. DG-ADAJ donne une vision SIU de la grille et offre des fonctionnalités pour l'exécution de workflow et pour l'équilibrage de charge.

Paradigme du calcul global : Le paradigme du calcul global, ou paradigme du calcul pair à-pair, consiste à utiliser un nombre de machines pouvant aller jusqu'à plusieurs millions réparties sur Internet pour l'exécution d'une application composée d'un grand nombre de tâches indépendantes.

BOINC (Berkeley Open Infrastructure for Network Computing) est un projet de l'UC Berkeley qui vise à fournir une infrastructure pour le développement d'applications distribuées qui utilisent une grille de type calcul pair-à-pair. BOINC est particulièrement adapté aux applications de type maître/travailleurs, fournit des facilités pour la sécurité, la tolérance aux défaillances et l'ordonnancement. Un exemple d'application portée sur BOINC est SETI@home dont l'objectif est d'analyser des données obtenues avec des radio-télescopes en vue de détecter une forme de vie extra-terrestre. En juin 2007, l'application SETI@home était exécutée sur un peu plus d'un million et demi de machines.

XtremWeb est un projet du LRI qui permet, comme BOINC, de développer des applications de calcul global. XtremWeb est adapté au paradigme maître/travailleurs et fournit une abstraction du concept d'appel de procédure à distance et propose des interfaces pour JavaRMI et XML-RPC. Dans la littérature, les auteurs proposent d'utiliser XtremWeb pour fédérer des Condor Pools. Dans ce cas, un Condor Pool est vu comme un travailleur pour XtremWeb.

HiPoP (Higly distributed Platform Of comPuting) est une plate-forme de calcul global développée au LIFC. HiPoP permet d'exécuter des applications Java décrites par un graphe orienté acyclique.

Paradigme ASP : Le paradigme ASP (Application Service Provider) consiste à utiliser un certain nombre de serveurs répartis sur des ressources et dédiés à la résolution de problèmes numériques.

Les intergiciels de type ASP masquent à l'utilisateur la localisation et le type des ressources et fournissent une API souvent fondée sur un modèle d'appel de procédures distantes pour le développement d'applications. Nous pouvons citer quelques exemples d'ASP : DIET (Distributed Interactive Engineering Toolbox), Ninf/G, AROMA (scAlable ResOurces Manager and wAtcher) ou encore GridSolve.

2.4 Adoption de l'intergiciel Globus

Nous pouvons noter que de nombreux travaux sont fondés sur Globus. La raison principale est que Globus implémente les standards proposés d'OGF, ce qui garantit l'interopérabilité entre les différents travaux. Pour cela on a décidé d'opter pour Globus.

Globus ne peut être défini sur le même plan que les autres projets de recherche. C'est plus que cela puisque ses concepteurs le pensent en tant qu'*écosystème* de composants et outils qui interopèrent ensemble. Le projet est constitué : d'une *communauté* d'utilisateurs et de développeurs open source centrés sur les thèmes du calcul distribué ; d'organisations virtuelles ; et d'une fédération de ressources.

La partie *software* est constituée du Globus Toolkit qui comporte un ensemble de bibliothèques et de programmes qui répondent aux problèmes rencontrés lorsque l'on conçoit des systèmes distribués. Globus est aussi l'*infrastructure* qui supporte la communauté : répertoires de codes, système de débogage, listes, etc... De cet ensemble doit émerger des solutions, des savoir-faire classiques sur lesquels s'appuyer pour faciliter la conception des futures applications sur Grille.

La partie concrète et visible pour les utilisateurs reste principalement le Globus Toolkit. Il fournit une variété de services allant d'implémentations de services tels que la gestion des ressources, la transmission des données et la découverte de services jusqu'aux outils pour en construire de nouveaux : Web Services (Java, C, Python). Globus fournit une infrastructure de sécurité pour l'authentification, l'identification, l'autorisation d'accès. Enfin il fournit aux clients à la fois des API (dans différents langages) ainsi que des programmes en ligne de commande pour accéder aux services Globus.

La dernière version du Globus Toolkit est la numéro quatre (GT4). Cette dernière fait un usage extensif des services web afin de définir ses interfaces et structurer ses

composants. Leurs protocoles sont adaptés pour les interactions faiblement couplées que certains argumentent comme préférables pour concevoir des systèmes distribués robustes. Leur but est de concevoir des architectures orientées services, structurées en services communicants décrits de manière uniforme.

Finalement, il n'existe pas de travaux qui concilient généricité et fonctionnalités de haut niveau. Seule la combinaison de Globus et d'ordonnanceurs permet d'atteindre cet objectif, au prix toutefois d'une complexité de déploiement des différents intergiciels sur les nœuds de la grille.

Conclusion

Nous pouvons dire dans ce chapitre que Globus Toolkit apporte un nouveau paradigme d'accessibilité aux ressources qui pallient aux limites de distribution web basée sur le modèle client/serveur. En effet, à la différence du web actuel, un environnement de Grid permet à chaque entité participante d'être à la fois client et serveur, ce qui constitue un atout majeur dans un environnement de formation (dans notre cas c'est la plateforme Sakai) où les participants doivent fortement collaborer. Les approches de distribution basées sur les services web sont limitées quant à la disponibilité, l'extensibilité et la distribution des ressources.

Après avoir adopté la plateforme Sakai et l'intergiciel Globus Toolkit (GT4), le prochain chapitre va présenter notre étude de cas qui propose une intégration de ces deux dernières plateformes.

Chapitre IV

La nouvelle plateforme collaborative proposée

Introduction

Pour valider notre approche nous avons mis en place dans ce chapitre un prototype d'une plateforme collaborative. Dans notre cas, la communauté virtuelle est constituée d'une collection dynamique d'ingénieurs et de techniciens spécialistes en informatique situés à des endroits différents, dont l'objectif est le déploiement collaboratif d'un service Grid (Datamining Service) en utilisant l'intergiciel Globus. Cette approche contient deux but incontournable : Le premier but est d'exploiter la boite à outil Globus Toolkit (Dans notre cas c'est la version GT4) en collaboration, en tirant profit de ses quatre principaux services qui sont : la découverte d'information avec le service MDS (monitoring and discovery System), la gestion des jobs avec le service GRAM (Grid Resource Allocation Manager) et le transfert de fichiers avec le service GridFTP (Grid File Transfer Protocol), sur la base d'une infrastructure de sécurité très fiable qui est GSI (Grid Security Infrastructure). Pour valider que ceci est réalisable on a effectué trois tests de soumission de jobs : d'un job existant sur la machine locale, d'un job existant sur un nœud distant en utilisant le service GSIFTP et d'un job distant sur autre nœud distant ; Le second but de cette approche est de remplacer les ingénieurs en leurs absences en répondant aux questions fréquemment posées par les techniciens.

1. L'architecture proposée

La Figure 4.1 présente le fonctionnement général de notre environnement de collaboration. Il est essentiellement composé de quatre couches : ressources (réseaux et matériels), OGSA (middleware), GLS (collaboration) et la couche de bureau partagé (application web).

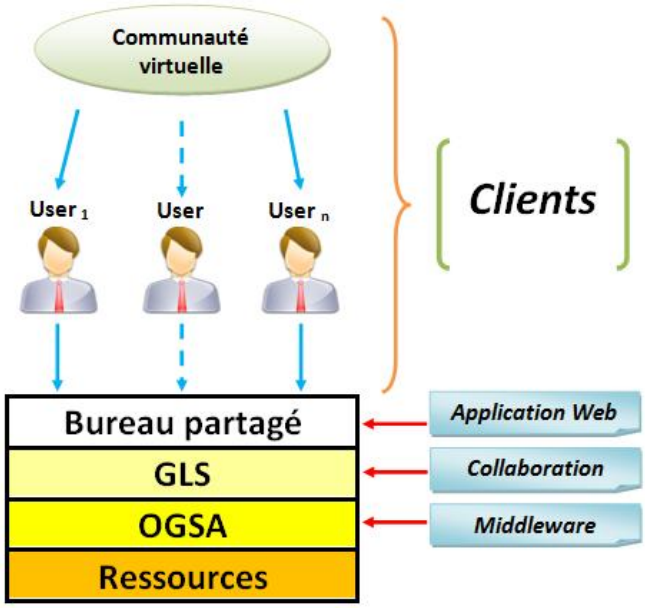


Figure 4.1. Schéma général de fonctionnement

1.1 La couche ressources

La couche ressources désigne l'ensemble des éléments utilisés dans une grille comme les stations de travail personnelles, les supercalculateurs, les centres de calculs, les systèmes de stockage de données, les catalogues de données…

Ces ressources agrégées gérées de façon coordonnée sont à même de satisfaire les besoin sans cesse croissants des applications. Besoins en puissance de calcul, quantité de mémoire, capacité de disque. Donc on peut distinguer deux types de ressources qui sont destinées chacun à son type d'applications :

Les ressources de traitement: Ressources de traitement sont destinées à des applications gourmandes en calcul, c'est-à-dire dont le goulot d'étranglement est situé au niveau du processeur.

Les ressources de stockage: Ressources de stockage sont destinées à des applications gourmandes en stockage sur disque.

En effet, les deux types de ressources élémentaires partagés dans GRID sont les ressources de traitements et de stockages, qui correspondent respectivement aux unités de traitements et de stockage. Ces unités peuvent être couplées dans un même HOST qui représente une association physique entre ces dernières.

1.2 La couche Middleware (OGSA)

L'OGSA est l'architecture de services de Grille. Elle est née du changement de paradigme dans le monde du génie logiciel c'est-à-dire du passage de la vision orientée-système vers la vision orientée-service et plus précisément de service à état. Cette approche, mieux connue sous le nom de Service Oriented Architecture (SOA), définit une architecture où les services sont à la base des communications. L'OGSA est issue du modèle d'architecture qui avait été proposé par deux chercheurs de Californie, Ian Foster et Carl Kesselman. L'idée était qu'une fois les ressources informatiques virtualisées, il est envisageable de mutualiser toutes ou partie de ces ressources pour générer des services. Initialement le service rendu se limitait au calcul intensif gourmand en ressource de traitement et de stockage. L'OGSA est une série de spécifications techniques par lesquelles on définit une infrastructure pour intégrer et gérer les services à l'intérieur d'une organisation virtuelle, distribuée et dynamique. Donc plus précisément, notre seconde couche Middleware a pour objectif de virtualiser des ressources. Dans cette couche, nous considérons la virtualisation des ressources comme moyen pour apporter portabilité et passerelles trans-paradigme de façon transparente à des services. Nous définissons la virtualisation comme suit : «La virtualisation consiste en l'émulation d'une ressource (dite virtuelle) sur une autre ressource (dite réelle). Les services utilisent la ressource virtuelle comme s'ils utilisaient une ressource réelle ; la ressource virtuelle simule le comportement de la ressource réelle». Ceci permet de faire utiliser par des services des ressources pour lesquelles ils ne sont pas conçus. La virtualisation se fait à l'aide d'un package de Services.

1.2.1 Partage des ressources

Le partage des ressources s'effectue grâce à deux processus (cf. figure 4.2): D'abord réalisé par un processus de Virtualisation pour cumuler globalement en entrée la capacité disponible et fournie par des unités de traitement et de stockage géographiquement distribuées. En sortie de ce processus, il y aura des ressources virtualisées, mais qui ne sont pas directement utilisables par les couches supérieures.

Figure 4.2. Processus de partage des ressources

Un autre processus dit Réification est indispensable. Il s'agit de restituer les ressources ainsi virtualisées dans des conteneurs de services qui en auront besoins. C'est ce qu'on appelle disponibilité des ressources à la demande. Les deux processus du partage des ressources sont accomplis au niveau du noyau GRID.

1.3 La couche collaboration (GLS)

Du point de vue architectural le Grid Learning Services (GLS) est juste au dessus de la couche service OGSA et considéré comme une extension de cette dernière. Il facilite : la collaboration, la diffusion de contenu et le partage d'objets d'apprentissage. C'est sur cette couche que s'appuie la couche applications pour mettre à disposition un environnement de collaboration aux utilisateurs. L'avantage du principe adopté c'est que chaque fois qu'une nouvelle fonctionnalité est ajoutée à la couche GLS, l'environnement applications en bénéficie.

Cette couche englobe toutes les instances dynamiques de services collaboratifs fournis par la plateforme (Service Collaboratif Synchrone & Asynchrone) à une session collaborative. Donc une caractéristique de service à état est indispensable pour déployer les deux types services collaboratifs. Une des principales raisons qui nous a motivé à utiliser les services GRID est la génération dynamique de services.

1.3.1 Services collaboratifs synchrones/asynchrones

Nous décrivons dans cette section les Services Collaboratifs Synchrone & Asynchrone qui visent à supporter les deux modes de collaboration, ce qui implique un environnement collaboratif flexible.

1.3.1.1 Service Collaboratif Synchrone

Un Service Collaboratif Synchrone est défini comme un service persistant capable de générer des services à état collaboratifs caractérisé par ses capacités à garder en mémoire les résultats d'actions antérieures comme par exemple un indicateur de présence lorsque des personnes se connectent sur un service de messagerie instantanée. Dès lors, il est nécessaire de dédier de la ressource à cet effet.

Les instances de ce type de service s'utilisent en temps réel entre des personnes situées à des endroits différents. La communication à distance entre les interlocuteurs se fait donc un peu comme au téléphone c'est-à-dire en temps réel. Le but de ce type de service est d'échanger des informations le plus rapidement possible pour décider de la suite des événements ou encore de réunir différentes personnes pour les informer en même temps de certaines informations à consigner (Exemple : réunions, cours). Parmi ces types de services collaboratifs synchrones nous citons : la vidéoconférence, la messagerie instantanée, l'utilisation de bureau partagé ou navigateur partagé.

1.3.1.2 Service Collaboratif Asynchrone

Nous appelons Service Collaboratif Asynchrone le service persistant qui peut produire des services à état utilisés à des moments différents dans le temps permettant une communication en continu sans tenir compte des obstacles liés à l'espace et au temps. (ex : La messagerie électronique, le forum de discussion, le transfert de fichiers et le Wiki).

Les services collaboratifs Synchrones et Asynchrones servent donc à :

- fournir des messages aux membres de la communauté pour leur permettre d'être immédiatement informé de la vie de la communauté virtuelle ;
- gérer les membres : ajouter, supprimer ou modifier les profils ;

- gérer de nouveaux services consacrés à la communauté ;
- instancier et gérer dynamiquement de nouveaux services de la couche supérieure.
- gérer les différentes sessions de collaborations;

1.3.2 Service à état

Un service à état est caractérisé par un générateur de service et des instances dynamiques de ce service. Ces services sont dits dynamiques dès lors qu'ils disposent de mécanismes qui limitent naturellement leurs instances dans le temps par un état final. Ainsi, un générateur de service peut émettre des instances qui utiliseront de la ressource et qui libéreront cette ressource en fin de vie (cf. figure 4.3). L'avantage immédiat est un usage efficace et bien réparti des ressources physiques en fonction de leur disponibilité. Cet aspect est crucial pour la mise à l'échelle d'applications réparties sur un système distribué.

Un service dynamique est appelé transitoire (transient) dans la terminologie Grid [19] par opposition à service persistant. Un service persistant ne dispose pas de mécanisme limitant la durée de ses instances. Il est à noter qu'un générateur de services dynamiques peut être lui même vu comme un service de type persistant.

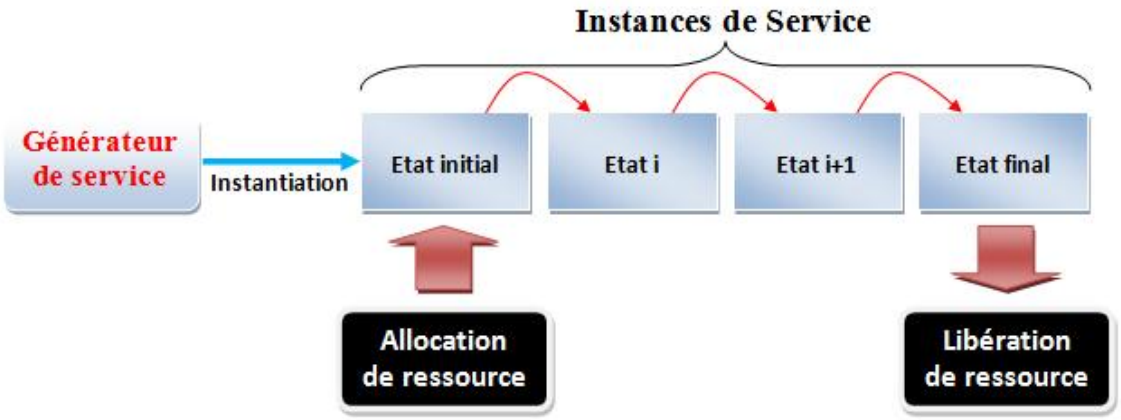


Figure 4.3. Modèle d'instanciation dynamique de services à état

1.4 La couche application web (Bureau partagé)

A la lumière du bilan des interactions identifiées lors d'une collaboration, il nous est rapidement apparu naturel d'opter pour le modèle du bureau comme interface homme-machine et d'étendre son usage à la manière d'une interface multimodale entres humains par une combinaison de bureaux imbriqués couplés à des canaux de services Grid.

Le Bureau partagé tel que nous le définissons ici peut être vu comme un environnement global de collaboration Flexible c-à-d il prend en compte les deux modes de collaboration en permettant à plusieurs utilisateurs (travaillant sur des ordinateurs différents et distants) d'utiliser de façon collaborative un service hébergé sur un seul ordinateur. Le bureau partagé est donc considéré comme un espace de collaboration et d'interaction immanent au sein d'une communauté virtuelle et contenant l'ensemble d'instances dynamiques des services de type collaboratif synchrone et

asynchrone accessibles à travers un terminal léger. Le tout est conçu de manière à rendre transparente à l'utilisateur toute la complexité de la technologie sous-jacente.

Notre bureau partagé d'Agora (cf. figure 4.4) s'appuie sur les services GLS pour organiser et faciliter les échanges entre les utilisateurs, il est accessible directement par un simple click sur un lien qui se trouve sur la plateforme de collaboration. Le bureau partagé est donc considéré comme un espace de collaboration qui tourne chez un seul utilisateur, mais il est rendu visible chez tous les utilisateurs distants qui se servent de ce partage.

La première tentative de proposition d'un bureau virtuel de GRID a été faite dans le cadre du projet d'IST-2001 GRIDLAB (Grid Application Toolkit and Testbed) [20]. Ce projet a développé GridSphere [92]. Ceci est basé sur ad hoc GUI et est moins susceptible d'être compatible avec tout genre d'applications. Une autre tentative est Entropia [22], un système développé en 2004 à l'université de la Californie. Entropia utilise la MV (Machine Virtuelle) pour mettre en évidence le bureau de GRID. Cependant, l'utilisation d'une MV pour chaque session d'utilisateurs consomme une quantité considérable de ressource du système.

Plus récemment, il a été proposé le collabureau comme un espace de collaboration à la puissance GRID dans [94], représenté comme une autre vision de GSD [23]. Cependant, le collabureau se limite à une description en mode synchrone.

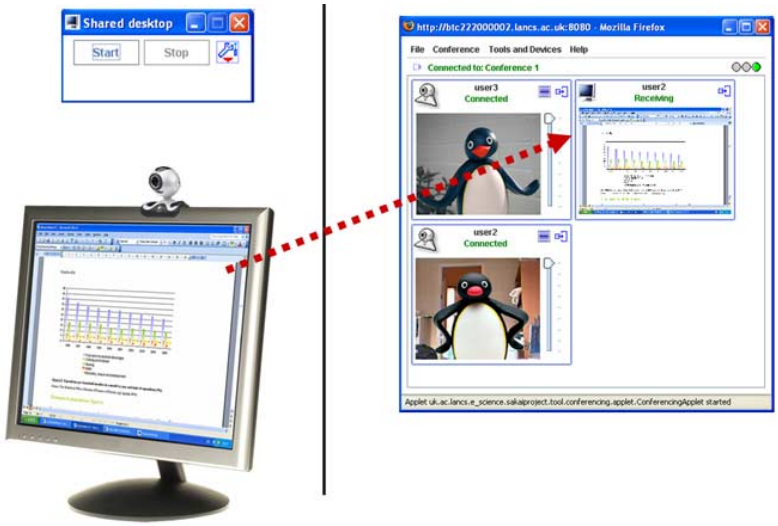


Figure 4.4. Le bureau partagé d'Agora

En effet, notre contribution à ce niveau s'appuie sur les deux modes de collaboration avec la possibilité d'accéder aux différents services de l'environnement collaboratif par un simple click (One-Click) à travers un terminal léger et sans aucune installation, configuration ou mise à jour logicielle sur la machine.

Le bureau partagé d'Agora maintenant est considéré comme un conteneur de services dont lequel s'exécute tout service sollicité et chaque service est symbolisé par des raccourcis (Icône par analogie au Microsoft Windows) sur le bureau qui peut être lancé en One Click. Par conséquent, on peut considérer les services à ce niveau comme des images des services collaboratifs de la couche inférieure directement accessibles par l'utilisateur de Grid.

2. Validation du fonctionnement de notre grille

Puisque la plateforme Sakai est devenue très riche en termes de services (services Web et services Grid), il est possible maintenant d'accéder aux services Grid de Globus (MDS, GRAM, GridFTP et GSI) et de développer des applications qui se basent sur ces derniers. Pour vérifier si ceci est faisable, nous avons préférer de faire des tests sur Globus en utilisant la couche application web qui est dans notre cas le Bureau partagé d'Agora. Ces tests consistent à ouvrir un Terminal Linux à travers ce Bureau (figure) et à soumissionner des Jobs (soumission locale ou distante).

2.1 Tests de soumission d'un job existant sur la machine locale

'globusrun-ws' :

- Soumet plusieurs requêtes simultanément.

- Organise les exécutables.

- Attend la Terminaison.

- Redirige vers stdout/stderr.

2.1.1 Test 1

Il s'agit d'un simple testde job sans nécessité d'écrire un programme descriptif de job. Dans cet exemple, nous lançons l'exécutable '/bin/touch' sur le nœud local (poste4).

```
[globus@poste4 ]$ globusrun-ws -submit –c /bin/touch touched_it
```

L'option '-c' prend comme premier argument une commande, dans notre cas c'est '/bin/touch' et le deuxième argument le fichier résultat, alors que l'option '-submit' est nécessaire pour la soumission de job, le résultat obtenu est :

```
Submitting job…Done.
Job ID : uuid : 4a92c06c-b371-11d9-9601-0002a5ad41e5
Termination time : 04/26/2009 11:43 GMT
Current job state : Active
Current job state : CleanUp
Current job state : Done
Destroying job… Done
```

Le résultat réside normalement sous le répertoire où le job est exécuté, dans notre cas c'est '/home/globus'.

2.1.2 Test 2

Il consiste à exécuter le job précédent sur un nœud distant. Pour utiliser les ressources distantes nous devons appeler son container qui se traduit par l'option '-F' lors du lancement de job. L'option '-c' est présente parce qu'il s'agit d'une commande '/bin/touch'.

```
[globus@poste4 ]$ globusrun-ws –submit -F
```

```
http://poste3.an.lri:8443/wsrf/services/ManagedJobFactoryService
-c /bin/touch touched
Submitting job…Done.
...
```

Le résultat est :

```
[globus@poste3 ]$ ls -1 touched
-rw-r-r- 1 globus globus 0 avr 26 11:50 touched
```

2.1.3 Test 3

Nous éditons un descripteur d'exécution d'un job c'est un fichier XML. Le contenu du fichier est :

```
[globus@poste4 ]$ cat test-job.xml
<job>
<executable>/bin/echo</executable>
<argument>this is an example_String</argument>
<argument>Globus was here</argument>
<stdout>$/home/globus/stdout</stdout>
</job>
```

Puis nous lançons la commande de soumission de job avec l'option '-f' qui prend en argument le fichier XML.

```
[globus@poste4 ]$ globusrun-ws –submit -f test-job.xml
Submitting job…Done.
…
```

Le contenu du fichier 'stdout' est :

```
[globus@poste4 ]$ ls -l stdout
-rw-r-r- 1 globus 5 avr 27 10:32 touched
[globus@poste4 ]$ cat stdout
Hello
```

2.1.4 Test 4

Il s'agit de lancer un job avec plusieurs arguments:

```
 [globus@poste4 ]$ cat test1-job.xml
...
<argument>12</argument>
<argument>abc</argument>
<argument>34</argument>
<argument>this is an exemple_String</argument>
<argument>Globus was here</argument>
....
```

Le résultat après lancement de la commande est l'obtention du fichier 'stdout1':

```
[globus@poste4 ]$ globusrun-ws –submit -f test1-job.xml
Submitting job…Done
…
```

```
[globus@poste4 ]$ cat stdout1
12 abc 34 this is an exemple_String globus was here
```

2.1.5 Test 5

Il s'agit de lancer un job en mode 'batch' afin de libérer le terminal de l'utilisateur. Le lancement du job en mode 'batch' se fait par l'option '-batch' ainsi que l'option '-o' dont le nom qui suit sert comme paramètre de récupération du résultat.

```
[globus@poste4 ]$ globusrun-ws –submit –batch -o job_epr -f testfils.xml
Submitting job…Done
Job ID: uuid: c6aecb80-ec16-11da-9f17-0010b5933618
Termination time : 05/26/2009 17 :49 GMT
```

La récupération du résultat se fait par la commande suivante:

```
[globus@poste4 ]$ globusrun-ws –monitor -j job_epr
Current job state: Done
Requesting original job description…Done
Destroying job…Done
```

2.2 Tests de soumission de jobs existants sur un nœud distant

2.2.1 Utilisation du service "GSIFTP"

Dans ce cas le fichier exécutable se trouve sur un autre nœud distant

```
[globus@poste4 ]$ cat test_gsiftp.xml
…
<transfer>
<sourceUrl>gsiftp ://poste3.an.lri :2811/bin/echo</sourceUrl>
<destinationURL>file :///home/globus/my_echo</destinationUrl>
</transfer>
…
```

Nous pouvons aussi transférer notre fichier résultat vers un autre nœud, avec le même service, par exemple:

```
[globus@poste4 ]$ cat test_gsiftp1.xml
```

```
…
<transfer>
<sourceUrl>gsiftp ://poste3an.lri :2811/bin/echo</sourceUrl>
<destinationUrl>file :///home/globus/my_echo</destinationUrl>
</transfer>
…
<transfer>
<sourceUrl>file  :///home/globus/stpout</sourceUrl>
<destinationUrl>gsiftp ://poste2.an.lri : 2811/home/globus/stdout
–ftp</destinationUrl>
</transfer>
…
```

Pour soumettre notre job, il est nécessaire d'utiliser l'option '-S' pour signaler la présence d'un transfert.

```
[globus@poste4 ]$ globusrun-ws –submit –S -f test-Ftp.xml
Delegating user credentials…Done.
Submitting job…Done.
…
```

Le fichier ' sdout ' -ftp contient :

```
[globus@poste3 ]$ cat stdout-Ftp
Hello World!
```

2.2.2 Soumission d'un job Multiple

Il s'agit de l'exécution de plusieurs jobs en même temps sur un nœud. dans cet exemple nous avons deux job : '/bin/date/' et '/bin/echo ' .

```
[globus@poste4 ]$ cat test_Multi.xml
```

```
[globus@poste4 ]$ globusrun-ws –submit -f test_Multi.xml
Delegating user credentials…Done.
Submitting job…Done.
…
Destroying job…Done.
Cleaning up any delegated credentials…Done.
```

Les fichiers résultats sont les suivants :

```
[globus@poste4 ]$ cat stdout.p1
Hello
[globus@poste4 ]$ cat stdout.p2
Hello World!
```

2.3 Test de soumission de jobs d'exécutable distants sur un nœud distant

2.3.1 1er cas

L'exécutable et le fichier 'stdout ' sont dans le nœud distant, dans notre cas c'est 'poste4.an.lri' mais le lancement de l'exécution se fait à partir du 'poste1.an.lri'.

```
[globus@poste1 ]$ cat job.xml
```

Puis le lancement de job avec :

```
[globus@poste1 ]$ Globusrun-ws -submit -F
https ://192.168.61.4 :8443/wsrf/services/
ManagedJobFactoryService -f  job.xml
```

2.3.2 2ème cas

L'exécutable est dans le 'poste3.an.lri', le lancement de job se fait à partir du 'poste1.an.lri' bien que l'exécution est lancée sur le nœud 'poste4.an.lri'. Les fichiers 'stdout' et 'stdin' résident dans le nœud 'poste4.an.lri'.

```
[globus@poste1 ]$ cat jobsiftp.xml
…
<argument>executable poste3 :gsiftp</argument>
<argument>remote hoste poste4</argument>
<argument>stdin,stdout,stderr local</argument>
…
<transfer>
<sourceUrl>gsiftp ://poste3.an.lri/bin/echo</sourceUrl>
<destinationUrl>file :///home/globus/my_output</destinationUrl>
</transfer>
…
```

Le lancement du job se fait par l'option '-S'pour signaler la présence d'un transfert dans le job.

```
[globus@poste1 ]$ globusrun-ws -submit -S -F
```

Le résultat est:

```
[globus@poste4 ]S cat gsiout
executable poste3 :gsiftp
remote hoste  poste4
Stdin,stdout,stderr local
```

2.3.3 3ème cas

Maintenant le fichier résultat sera transféré par le service 'gsiftp' vers une autre machine.

```
[globus@poste1 ]$ cat test2.xml
```

Lancement du job:

```
[globus@poste1 ]$ globusrun -ws -submit -S -F
```

Le résultat sera le fichier :

```
[globus@poste5 ]$ cat stdout
Le nombre de caractère dans le fichier est : 13
```

2.3.4 4ème cas

Dans ce cas notre exécutable est un programme C qui prend en argument un fichier texte et calcule le nombre de caractères contenus dans ce fichier. L'exécutable ainsi que le fichier argument seront transférés du 'poste3.an.lri ' au 'poste4.an.lri' avec le service 'gsiftp'.

```
[globus@poste1 ]$ vi test3.xml
[globus@poste1 ]$ cat test3.xml
```

Lancement du job:

```
[globus@poste1 ]$ globusrun -ws -submit –S -F
https ://192.168.61.4 :8443/wsrf/services/ManagedJobFactoryService -f
test3.xml
Delegating user credentials…Done.
Submitting job…Done.
…
Cleaning up any delegated credentials…Done.
```

2.3.5 5ème cas

Il s'agit d'un job Multiple sur des machines différentes. L'une sur 'poste4.an.lri' et l'autre sur 'poste5.an.lri'.

```
[globus@poste1 ]$ cat test4.xml
…
<wsa :Address>https ://poste4.an.lri
:8443/wsrf/services/ManagedJobFactoryService</wsa:Address>
…
<wsa :Address>https ://poste5.an.lri :
8443/wsrf/services/ManagedJobFactoryService</wsa:Address>
…
```

Lors du lancement, l'option '-j' est importante parce qu'il s'agit d'un job Multiple.

```
[globus@poste1 ]S  globusrun -ws – submit -j -F
Https ://192.168.61..4.: _8443/wsrf/services/
ManagedJobFactoryService –f test4.xml
Delegating user credentials…Done.
Submitting job…Done.
…
Cleaning up any delegated credentials…Done.
```

Les fichiers résultats sont :

```
[globus@poste4 ]$ cat sdout.p1
[globus@poste5 ]$ cat sdout.p2
 Hello World !
```

2.3.6 6ème cas

Les jobs s'exécutent sur 'poste3.an.lri' et sur 'poste5.an.lri' puis nous récupérons le résultat sur 'poste4.an.lri'.

```
[globus@poste1 ]$ cat test5.xml
…
<wsa :Address>
https ://poste4.an.lri :8443/wsrf/services/ManagedJobFactoryService
</wsa :Address>
…
<wsa :Address>https ://poste3.an.lri
:8443/wsrf/services/ManagedJobFactoryService</wsa :Address>
…
<wsa :Address>https ://poste5.an.lri
:8443/wsrf/services/ManagedJobFactoryService</wsa :Address>
…
```

Lancement du job :

```
[globus@poste1 ]$ globusrun-ws -submit -J -F
https ://192.168.61.4 :8443/wsrf/services/ManagedJobFactoryService –f
test5.xml
```

Le résultat est :

```
[globus@poste4 ]$ cat sout.p1
[globus@poste4 ]$ cat sout.p2
Hello World!
```

2.3.7 7ème cas

Dans ce test le fichier résultat de l'exécution du premier job sur 'poste3.an.lri' est l'argument du deuxième job qui s'exécutera sur 'poste5.an.lri':

```
[globus@poste1 ]$ cat test7.xml
```

Dans ce cas on utilise les options '-S' et '-J' car il s'agit en même temps d'un multi-job et d'un transfert.

```
[globus@poste1 ]$ globusrun –ws -submit –J –S -F
https ://192.168.61.4 :8443/wsrf/services/ManagedJobFactoryService –f
test7.xml
```

Le fichier sout 1.p1 est dans le poste3.an.lri :

```
[globus@poste1 ]$ globus-url-copy gsiftp :
//poste3.an.lri/home/globus/sout1.p1
file :///home/globus/sout1.p1
[globus@poste1 ]$ cat sout1.p1
le nb de ligne est :13
le nb de ligne est :13
le nb de ligne est :13
le nb de ligne est :13
le nb de ligne est :13
le nb de ligne est :13
```

Le fichier stout 2.p2 est dans le 'poste5.an.lri'

```
[globus@poste1 ]$ globus-url-copy gsiftp :
//poste5.an.lri/home/globus/sout74.p2
file :///home/globus/sout2.p2
[globus@poste1 ]$ cat sout2.p2
Le nb de ligne est :138
```

3. Etude de cas (DEPLOIEMENT D'UN SERVICE DE GRILLE : DATAMINING SERVICE)

3.1 Introduction

Le scénario de notre expérimentation est décrit comme suit :

Un Service Messagerie Electronique est utilisé pour que les participants se mettent d'accord sur un rendez-vous. Puis à chaque participant est attribué un passeport électronique qui correspond à un service d'authentification reposé sur la notion de proxy d'authentification de l'infrastructure GSI et qui consiste à définir les accès aux sessions, ainsi que les services disponibles après identification. Chaque membre de la communauté virtuelle peut accéder à la session collaborative et y participer grâce à un terminal léger connecté directement via Internet au serveur Web. Lors du déploiement du service Grid, les participants utilisent collectivement différents services fournis par Sakai (cf. figure 4.5) qui sont le complément parfait d'Agora (cf. figure 4.6): Service de la Vidéoconférence (se réunir et communiquer en même temps), Service tableau (collaborer, joindre des photos…), Service de Movicasting (diffusion des films), Service de la messagerie instantanée (un simple outil chat), Service de bureau partagé (transmettre le bureau, aucune installation de logiciel, une manière très simple pour l'édition collaborative) et enfin le Service d'enregistrement des réunion).

Figure 4.5. Les outils de Sakai

Notre architecture (cf. Figure 15) montre les différents composants du système :

Nous allons mettre en place un réseau LAN constitué de quatre machines, sur chacune est installé le système Linux Fedora core 4, dont le but de construire notre grille de calcul. Linux est le système d'exploitation le plus approprié pour une installation complète de Globus. Ainsi, il offre un environnement très fiable et sécurisé. Les machines utilisées comportent les caractéristiques suivantes :

- Fabriquant : Intel.
- Processeur: Intel(R) Pentium(R) 4 CPU 3.00GHZ (2CPUs).
- Mémoire: 1 GB RAM.

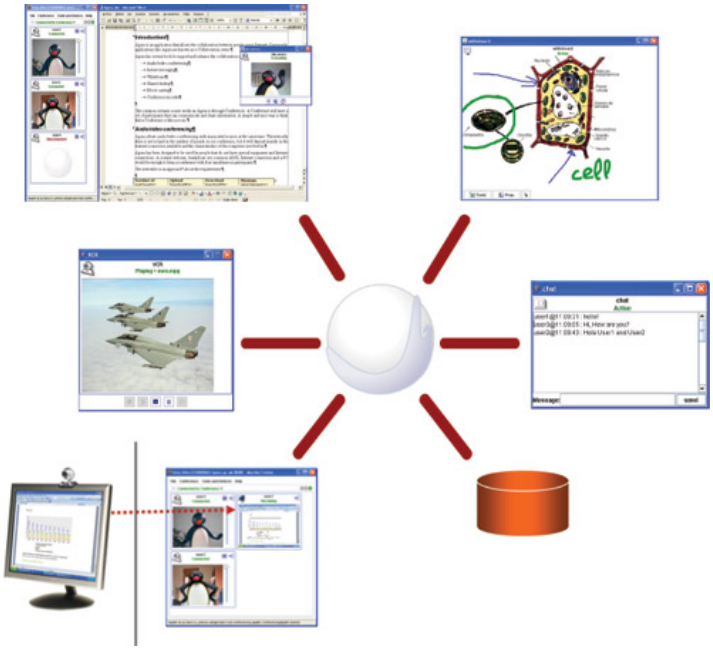


Figure 4.6. Les caractéristiques d'Agora

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

- type d'installation poste de travail (WorkStation).
- désactivation de pare-feu pour ne pas gêner le fonctionnement des services de l'intergitiel.
- date et heure doivent être réglées pour chaque machine, cette condition est indispensable dans la phase de signature de certificat pour les autres machines afin de vérifier la validité du proxy.

Lors de l'installation de linux, nous avons attribué un nom et une adresse IP à chaque noeud de la grille qui vont servir par la suite à la configuration de l'intergiciel Globus. Chaque nom du hôte doit avoir la forme suivante:'nom_machine.nom_domaine' (une exigence de l'intergiciel Globus toolkit).

Dans cette illustration la grille est configurée de la manière suivante:

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

On a quelques outils nécessaires à installer qui sont :

1. **Apache Ant :** c'est un exécuteur de taches il permet le déploiement des programmes (déploiement de services)

2. **JDK :** nécessaire pour la compilation du code de Globus car la plus grande partie est écrite en java.

3. **PostgresSQL :** système de gestion de base de données relationnelle fonctionnant sur des systèmes UNIX. Il est composé de deux parties :

    - **Partie serveur :** c'est la partie fonctionnant sur la machine hébergeant la base de données capable de traiter les requêtes des clients

    - **Partie client :** cette partie est installée sur les postes client. Les clients interrogent le serveur de base de données par des requêtes SQL.

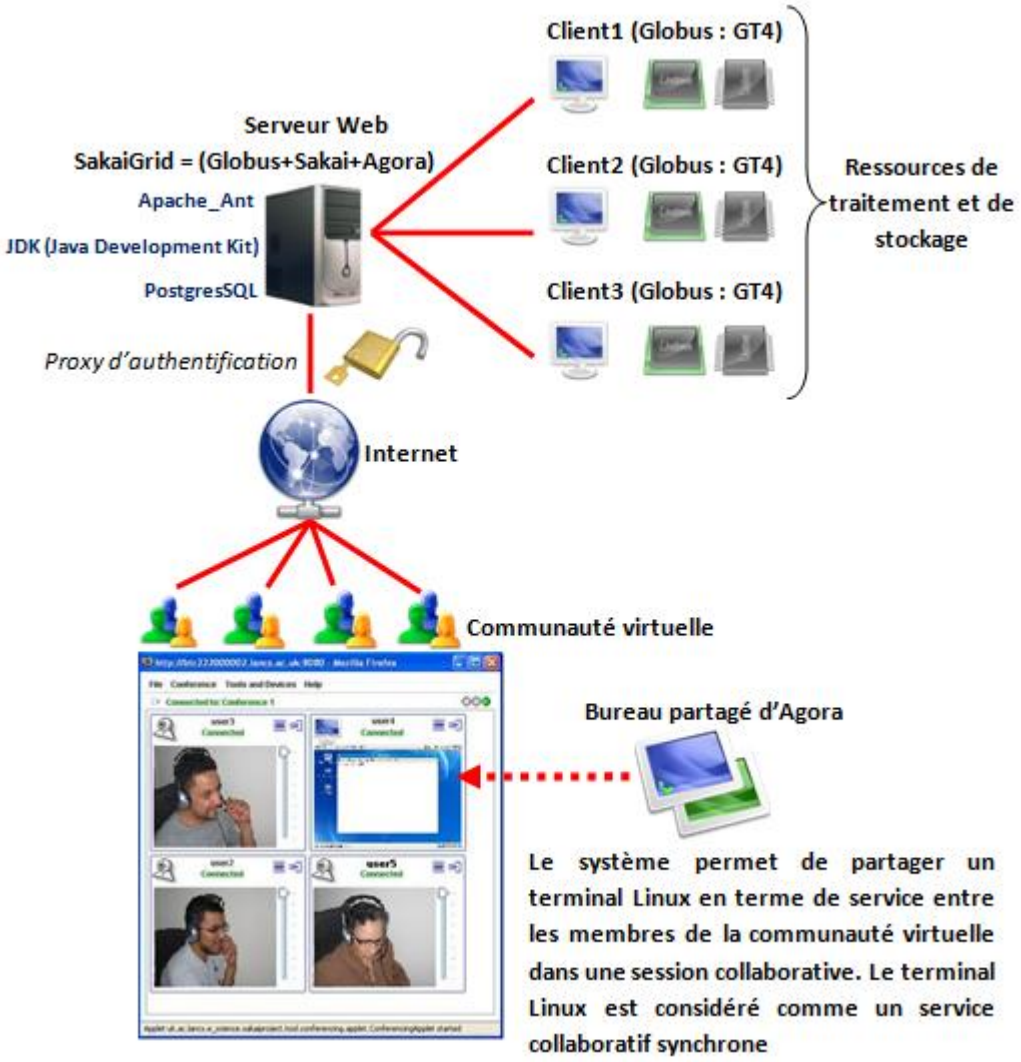


Figure 4.7: déploiement d'un service Grid (Datamining Service) partagé

Dans cette section, nous présentons un exemple de service de grille qui permet d'exécuter les deux algorithmes de Datamining "apriori" et "sampling". Nous présentons les codes nécessaires pour chaque étape du déploiement du service " Datamining service".

Ce diagramme montre la structure des dossiers nécessaires pour l'implantation et le déploiement de notre service. On doit donc commencer par les créer.

```
[globus@poste4 ]$ mkdir Datamining Datamining/schema
Datamining/Schema/examples
[globus@poste4 ]$ mkdir org org/globus org/globus/examples
org/globus/rxamples/services org/globus/examples/clients
```

Tous les dossiers requis pour établir ce service sont inclus dans le dossier "Datamining" qui sont :

- Fichier ant build (globus-build -service.sh)
- Fichier de construction (Build script)
- Fichier namespace mappings (build.mappings)

3.2 Étape de création d'un service de grille

La première étape consiste à définir l'interface du service. Pour cela nous avons besoin de spécifier ce que notre service va fournir au monde externe. A ce point nous ne somme pas intéressés aux mécanismes intérieurs de ce service (quels algorithmes il utilise ; a quelles bases de Données il accède ….). Nous avons juste besoin de savoir quelles opérations sont disponibles à nos utilisateurs. Dans le langage de Service Web, l'interface de service est appelée " portType" .

Pour notre service, nous avons définit une simple interface avec les méthodes startDatamining, apriori Datamining, samplingDatamining et stopDatamining.

```
public interface Datamining {
public void startDatamining();
public void aprioriDatamining(String S);
public void samplingDatamining(String S);
public void stopDatamining();
}
```

Le fichier suivant Datamining.wsdl écrit en langage WSDL et la description équivalente de l'interface ci-dessus, nous définissons ainsi après le nom de service, les types des opérations avec un schéma XML, les messages d'entré et de sortie pour chaque opération et enfin les PortTypes. Ce fichier est sous :

```
/home/globus/Datamining/schema/examples/DataminingService_instance/Dataming
.wsdl
```

La deuxième étape consiste à implémenter cette interface avec une classe java "DataminingService.java". Ce fichier est sous:

```
/home/globus/Datamining/org/globus/examples/Services/core/first/impl/Datami
ning.java
```

Voici son contenu:

```
package   org.globus.examples.services.core.first.impl;
Import java.rmi.RemoteException;
…
public class DataminingService implements Resource, ResourceProperties {
/*Resource Property set */
private ResourcePropertySet propSet;
/* Constructor.Initializes RPs */
public DataminingService () throws RemoteException {
/* Create RP set*/
this.propSet = new SimpleResourcePropertySet
(DataminingQNames.RESOURCE_PROPERTIES);
}
public StartDataminingResponse startDatamining (StartDatamining params)
throws
RemoteException {
StopWatche sw = new StopWatch();
Sw.start();
return new StartDataminingResponse();
}
public AprioriDataminingResponse aprioriDatamining(String S) throws
RemoteException {
try {
StringTokenizer st = new StringTokenizer(S," ");
String testfile = new String (st.nextToken());
int support = Integer.parseInt (st.nextToken());
String outfile = new String (st.nextToken());
Apriori_apriori = new Apriori (testfile,support,outfile);
_apriori.findFrequentSets();
_apriori.printFrequentSets();
}
catch (Exception e) {
e.printStackTrace();
}
return new AprioriDataminingResponse();
}
public SamplingDataminingResponse samplingDatamining(String S)
throws RemoteException {
try {
StringTokenizer st = new StringTokenizer (S," ");
```

```
String testfile = new String (st.nextToken());
int support = Integer.parseInt (st.nextToken());
int samplesize = Integer.parseInt (st.nextToken());
Sampling_sampling = new Sampling (testfile,support,samplesize);
_sampling.findFrequentSets ();
_sampling.printFrequentSets ();
}
catch (Exception e) {
e.printStackTrace();
}
return new SamplingDataminingResponse();
}
public StopDataminingResponse stopDatamining (StopDatamining params)
throws
RemoteException {
StopWatch sw = new StopWatch();
sw.stop();
sw.print();
return new StopDataminingResponse();
}
/* Required by interface ResourceProperties */
public ResourcePropertySet getResourcePropertySet() {
return this.propSet;
}
}
```

Dans le constructeur de notre classe nous avons fait une simple initialisation des propriétés des ressources (ResourceProperties) en faisant appelle à la classe DataminingQNames.java dont le contenu est le suivant:

```
package  org.globus.examples.services.core.first.impl;
import javax.xml.namespace.QName;
public interface DataminingQNames {
public static final String NS =
"http://www.globus.org/namespaces/examples/core/DataminingService_instance"
;
public static final QName RESOURCE_PROPERTIES = new QName(NS,
"DataminingResourceProperties");
}
```

Ce fichier est sous:

```
/home/globus/Datamining/org/globus/examples/services/core/first/impl/Datami
ningQNames.java
```

Pour les méthodes de la classe DataminingService nous avons fait pratiquement l'instanciation des trois classes de Datamining (StopWatch.java, Apriori.java, Sampling.java).

Pour cela nous avons les importé tous dans notre paquetage:

```
package  org.globus.examples.services.core.first.impl
```

Maintenant, nous devons rassembler tous ces morceaux et les rendre disponibles à travers le container de notre grille. Cette phase est appelée déploiement du service de grille. Un des composants clés de la phase du déploiement est un fichier appelé: Descripteur du déploiement, écrit en langage WSDD (Web Service Deployment Descriptor).

Le descripteur du déploiement de notre service deploy_server.wsdd est sous:

```
/home/globus/Datamining/org/globus/examples/services/core/first/deploy-
server.wsdd
```

Un autre fichier deploy-jndi-config.xml est inclus pour un déploiement plus simple avec le service factory. Ainsi, on aura le fichier du déploiement JNDI. Son chemin est :

```
/home/globus/Datamining/org/globus/examples/services/core/first/deploy-
jndi-config.xml
```

A partir des fichiers créés, nous constituons un paquetage de type GAR via l'outil Ant.

```
./globus-build-service.sh
-d org/globus/examples/services/core/first/
-s schema/examples/DataminingService_instance/Datamining.wsdl
```

Après déploie de service dans le container des autres services grille.

```
globus-deploy–gar Datamining/
Org–globus–examples-services–core-first.gar
```

La dernière étape consiste à écrire le code client afin d'invoquer le service DataminingService déjà déployé. Le code présenté ci-dessous est celui du fichier client.java :

```
package  org.globus.examples.clients.DataminingService_instance;
import org.apache.axis.message.addressing.Address;
…
import java.lang.String;
public class Client {
public static void main (String[] args) {
System.out.println(System.CurrentTimeMillis());
Process p;
String testfile = "test.dat";
String outfile = "";
String support = "";
int samplesize = 10;
try {
```

```
support = args[3];
catch (Exception  e) {
System.out.println("Didn't get number of iteration");
}
String S;
try {
testfile = args[2];}
catch (Exception  e) {
System.out.println("Didn't get filename. Using'" + testfile + " '.");
}
DataminingServiceAddressingLocator locator = new
DataminingServiceAddressingLocator();
try {
String serviceURI = args[0];
EndpointReferenceType endpoint = new EndpointReferenceType();
Endpoint.setAddress(new Address(serviceURI));
DataminingPortType datamining = locator.getDataminingPortType(endpoint);
System.out.println("Hello");
if (args[1].equals("Apriori"))
{
try {
outfile = args[4];
}
catch(Exception  e) {
System.out.println(“Didn't get output file name. Not printing");
}
S = testfile+" "+support+" "+outfile;
Sytem.out.println(S);
datamining.startDatamining(new StartDatamining());
datamining.aprioriDatamining(S);
datamining.stopDatamining(new StopDatamining());
}
else if(args[1].equals("Sampling"))
{
Try {
sampleize = Integer.parseInt(args[4]);
}
catch (Exception e) {
System.out.println(“Didn't get a sampleize param : it must be integer");
}
```

```
outfile = "outSampling";
S = testfile+" "+support+" "+sampleize;
System.out.println(S);
datamining.startDatamining(new StartDatamining());
datamining.samplingDatamining(S);
datamining.stopDatamining(new StopDatamining());
}
String adrIP = " ";
try {
StringTokenizer stURI = new StringTokenizer(serviceURI,"//");
Stirng s1 = new String (stURI.nextToken());
String s 2 = new String ( stURI.nextToken());
StringTokenizer stIP = new StringTokenizer(s2,":");
adrIP = new String(stIP. nextToken());
}
catch (Exception  e) {
e.printStackTrace();
}
System.out.println(adrIP);
p = Runtime.getRuntime().exec("globus–url-copy gsiftp :
//"+adreIP+"/home/globus/"+outfile+" "+"file :///tmp/"+outfile);
System.out.println(System.currentTimeMillis());
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

La méthode curentTimeMillis () est utilisée pour récupérer le temps de réponse de notre service suite à une invocation par le client.

La commande globus–url–copy est implémentée dans le code client pour lui permettre de récupérer ses fichiers de sortie. après la compilation du code du client avec la commande javac :

```
javac
-classpath ./build/stubs/classes/ :$CLASSPATH
org/globus/examples/clients/DataminingService_instance/
Client.java
```

L'invocation du service DataminingService par le client désirant exécuter l'algorithme apriori se fait avec la commande :

```
java
-classpath ./build/stubs/classes/ :$CLASSPATH
```

```
org.globus.examples.clients.DataminingService_instance.Client/
http://192.168.61.4:8080/wsfr/services/examples/core/
first/DtaminingService Apriori test.dat 2fileout
```

Pour lancer l'algorithme sampling avec le service Datamining service :

```
java
classpath ./build/stubs/classes/:$CLASPATH
org.globus.examples.clients.DataminingService–instance.client
http ://192.168.61.4:8080:wsrf/services/examples/core/
first/DataminingService Sampling test.dat 5 10
```

Le résultat est le suivant :

- **Coté client :**

```
test.dat 1 output
Pleae find the output file in :/tmp/output
37s 164ms
```

- **Cote serveur**

```
pass : 1, total : 5, candidates : 6, pruned : 1
pass : 2, total : 12, candidates : 10, pruned : 3
pass : 3, total : 14, candidates : 3, pruned : 1
number of frequent itemsets found : 14
0.0 seconds
```

3.3 Le service Factory DataminingFactoryService:

Afin de déployer notre service DataminingService sur toute la grille et de le rendre accessible a partir des nœuds distants (un client distant peut invoquer une instance de ce service en passant par le FactoryService pour lui créer une nouvelle référence), nous avons développé et implémenté le service DataminingFactoryService qui répond à notre besoin .

Ainsi la première étape consiste à écrire le fichier Factory.wsdl sous le chemin suivant :

```
/home/globus/Datamining/schema/examples/FactoryService/Factory.wsdl
```

Maintenant, nous devons écrire l'implémentation java du service Factory "DataminingFactoryService.java ". C'est une simple classe avec une simple méthode createResource(). Ce script existe sous le chemin suivant :

```
/home/globus/Datamining/org/globus/examples/services/core/
Factory/impl/DataminingFactoryService.java
```

```
package org.globus.examples.services.core.factory.impl;
import java.rmi.RemoteException;
```

```
…
Import org.globus.www.namespaces.examples.core.FactoryService
.CreateResourceResponse;
public class DatamingFactoryService {
/*Implementation of createResource Operation*/
public CreateResourceResponse createResource(CreateResource request)
throws RemoteException {
ResourceContext ctx = null;
DataminingResourceHome home = null;
ResourceKey key = null;
try {
ctx = ResourceContext.getResourceContext();
home = (DataminingResourceHome) ctx.getResourceHome();
key = home.create();
}catch (Exception  e) {
throw new RemoteException("",e);
}
EndpointReferenceType epr = null;
try {
URL baseURL = ServiceHost.getBaseURL();
String instanceService = (String) MessageContext
.getCurrentContext().getService().getOption("instance");
String instanceURI = baseURL.toString()+instanceService;
//the endpoint Reference includes the instance's URI and the resource key
epr = AddressingUtils.createEndpointReference(instanceURI,key);
} catch (Exception e) {
throw new RemoteException("",e);
}
CreateResourceResponse response = new CreateResourceResponse();
response.setEndpointReference(epr);
return response ;
}
}
```

Dans ce qui suit nous détaillons les scripts nécessaires pour l'implémentation du service DataminingService. Un premier script pour les ressources, notant que chaque ressource aura une clé unique pour l'identifier, nous avons recours à implémenter l'interface ResourceIdentifier pour que nous puissions utiliser la méthode getID ( ). Le script DataminingResourrce.java est le suivant :

```
package org.globus.examples.services.core.factory.impl;
```

```
import org.globus.wsrf.Resource ;
…
import org.globus.wsrf.impl.ReflectionResourceProperty ;
public class DataminingResource implements Resource, ResourceIdentifier,
ResourceProperties {
/* Resource key. This uniquely identifies this resource.*/
private Object key;
/*Resource Property Set*/
private ResourcePropertySet propSet;
/* Required by interface ResourceIdentifier*/
public Object getID(){
return this.key;
}
/*Initializes RPs and returns a unique identifier for this resource*/
public Object initialize() throws Exception {
this.key = new Integer(hashCode());
this.propSet = new SimpleResourcePropertySet
(DataminingQNames.RESOURCE_PROPERTIES);
//Initialize the resource properties
return key;
}
public ResourcePropertySet getResourcePropertySet() {
return this.propSet;
}
}
```

Le chemin de ce script est le suivant:

/home/globus/Datamining/org/globus/examples/services/core/Factory/impl/DataminingResource.java

Un deuxième script DataminingRessourceHome.java est ajouté sous le chemin suivant:

/home/globus/Datamining/org/globus:examples/services/core/Factory/impl/DataminingResourceHome.java

```
package org.globus.examples.services.core.factory.impl;
import org.globus.wsrf.ResourceKey;
import org.globus.wsrf.impl.ResourceHomeImpl;
import org.globus.wsrf.SimpleResourceKey;
Public class DataminingResourceHome extends ResourceHomeImpl {
public ResourceKey create() throws Exception {
```

```
//Create a resource and initialize it
DataminingResource dataminingResource =
(DataminingResource) createNewInstance(); (1)
dataminingResource.initialize(); (2)
//Get key
ResourceKey key = new SimpleResourceKey (keyTypeName,
dataminingResource.getID()); (3)
//Add the resource to the list of resources in this home
add(key,dataminingResource); (4)
return key; (5)
}
}
```

(1) : Création d'une nouvelle instance de ressource. Notons que ceci ne nécessite pas une nouvelle classe mais tout simplement l'utilisation de la méthode protégée createNewInstance. En outre, puisque createNewInstance renvoie un objet de type ressource, nous devons faire le transtypage de notre type spécifique : DataminingResourceType.

(2) : Initialisation de la ressource.

(3) : Récupération de l'identifiant de ressource en utilisant la méthode getID, et l'employer pour créer un objet SimpleRessourceKey.

(4) : Ajout de la ressource récemment créée et sa clé à la liste interne des ressources.

(5) : Enfin, nous renvoyons la clé de la ressource.

L'implémentation de l'instance du service DataminingService est sous :

/home/globus/Datamining/org/globus/examples/services/core/Fectory/impl/DataminingService.java

```
package org.globus.examples.services.core.factory.impl;
import java.rmi.RemoteException;
…
import java.io.*;
public class DataminingService {
/*Private method that gets a reference to the resource specified in the
endpoint reference.
*/
private DataminingResource getResource() throws RemoteException {
Object resource = null;
public class DataminingService {
/*Private method that gets a reference to the resource specified in the
```

```
endpoint reference.
*/
private DataminingResource getResource() throws RemoteException {
Object resource = null;
try {
resource = ResourceContext.getResourceContext().getResource();
} catch (Exception e) {
throw new RemoteException("Unable to access resource.",e);
}
DataminingResource dataminingResource = (DataminingResource) resource;
return dataminingResource;
}
public StartDataminingResponse startDatamining
(StartDatamining params) throws RemoteException {
StopWatch sw = new StopWatch();
sw.start();
return new StartDataminingResponse();
}
public AprioriDataminingResponse aprioriDatamining(String S)
throws RemoteException {
try {
StringTokenizer st = new StringTokenizer(S," ");
String testfile = new String(st.nextToken());
int support = Integer.parseInt(st.nextToken());
String outfile = new String(st.nextToken());
Apriori_apriori = new Apriori(testfile,support,outfile);
_apriori.findFrequentSets();
_apriori.printFrequentSets();
}
catch ( Exception e ) {
e.printStackTrace();
}
return new AprioriDataminingResponse();
}
}
public SamplingDataminingResponse samplingDatamining(String S)
throws RemoteException {
try {
StringTokenizer st = new StringTokenizer(S," ");
String testfile = new String(st.nextToken());
```

```
int support = Integer.parseInt(st.nextToken());
int sampleize = Integer.parseInt(st.nextToken());
Smpling_sampling = new Sampling(testfile,support,samplesize);
_sampling.findFrequentSets();
_sampling.printFrequentSets();
}
catch (Exception e) {
e.printStackTrace();
}
return new SamplingDataminingResponse();
}
public StopDataminingResponse  stopDatamining (StopDatamining params)
throws RemoteException {
StopWatch sw = new StopWatch();
sw.stop();
sw.print();
return new StopDataminingResponse();
}
}
```

Notons que nous avons importé toutes les classes nécessaires pour les deux algorithmes Apriori et Sampling comme Apriori.java, StopWatch.java.

Après avoir écrire les deux parties importantes de l'implémentation, il nous reste que le déploiement de notre nouveau service.

Le premier fichier responsable du déploiement de FactoryService et de l'instance DataminingFactoryService est le deploy-server.wsdd qui est sous le chemin suivant:

```
/home/globus/Datamining/org/globus/examples/services/core/
Factory/deploy-server.wsdd
```

En plus le fichier deploy-jndi-config.xml doit inclure les deux services (Factory Service et Instance Service) et il est au même niveau que le fichier WSDD.

Nous déployions le nouveau service avec la commande suivante (elle est responsable de générer tous les stubs nécessaires et de compiler tous les scripts) :

```
./globus-build-service.sh
-d org/globus/examples/services/core/factory/
-s schema/examples/DataminingFactoryService/Factory.wsdl
```

Maintenant on doit rassembler tous ces scripts pour générer le paquetage GAR, celui-ci est fait avec Ant en lançant la commande.

```
globus-deploy-gar Datamining/org_globus_examples_services_core_factory.gar
```

Une fois que le service DataminingFactoryService est bien configuré, on vérifie s'il appartient au container des autres services de grille avec la commande :

```
Globus-start-container-nosec
```

Après avoir déployé le service DataminigFactoryService et vérifié qu'il appartient à la liste des autres services, il nous reste d'écrire le code client.

Avec se nouveau service le client doit créer une instance du localisateur (instanceLocator) du service DataminigFactoryService et une instance factory Locator du service FactoryService. Ce dernier lui permet de créer une ressource avec une clé unique qui lui permet de s'identifier afin d'appeler les opérations du service comme si localement. Le code client est le suivant : (ce qui est en gras représente la différence avec le client précédent du service DataminingService).

```
package org.globus.examples.clients.FactoryService_Datamining;
import org.apache.axis.message.addressing.Address;
…
import java.lang.String;
public class Client {
public static void main (String[]args) {
lang l1 = System.currentTimeMillis());
Process p;
String testfile = "test.dat"
String outfile = " ";
String support = " ";
int sampleize = 10;
try {
support = args[3];
}
catch (Exception e){
System.out.println("Didn't get number of iteration");
}
String S;
try {
testfile = args[2]; }
catch (Exception e){
System.out.println("Didn't get filename. Using '"+testfile+"'.");
}
FactoryServicesAddressingLocator factoryLocator = new
FactoryServicesAddressingLocator();
DataminingServicesAddressingLocator instanceLocator = new
DataminingServicesAddressingLocator();
try {
```

```
String factoryURI = args[0];
EndpointReferenceType factoryEPR, instanceEPR;
FactoryPortType dataminingFactory;
DataminingPortType datamining;
factoryEPR = new EndpointReferenceType();
factoryEPR.setAddress(new Address(factoryURI));
dataminingFactory = factoryLocator.getFactoryPortTypePort(factoryEPR);
CreateResourceResponse createResponse = dataminingFactory
.createResource (new CreateResource());
instanceEPR = createResponse.getEndpointReference());
datamining = instanceLocator.getDamatiningPortTypePort(instanceEPR);
System.out.println("Created instance.");
if (args[1].equals("Apriori"))
{
try {
outfile = args[4];
}
catch (Exception e) {
System.out.println("Didn't get output filename. Not printing");
}
S = testfile+" "+support+" "+outfile;
System.out.println(S);
datamining.startDatamining(new StartDatamining());
datamining.aprioriDatamining(S);
datamining.stopDatamining (new StopDatamining());
}
else if(args[1].equals("sampling"))
}
{
try {
Samplesize = Integer.parseInt(args[4]);
}
catch (Exception e) {
System.out.println("Didn't get a samplzise param : it must be integer");}
outfile = "outSamlping";
S = testfile+" "+support+" "+sampleseize;
System.out.println(S);
Datamining.startDatamining(new StartDatamining());
Datamining.samplingDatamining(S);
datamining.stopDatamining(new StopDatamining());
```

```
String adrIP = "";
try{
StringTokrnizer stURI = new StringTokenizer(factoryURI, "//");
String s1 = new String(stURI.nextToken());
Sring s2 = new String(stURI.nextToken());
StringTokenizer stIP = new StringTokenizer(s2, ":");
adrIP = new String(stIP.nextToken());
}
catch (Exception  e) {
e.printStackTrace();
}
System.out.println(adrIP);
}
p = Runtime.getRuntime().exec("globus–url–copy gsiftp
://"+adrIP+"/home/globus/"+outfile+" "+file :///tmp/"+outfile);
long l2 = System.currentTimeMillis());
long l = (l1-l2)/1000;
long l3 = ( l1-l2) mode 1000;
System.out.println(l+"s"+" "+l3+"ms");
} catch (Exception e) {
}
}
}
```

La procédure de compilation et d'exécution est la même que précédemment sauf que maintenant un client distant peut invoquer notre service.

Conclusion

Dans ce chapitre nous avons présenté une proposition d'architecture d'une plateforme collaborative flexible exprimée en terme de couches basée sur les concepts clés d'OGSA ainsi que ses processus de partage efficace des ressources. Les participants de la communauté virtuelle peuvent communiquer en deux mode d'interaction (synchrone/asynchrone) en utilisant ainsi le bureau partagé d'Agora qui est un outil très simple à utiliser et très puissant. Cet outil s'adapte bien avec la classe virtuelle et les contextes d'apprentissage à distance. Ainsi il offre des caractéristiques intégrées telles que la vidéoconférence, le bureau partagé, le tableau blanc et la messagerie instantanée. Agora n'a pas besoin d'un matériel spécifique coûteux pour s'exécuter, il a besoin seulement d'une webcam de 40$ et d'un casque. La nouvelle plateforme Sakai, est devenue très riche en terme de services (services Web/Grid). Avec la présence de la gestion de l'état dans les services, les services Grid sont adaptés aux services collaboratifs synchrones et asynchrones. Une démonstration pour le co-déploiement d'un Service Grid partagé a montée la flexibilité et l'efficacité de notre approche en palliant aux problèmes souvent rencontrés pour une telle collaboration: problème de la distance, ressources hétérogènes, nombre important de participants, coût élevé de la collaboration, absence d'historique, collaboration limitée dans le temps, rentabilité et productivité critiquables.

Conclusion générale

Après avoir décrit un état de l'art pour caractériser le concept espaces collaboratifs ainsi que le passage des services Web aux services Grid, on a adopté dans la partie contribution deux plateformes : la plateforme collaborative Sakai et L'intergiciel Globus Toolkit. La première est choisie parmi quatre plateformes collaboratives très célèbres au monde selon leurs fonctionnalités, leurs technologies en matière d'applications d'entreprise ainsi que leurs évolutions à venir. La deuxième consiste à étudier son architecture en faisant apparaitre celle qui constitue la base de nombreux travaux. Ainsi on a terminé cette partie par une expérimentation. On a proposé notre architecture de la nouvelle plateforme collaborative exprimée en termes de couches. Ainsi, des tests sur notre grille de calcul « Globus » sont fait afin de montrer le bon fonctionnement de celle-ci avec un déploiement d'un service Grid, en utilisant un outil de collaboration très simple et très fiable en même temps qui est Agora. Ces résultats sont essentiellement qualitatifs afin de comprendre le point de vue utilisateur.

La grille apporte un nouveau paradigme d'accessibilité aux ressources qui pallient aux limites de distribution web basée sur le modèle client/serveur. En effet, à la différence du web actuel, dans un environnement de Grid il est possible à chaque entité participante d'être à la fois client et serveur, ce qui constitue un atout majeur dans un environnement de formation où les participants doivent fortement collaborer. Les approches de distribution basées sur les services web sont limitées quant à la disponibilité, l'extensibilité et la distribution des ressources. L'objectif ultime d'une grille est de transformer le réseau mondial des ordinateurs en ressources immenses et unifiées de capacité de traitement. Selon Rob Bjornson et Andrew Sherman**,** une grille permet aux utilisateurs de collecter et d'organiser les ressources disparates en une entité visuelle plus uniforme et plus maniable, et de rendre ces ressources virtuelles simultanément accessibles à plusieurs utilisateurs. Cette définition de la grille concorde parfaitement avec les exigences d'un système d'elearning, qui sont la collaboration et la coopération entre tous les participants. Autrement dit, au cours d'un apprentissage en ligne, il doit exister des moyens qui rendent possibles les interactions entre les entités qui participent à la formation.

Ainsi, Là où les services Web sont sans état et persistants, les instances de service Grid peuvent être soit avec ou sans état et éphémères ou persistants. Les services sans état sont synchrones (i.e., les messages ne peuvent être "bufférisés"), point-à-points (i.e., utilisables par un seul utilisateur), et interagissent par de simple requête/réponse. Ils retournent simplement une réponse à une

invocation précise. Les services à état peuvent être asynchrones, multi-points et interagissent par des conversations. Ils permettent plus d'adaptation dans le service fourni.

Enfin, une autre raison et pas des moindre qui motive l'usage de Grid est la sécurité. Pour qu'un espace de collaboration trouve pleinement de son potentiel d'utilisation la sécurité joue un rôle fondamental. La notion de sécurité s'étend au delà de l'usage que l'on en fait habituellement. Pris au sens large, la sécurité telle que nous l'entendons ici vise à donner aux utilisateurs une confiance suffisante pour s'approprier leur environnement sans craindre les risques liés à (i) la confidentialité: l'utilisateur doit être maître de choisir la partie de son environnement qu'il souhaite partager et celle qu'il souhaite conserver privée; (ii) la disponibilité du service: le système doit être prévu pour se prémunir des risques de défaillance d'une ressource afin de garantir une continuité du service; (iii) l'authentification: la cohésion d'un groupe réside dans la confiance mutuelle de ses membres. L'authentification doit être suffisamment robuste afin d'éviter les situations désagréables comme l'usurpation d'identité par exemple; (iv) l'intégrité des messages échangés.

Bibliographie

[1] Ahuja, S.R., Ensor, J.R., Lucco, S.E. “A comparison of application sharing mechanisms in real-time desktop conferencing systems” ACM Conference on Supporting Group Work, Proceedings of the conference on Office Information Systems pp.238–248, Cambridge (United States), 1990.

[2] Aslst, W.v.d., Hee, K.v. “Workflow management: models, methods, and systems” MIT Press, Cambridge (USA), 2002.

[3] Alfieri. 2003 (fév.). VOMS: an Authorization System for Virtual Organizations. In: 1st European Across Grids Conference. 60

[4] Andrews, T. Curbera, F. Dholakia, H. Goland, Y. Klein, J. Leymann, F. Liu, K. Roller, D. Smith, D. (Editor), S. T. Trickovic, I. and Weerawarana, S. "Business Process Execution Language for Web Services, Version 1.1," 2003.

[5] Begole, J., Rosson, M.B., Shaffer., C.A. “Flexible collaboration transparency: supporting worker independence in replicated application-sharing systems”. ACM Transactions on Computer-Human Interaction v.6, n.2, pp.95–132, Jun. 1999.

[6] Berlanga, A. et Garcia,F. Sistemas hipermedia adaptativos en el ámbito de la educación. Rapport technique. Salamanca, Espagne: Universidad de Salamanca, 2004.

[7] Bernard, S. “Spécification d’un environnement d’ingénierie collaborative mulitisite-Application à l'industrie aéronautique européenne” Thèse de doctorat Génie Industriel École nationale supérieure d’arts et métiers, Centre d’Aix-en-Provence France, Nov. 2004.

[8] Bruckman. A. “MOOSE Crossing: Construction, Community, and Learning in A Networked Virtual World for Kids” Doctoral Dissertation, MIT Media Lab Cambridge (USA), 1997

[9] Booth, David, Haas, Hugo, McCabe, Francis, Newcomer, Eric, Champion, Michael, Ferris, Chris, & Orchard, David. “Web services architecture”. W3C working group note NOTE-ws-arch-20040211. World Wide Web Consortium. www.w3.org/TR/2004/NOTE-wsarch- 20040211/. 12, 50. February. 2004.

[10] Certeau, M. de (1990). L'invention du quotidien : 1. Arts de faire. Paris : Gallimard

[11] Czajkowski, K. Ferguson, D. F. Foster, I. Frey, J. Graham, S. Sedukhin, I. Snelling, D. Tuecke, S. & Vambenepe, W. “TheWS-Resource Framework”. Whitepaper Ver. 1.0. The Globus Alliance. 50. May. 2004b.

[12] Czajkowski, K. Ferguson, D.F., Foster, I. Frey, J. Graham, S. Maguire, T. Snelling, D. & Tuecke, S. "From Open Grid Services Infrastructure to WS-Resource Framework: refactoring and evolution". Whitepaper Ver. 1.0. The Globus Alliance. 50. May. 2004a.

[13] Cabrera, L. F. Copeland, G. Feingold, M. Freund, R. W. Freund, T. Johnson, J. Joyce, S. Kaler, C. Klein, J. Langworthy, D. Little, M. Nadalin, A. Newcomer, E. Orchard, D. Robinson, I. Shewchuk, J and Storey, T. "Web Services Coordination (WSCoordination), Version 1.0,". August 2005.

[14] Dix, A. "Challenges for Cooperative Work on the Web: An analytical approach" Computer Supported Cooperative Work: The Journal of Collaborative Computing, Kluwer Academic Publishers v.6, n.2-3, pp.135-156, 1997. ISSN 0925-9724

[15] Dommel, H. P. and Garcia Luna Aceves, J. J., "Group Coordination Support for Internet collaboration". In: IEEE Internet Computing Magazine, Special Issue on Multimedia and Collaborative Computing over the Internet, Vol. 3, No. 2, pp. 74-80. Mars-Avril, 1999.

[16] Dourish, P., Bellotti, V. "Awareness and coordination in shared workspaces" ACM Conference on Computer Supported Cooperative Work (CSCW'92), ACM Press pp.107–114, Toronto (Canada), 1992. ISBN 0-89791-542-9.

[17] Drira, K., Martelli, A. and Villemur, T. (Eds.). "Cooperative Environments for Distributed Systems Engineering". The Distributed System Engineering Report, State of the Art Survey. In: Lecture Notes in Computer Sciences, Vol. 2236, N°ISBN 3-540-43083-0, Springer, 2001.

[18] Dugénie, P. "Orientation et usage de l'architecture de services Grille OGSA". Pages 283–290 of : Vautier, Alexandre, & Saget, Sylvie (eds), Manifestation des Jeunes Chercheurs francophones dans les domaines des STIC, MajecSTIC 2005. IRISA - IETR - LTSI. hal.inria.fr/inria-00000680. 14, 56, 61. 2005.

[19] Duvert, F. Jonquet, C. Dugénie, P & Cerri, S. A. "Agent-Grid Integration Ontology". Pages 136–146 of: Meersman, R., Tari, Z., & Herrero, P. (eds), International Workshop on Agents, Web Services and Ontologies Merging, AWeSOMe'06. Lecture Notes in Computer Science, vol. 4277, no. 1. hallirmm.ccsd.cnrs.fr/lirmm-00110569. 62. Novembre. 2006.

[20] Ellis, C.A. "Workflow Technology" Computer-Supported Cooperative Work, Trends in Software (Beaudouin- Lafon eds.) Pages 29-54, John Wiley & Sons

[21] Ellis, C., Gibbs, S. et Rein, G. Groupware. Some issues and experiences. Communications of the ACM, 34(1):38-58, Janvier 1991.

[22] Ellis, C.A., and Wainer, J. "A Conceptual Model of Groupware" ACM Conference on Computer Supported Cooperative Work (CSCW'94), ACM press pp.79-88, Chapel Hill (USA), 1994. ISBN:0-89791-689-1

[23] Foster, I. 2002. What is the Grid? A Three Point Checklist. GRID Today. 44

[24] Foster, I. Kesselman, C. Nick, J. & Tuecke, S. 2002 (June). The physiology of the Grid: an Open Grid Services Architecture for distributed systems integration. In: Open Grid Service Infrastructure WG, Global Grid Forum. The Globus Alliance. 50, 61

[25] Foster, I. Maguire, T., & Snelling, D. 2005. OGSA-WG recommendation: OGSA WSRF basic profile. Recommendation 1.0. Open Grid Forum. 50

[26] Foster, I. Kesselman, C. Tsudik, G. & Tuecke, S. 1998. A security architecture for computational Grids. Pages 83–92 of: 5th ACM conference on Computer and communications security. San Francisco, CA, USA: ACM Press. 50, 57

[27] Foster, I. Kesselman, C. & Tuecke, S. 2001. The anatomy of the Grid: enabling scalable virtual organizations. Supercomputer Applications, 15(3), 200–222. 59, 61

[28] Foster, I. & Kesselman, C (eds). 2003. The Grid 2 : blueprint for a new computing infrastructure. San Francisco, CA, USA: Morgan Kaufmann. 61

[29] Foster, I. Kesselman, C. Nick, J. & Tuecke, S. 2002 (June). The physiology of the Grid: an Open Grid Services Architecture for distributed systems integration. In: Open Grid Service Infrastructure WG, Global Grid Forum. The Glob us Alliance. 50, 61

[30] Graham, T.C.N., Grundy, J. "External Requirements of Groupware Development Tools" Engineering for Human-Computer Interaction, Kluwer Academic Press pp.363-376, Heraklion (Grèce), 1999.

[31] Grudin, J. "CSCW: History and Focus". IEEE Computer, IEEE Computer Society Press v.27, n.5, pp.19-26, 1994.

[32] Gutwin, C., Greenberg, S. "A Descriptive Framework of Workspace Awareness for Real-Time Groupware" Computer Supported Cooperative Work: The Journal of Collaborative Computing, Kluwer Academic Publishers v.11, n.3-4, pp.411-446, Jan. 2002. ISSN 0925-9724

[33] Haake, J. et Wang, W. (1998). « Collaboration support in open hypermedia environments », in Wiil, U. (ed.) 4th. Workshop on Open Hypermedia Systems. Rapport technique. Esbjerg, Danemark: Aalborg University. pp. 39-44.

[34] Johansen, R. "Groupware: Computer Support for Business Teams" The Free Press 205p, 1988. ISBN 0-02-916491-5

[35] Jonquet, C. Dugénie, P. & Cerri, S. A. 2007. Agent-Grid Integration Language. Multiagent and Grid Systems, (à paraître). hal-lirmm.ccsd.cnrs.fr/lirmm-00139691. 14, 56, 62

[36] Jonquet, C. 2006. Dynamic Service Generation: Agent interaction for service exchange on the GRID. Ph.D. thesis, Université de Montpellier II, Montpellier. 62

[37] Kam, M. "Livenotes: a system for cooperative and augmented note-taking in lectures" ACM Conference on Human Factors in Computing Systems, ACM press pp.531-540, Portland (USA), 2005 ISBN 1-58113-998-5.

[38] Karsenty, A. "Le collecticiel : de l'interaction homme-machine à la communication homme-machine-homme" Technique et Science Informatiques v.13, n.1, pp.105-127, 1994.

[39] Kilgore, R. Chignell, M. Smith, P. "Spatialized audioconferencing: what are the benefits?" Conference of the Centre for Advanced Studies on Collaborative research IBM Centre for Advanced Studies Conference, IBM press. pp.135-144, Toronto (Canada), Oct. 2003

[40] Kraemer, K. L et King, J. L. Computer-Based Systems for Cooperative Work and Group Decision Making. ACM Computing Surveys, 20(2):115-146, Juin 1988.

[41] Laurillau, Y. "Conception et réalisation logicielles pour les collecticiels centrées sur l'activité de groupe : le modèle et la plate-forme Clover" Thèse de doctorat Informatique Université Joseph Fourier - Grenoble I, Grenoble, France, Sep. 2002

[42] Lévy, P. Les technologies de l'intelligence, Éditions La découverte 233pp, Jan. 1990.

[43] Lonchamp, J. Le travail coopératif et ses technologies. Hermès Science, 320 pages, 2003

[44] Lonchamp, J. Le travail coopératif et ses technologies, Hermès Lavoisier, 318pp, 2003 ISBN 2-7462-0668-4

[45] Lorcy, S. “Infrastructure logicielle pour la gestion de la cohérence et de la qualité de service d'un environnement à objets réparti : application au télétravail coopératif” Thèse de doctorat Informatique Université de Rennes 1, Rennes France, Jan. 2000

[46] Maesano, L. Bernard, C. and X. Galles, L. Services Web avec J2EE et .NET (Conception et Implementation): Edition Eyrolles, 2003.

[47] Mora, M., Forgionne, G.A., Gupta, J.N.D. Decision making support systems: achievements, trends, and challenges for the new decade, Idea Group Pub 418 pp., Hershey (USA), 2002 ISBN 1-59140-045-7

[48] Munson, J., Dewan, P. “A concurrency control framework for collaborative systems” ACM conference on Computer supported cooperative work (CSCW’96), ACM press pp.278-287, Boston (USA), 1996. ISBN:0-89791-765-0

[49] O'Reilly, T. (2005). What is Web 2.0: Design patterns and business models for the next generation of software. En ligne : http://www.oreillynet.com/lpt/a/6228

[50] OASIS, "Universal Description, Discovery, and Integration (UDDI) - Version 3," UDDI Spec Technical Committee Draft, http://uddi.org/pubs/uddi_v3.htm, October 2004.

[51] Palen, L. Grudin, J. “Discretionary adoption of group support software: lessons from calendar applications” Implementing collaboration technologies in industry: case examples and lessons learned (Bjørn Erik Munkvold ed.), Springer-Verlag pp.159-179, 2003. ISBN 1-85233-418-5.

[52] Prakash, A “Group Editors” Computer-Supported Cooperative Work, Trends in Software,
(Beaudouin-Lafon eds.), John Wiley & Sons pp.103-133, 1999

[53] Pearlman, L. Welch, V. Foster, I. Kesselman, C. & Tuecke, S. 2002. A Community Authorization Service for group collaboration. Pages 50–59 of: 3rd International Workshop on Policies for Distributed Systems and Networks, POLICY’02. Monterey, CA, USA : IEEE Computer Society. 57, 60

[54] Reyes García, E. "L'objet technique hypermédia : repenser la création de contenu éducatif sur le Web" Thèse de doctorat Informatique Université de PARIS VIII France, Fév. 2007.

[55] Richardson, T.; Stafford-Fraser, Q.; Wood, K.R. et A. “Virtual Network Computing” IEEE Internet Computing, IEEE Computer Society Press v.2, n.1, Jan./Feb. 1998

[56] Saleh, I., A. Mkadmi et E. Reyes (2005). "L'hypermédia au service du travail collaboratif", in Saleh, I. (ed.) Les hypermédias : conception et réalisation. Paris : Hermès Lavoisier, pp.61-91

[57] Schmidt, K., Rodden, T. “Putting it all together: Requirements for a CSCW platform” The Design of Computer Supported Cooperative Work and Groupware Systems (Shapiro, D., Tauber, M., Traünmuller, R. eds.) pp.157-176, Amsterdam (Netherlands),1996.

[58] Singh, Munindar P., & Huhns, Michael N. 2005. Service-Oriented Computing, Semantics, processes, agents. John Wiley & Sons. 53, 54, 62, 63

[59] Talia, D. 2002. The open Grid services architecture - where the Grid meets the Web. Internet Computing, 6(6), 67–71. 44

[60] Tang, J.C., Isaacs, E. “Why do users like video? Studies of multimedia-supported collaboration” Computer Supported Cooperative Work: The Journal of Collaborative Computing, Kluwer Academic Publishers v.1, n.3, pp.163-196, Sep. 1992. ISSN 0925-9724

[61] Terzis, S., Nixon P. “Building the next generation groupware: A survey of groupware and its impact on the virtual enterprise”. Technical Report TCD-CS-1999-08, Department of Computer Science, Trinity College Dublin (Ireland), 1999 https://www.cs.tcd.ie/publications/tech-reports/tr-index.99.html

[62] Villemur, T. "Modèles Et Services Logiciels Pour Le Travail Collaboratif". Thèse pour l’habilitation à diriger des recherches de l’université Paul Sabatier de Toulouse. Informatique. LABORATOIRE D'ANALYSE ET D'ARCHITECTURE DES SYSTÈMES DU CNRS. France, pp. 27-47, 2006.

[63] Yang, Y., Li, D. “Separating data and control: support for adaptable consistency protocols in collaborative systems”. ACM Conference on Computer Supported Cooperative Work (CSCW’04), ACM press. pp.11-20, 2004. ISBN:1-58113-810-5.

[64] CCITT. 1988. Recommandation X509. The directory authentication framework. 58

[65] Grand dictionnaire terminologique http://www.olf.gouv.qc.ca/ressources/gdt_bdl2.html

[66] Projet DIANE. – http://www.silogic.fr/diane/Default2.htm

[67] http://www.activeworlds.com/

[68] http://www.asti.asso.fr/

[69] CA, Certification Authority. url. Autorité de Certification Grid Canada. www.gridcanada.ca/ca. 58

[70] DataGrid. url. Projet EDG, the European Data Grid. eu-datagrid.web.cern.ch. 46, 60

[71] DataTag. url. Projet DataTag. datatag.web.cern.ch. 60

[72] http://developer.blaxxun.com/

[73] W3C, "Extensible Markup Language (XML) 1.0 (Third Edition)," W3C Recommendation, http://www.w3.org/TR/REC-xml/, 04 February 2004.

[74] Linux40TF. url. Linux va tourner à 40 teraFlops. www.pcinpact.com/actu/news/Europe_Linux_va_tourner_a_40_teraflops.htm. 46

[75] MD5. url. L’algorithme de hachage Message Digest. fr.wikipedia.org/wiki/Md5. 58

[76] W3C, "Simple Object Access Protocol (SOAP) 1.2," http://www.w3.org/TR/soap/, 2003.

[77] W3C, "SOAP Version 1.2 Part 0: Primer."

[78] W3C, "SOAP Version 1.2 Specification Assertions and Test Collection."

[79] SSL. url. Secure Sockets Layer. www.openssl.org. 58

[80] TLS. url. Transport Layer Security. www.ietf.org/html.charters/tls-charter.html. 58

[81] W3C, "Web Services Definition Language (WSDL) 1.1," http://www.w3.org/TR/wsdl, 2001.

[82] W3C, "Web Services Definition Language (WSDL) 1.2," W3C Working Draft,

[83] W3C, "XML Schema," 28 Octobre 2004.

Annexe

Techniques d'implémentation de l'intergiciel Globus et de la plateforme Sakai

1. La mise en œuvre de l'intergiciel Globus (GT 4.0.6) sous Linux Fedora Core version 4

1.1 Présentation

Dans cette section, nous allons présenter les principales étapes de mise en œuvre de l'intergiciel Globus dans sa version 4.0.6, sous le système LINUX Fedora core4.

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

Une mise en œuvre d'un nœud de grille de calcule, consiste à préparer d'abord l'environnement, l'initialiser, installer l'intergiciel, et enfin, configurer les services de cet intergiciel et notamment sa personnalisation.

Dans notre cadre d'étude, nous allons mettre en place un réseau LAN constitué de quatre machines, sur chacune est installé le système Linux Fedora core 4.

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

1.2 Mise en place du réseau

a. Quelques caractéristiques des machines utilisées

- Fabriquant : Intel.
- Processeur: Intel(R) Pentium(R) 4 CPU 3.00GHZ (2CPUs).
- Mémoire: 1 GB RAM.

b. Installation du système linux

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

- Type d'installation poste de travail (WorkStation).
- Désactivation de pare-feu pour ne pas gêner le fonctionnement des services de l'intergitiel.
- Date et heure doivent être réglées pour chaque machine, cette condition est indispensable dans la phase de signature de certificat pour les autres machines afin de vérifier la validité du proxy.

c. Configuration du réseau

Lors de l'installation de linux, nous avons attribué un nom et une adresse IP à chaque noeud de la grille qui vont servir par la suite à la configuration de l'intergiciel globus. Chaque nom du hôte doit avoir la forme suivante:'nom_machine.nom_domaine' (une exigence de l'intergiciel globus toolkit).

Dans cette illustration la grille est configurée de la manière suivante:

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

d. outils nécessaires

- **Apache Ant :** c'est un exécuteur de taches il permet le déploiement des programmes (déploiement de services)
- **JDK :** nécessaire pour la compilation du code de globus car la plus grande partie est écrite en java.
- **PostgresSQL :** système de gestion de base de données relationnelle fonctionnant sur des systèmes UNIX. Il est composé de deux parties :

1. Partie serveur : c'est la partie fonctionnant sur la machine hébergeant la base de données capable de traiter les requêtes des clients

2. Partie client : cette partie est installée sur les postes client. Les clients interrogent le serveur de base de données par des requêtes SQL.

- **Globus Toolkit :** Une boite à outils qui contient des fichiers et des Ordonnanceurs et d'autres outils nécessaires pour faire le calcul distribué, assurer la sécurité …

1.3 Préparation de l'installation de globus toolkit 4.0.6

1.3.1 Création des comptes utilisateurs

Afin d'installer globus et pouvoir tester son bon fonctionnement, nous devons créer trois types d'utilisateurs: l'administrateur du système de ce poste qui permet le lancement et l'arrêt du container, c'est un utilisateur non privilégié par apport à root qui sera créé automatiquement lors de l'installation du système Linux et qui possède tous les droits, l'utilisateur 'globus' qui servira à l'installation de l'intergiciel globus et un utilisateur simple 'user'.

La création des utilisateurs peut se faire lors de l'installation de linux ou à l'aide de la commande 'adduser', ou même avec le gestionnaire des groupes et utilisateurs sous Fedora.

1.3.2 Création des répertoires d'installation

Nous créons deux répertoires dans chaque noeud de la grille, sous le répertoire '/usr/local' un pour globus et l'autre pour les outils (jdk, ant...).

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

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

2. Création du répertoire d'installation des outils 'outils' sous '/usr/local', de la même façon.

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

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

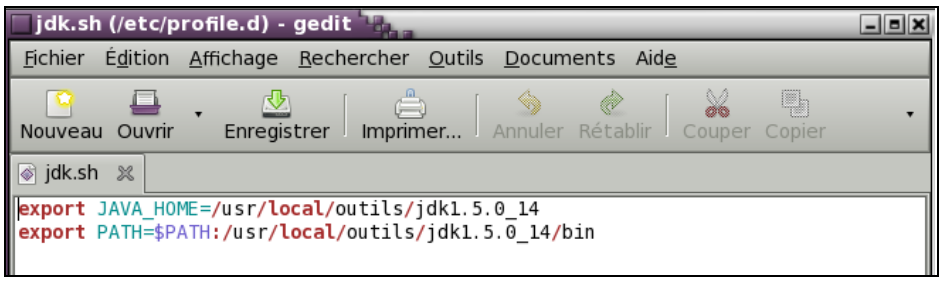
1.3.3 Installation des outils apache Java, Apache-ant et Postgresql

a. Installation de Java

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

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

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



Une vue du fichier */etc/profile.d/jdk.sh*

Remarque : Il faut supprimer le lien vers l'ancienne version de Java représenté par le fichier raccourci 'java' trouvée sous le chemin /usr/bin, et nous pouvons tester la réussite de l'installation du JDK en tapant la commande :

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

Le résultat donc sera :

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

b. Installation de Apache ant

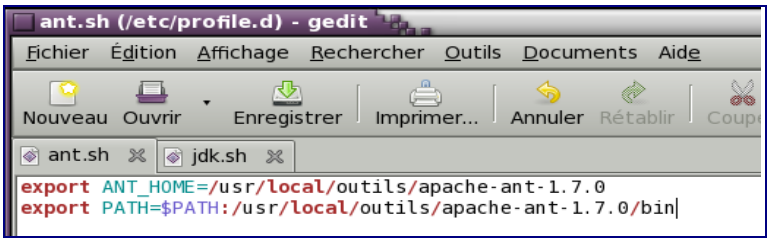
Apache ant est un exécuteur de tâches permettant de compiler et déployer des programmes, après le 'téléchargement de la version de Apache ant 'appache-ant-1.7.0-bin.tar.gz', on l'installe dans le répertoire '/usr/local/outils'.

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

La décompression se fait par :

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

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



Une vue du fichier */etc/profile.d/ant.sh*

Nous devons tester si apache ant est bien installé en utilisant la commande suivante:

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

Le résultat sera comme suit :

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

c. Installation de postgresql :

Afin d'installer Postgresql, on télécharge le fichier postgresql-8.2.6.tar.gz, on le décompresse dans '/usr/local/' :

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

On lance la commande suivante:

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

On lance l'installation:

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

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

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

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

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

On lance le serveur de base de données :

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

Le résultat sera:

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

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

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

Le résultat de cette commande est :

```
CREATE DATABASE
```

Nous testons la création de la base par cette commande :

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

Le résultat est:

```
Welcome to psql 8.1.6, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit

test=#
```

1.4 L'installation de globus toolkit

1.4.1 Lancement du script d'installation

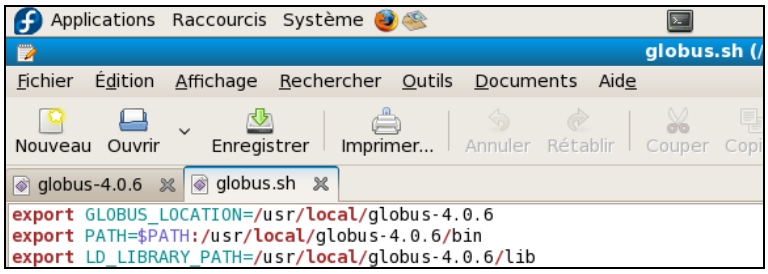
La version téléchargée de globus toolkit est 'gt4.0.6-x86_fc_4-installer.tar.gz'. Nous commençons par la décompression du fichier dans '/usr/local/globus-4.0.6', copier les fichiers décompressés dans le même répertoire, ajouter la variable $GLOBUS_LOCATION au path et ensuite lancer l'opération d'installation :

a. Décompresser et copier :

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

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

b. Aperçu sur le fichier '/etc/profile.d/globus.sh'



Une vue du fichier */etc/profile.d/globus.sh*

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

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

d. L'installation de globus toolkit doit être faite sous l'utilisateur globus, nous créons un fichier Makefile avec la commande ./configure :

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

Le résultat sera :

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

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

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

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

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

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

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

e. Construire le programme exécutable avec la commande make, nous utilisons tee pour garder la trace de l'exécution des commandes d'installation, afin d'aider à localiser les erreurs qui peuvent se produire durant l'installation :

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

Une partie du résultat de cette commande sera comme suit :

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

f. Lancer l'installation de globus avec la commande make install :

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

Une partie du résultat est la suivante :

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

```
checking for mpirun... no
find-fork-tools: creating ./config.status
config.status: creating fork.pm
..Done
```

1.4.2 L'installation de l'Autorité de Certification (CA)

L'installation de l'autorité de certification doit se faire sur une seule machine qu'on appel serveur de certificat, cette étape est très critique pour le reste des étapes d'installation et de configuration de différents services de l'intergiciel.

Étapes d'installation

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

1.4.2.1 Exécution du script d'installation

a) Nous rajoutons au fichier «hosts» qui se trouve dans le répertoire '/etc/' le nom de la machine et l'adresse IP correspondante de la façon suivante: 192.168.0.101 poste4.lri.net poste4

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

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

c) Sous globus, nous lançons l'exécution du script :

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

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

```
WARNING: GPT_LOCATION not set, assuming:
      GPT_LOCATION=/usr/local/globus-4.0.6
  C e r t i f i c a t e  A u t h o r i t y  S e t u p
This script will setup a Certificate Authority for signing Globus
users certificates.  It will also generate a simple CA package
that can be distributed to the users of the CA.
The CA information about the certificates it distributes will
be kept in:
```

```
/home/globus/.globus/simpleCA/
The unique subject name for this CA is:
cn=Globus Simple CA, ou=simpleCA-poste4.lri.net, ou=GlobusTest, o=Grid
Do you want to keep this as the CA subject (y/n) [y]:y
Enter the email of the CA (this is the email where certificate
requests will be sent to be signed by the CA):seidali.rehab@lri-annaba.net
The CA certificate has an expiration date. Keep in mind that
once the CA certificate has expired, all the certificates
signed by that CA become invalid.  A CA should regenerate
the CA certificate and start re-issuing ca-setup packages
before the actual CA certificate expires.  This can be done
by re-running this setup script.  Enter the number of DAYS
the CA certificate should last before it expires.
[default: 5 years (1825 days)]:1825
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
creating CA config package...done.
A self-signed certificate has been generated
for the Certificate Authority with the subject:
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA
If this is invalid, rerun this script
/usr/local/globus-4.0.6/setup/globus/setup-simple-ca
and enter the appropriate fields.
-----------------------------------------------------------------
The private key of the CA is stored in /home/globus/.globus/simpleCA//private/cakey.pem
The public CA certificate is stored in /home/globus/.globus/simpleCA//cacert.pem
The distribution package built for this CA is stored in
/home/globus/.globus/simpleCA//globus_simple_ca_637244ab_setup-0.19.tar.gz
This file must be distributed to any host wishing to request
certificates from this CA.
CA setup complete.
The following commands will now be run to setup the security
configuration files for this CA:
```

```
$GLOBUS_LOCATION/sbin/gpt-build
/home/globus/.globus/simpleCA//globus_simple_ca_637244ab_setup-0.19.tar.gz
$GLOBUS_LOCATION/sbin/gpt-postinstall
------------------------------------------------------------
setup-ssl-utils: Configuring ssl-utils package
Running setup-ssl-utils-sh-scripts...
************************************************************************
Note: To complete setup of the GSI software you need to run the
following script as root to configure your security configuration
directory:
/usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/setup-gsi
For further information on using the setup-gsi script, use the -help
option.  The -default option sets this security configuration to be
the default, and -nonroot can be used on systems where root access is
not available.
************************************************************************
setup-ssl-utils: Complete
```

1.4.2.2 Installation du service GSI

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

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

Le résultat est le suivant :

```
setup-gsi: Configuring GSI security
Making trusted certs directory: /etc/grid-security/certificates/
mkdir /etc/grid-security/certificates/
Installing /etc/grid-security/certificates//grid-security.conf.637244ab...
Running grid-security-config...
Installing Globus CA certificate into trusted CA certificate directory...
Installing Globus CA signing policy into trusted CA certificate directory...
setup-gsi: Complete
[globus@poste4 globus-4.0.6]$ grid-cert-request -ca
nondefaultca=true
The available CA configurations installed on this host are:
```

1) 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA

Enter the index number of the CA you want to sign your cert request: 1

Using CA: 637244ab - /O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=Globus Simple CA

A certificate request and private key is being created.

You will be asked to enter a PEM pass phrase.

This pass phrase is akin to your account password,

and is used to protect your key file.

If you forget your pass phrase, you will need to

obtain a new certificate.

Generating a 1024 bit RSA private key

.....++++++

.....++++++

writing new private key to '/home/globus/.globus/userkey.pem'

Enter PEM pass phrase:

Verifying - Enter PEM pass phrase:

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:

A private key and a certificate request has been generated with the subject:

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

If the CN=globus is not appropriate, rerun this

script with the -force -cn "Common Name" options.

Your private key is stored in /home/globus/.globus/userkey.pem

Your request is stored in /home/globus/.globus/usercert_request.pem

Please e-mail the request to the Globus Simple CA seidali.rehab@lri-annaba.net

You may use a command similar to the following:

```
cat /home/globus/.globus/usercert_request.pem | mail seidali.rehab@lri-annaba.net
Only use the above if this machine can send AND receive e-mail. if not, please
mail using some other method.
Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Simple CA at seidali.rehab@lri-annaba.net
```

1.4.2.3 Génération du certificat pour le hôte

a. Demande certificat pour un nœud hôte 'host certificates'

Se fait par la commande suivante sous root:

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

Cette commande nous donne comme résultat trois fichiers en sortie dans '/etc/grid-security', nous remarquons bien que dans le résultat, nous trouvons que le fichier 'hostcert.pem' qui va porter le certificat valide et de taille zéro, en faite, ce fichier va contenir le résultat de signature du fichier portant la demande de certificat 'hostcert_request.pem' :

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

b. Signature du certificat «host certificates»

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

Nous procédons à la signature du certificat de l'hôte :

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

Le résultat sera :

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

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

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

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

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

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

1.4.2.4 Génération du certificat pour l'utilisateur globus

a. Demande du certificat de l'utilisateur globus:

Avec la commande suivante:

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

Le résultat de cette commande est comme suit :

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

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

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

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

b. Signature du certificat de l'utilisateur globus:

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

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

Le résultat est:

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

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

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

```
-r-------- 1 globus globus  963 jui 20 22:47 userkey.pem
```

1.4.2.5 Génération du certificat pour l'utilisateur user

a. demande du certificat de l'utilisateur ‘user’:

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

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

Le résultat sera :

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

```
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit
[simpleCA-poste4.lri.net]:Level 2 Organizational Unit [lri.net]:Name (e.g., John M. Smith) []:


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

b. signature du certificat de l'utilisateur user:

Nous signons le fichier ‘usercert_request.pem’ avec la commande suivante sous globus.

```
[user@poste4 ~]$ cp /home/user/.globus/usercert_request.pem /tmp/user_usercertrequest.pem
[globus@poste4 ~]$ grid-ca-sign -in /tmp/user_usercertrequest.pem -out /tmp/usersigned2.pem
[user@poste4 ~]$ cp /tmp/usersigned2.pem /home/user/.globus/
[user@poste4 ~]$ mv /home/user/.globus/usersigned2.pem /home/user/.globus/usercert.pem
[root@poste4 ~]# chmod 644 /home/user/.globus/usercert.pem
```

Nous vérifions le contenu de ‘/home/user/.globus/’ :

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

1.4.2.6 Création du certificat du 'container'

Il est pratiquement nécessaire pour le lancement des services web, nous procédons comme suit :

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

1.4.2.7 Ajout des autorisations

a. Ajout des autorisations pour l'utilisateur globus :

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

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

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

```
[globus@poste4]$ grid-cert-info -subject
/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/OU=lri.net/CN=globus
[globus@poste4]$ whoami
globus
```

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

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

Le résultat sera :

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

b. Ajout des autorisations pour l'utilisateur user :

Nous procédons de la même manière :

Pour récupérer le sujet du certificat et son propriétaire, nous exécutons :

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

Le résultat sera :

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

Aprés, nous lançons cette commande:

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

Résultat:

```
user
```

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

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

Résultat:

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

1.4.2.8 Vérification des certificats des utilisateurs

Pour la vérification du certificat de l'utilisateur globus, nous utilisons la commande suivante:

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

Le résultat est le suivant:

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

De même, sous l'utilisateur user :

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

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

1.4.3 Installation de certificat pour plusieurs machines

Le certificat sera installé sur les machines autres que celle où nous avons installé le certificat d'autorité.

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

```
[root@poste4 sbin]# scp
/home/globus/.globus/simpleCA/globus_simple_ca_637244ab_setup-
0.19.tar.gz root@poste.lri.net/home/globus/

[globus@poste ~]$ $GLOBUS_LOCATION/sbin/gpt-build
globus_simple_ca_637244ab_setup-0.19.tar.gz gcc32dbg
```

Nous obtenons :

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

```
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-doc
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-pgm_static
gpt-build ====> REMOVING empty package globus_simple_ca_637244ab_setup-noflavor-rtl
```

Puis la commande:

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

Le résultat sera:

```
running /usr/local/globus-4.0.6//setup/globus/./setup-ssl-utils.637244ab..[ Changing to /usr/local/globus-4.0.6/setup/globus/. ]
setup-ssl-utils: Configuring ssl-utils package
Running setup-ssl-utils-sh-scripts...
************************************************************************
Note: To complete setup of the GSI software you need to run the
following script as root to configure your security configuration
directory:
/usr/local/globus-4.0.6//setup/globus_simple_ca_637244ab_setup/setup-gsi
For further information on using the setup-gsi script, use the -help
option.  The -default option sets this security configuration to be
the default, and -nonroot can be used on systems where root access is
not available.
************************************************************************
setup-ssl-utils: Complete
..Done
WARNING: The following packages were not set up correctly:
        globus_simple_ca_637244ab_setup-noflavor-pgm
Check the package documentation or run postinstall -verbose to see what happened
```

Maintenant nous installons le gsi par la commande suivante:

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

Le résultat :

```
bash: /usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/etup-gsi: Aucun fichier ou répertoire de ce type
[root@poste ~]# /usr/local/globus-4.0.6/setup/globus_simple_ca_637244ab_setup/setup-gsi -default
setup-gsi: Configuring GSI security
```

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

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

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

Et nous ajoutons ce qui suit :

192.168.0.103 poste2.lri.net

192.168.0.104 poste3.lri.net

192.168.0.101 poste4.lri.net

Puis, nous lançons la demande du certificat pour l'hôte du 'poste' :

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

Le résultat :

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

Level 0 Organization [Grid]:Level 0 Organizational Unit [GlobusTest]:Level 1 Organizational Unit [simpleCA-poste4.lri.net]:Name (e.g., John M. Smith) []:

A private host key and a certificate request has been generated

with the subject:

/O=Grid/OU=GlobusTest/OU=simpleCA-poste4.lri.net/CN=host/poste.lri.net

---

The private key is stored in /etc/grid-security/hostkey.pem

The request is stored in /etc/grid-security/hostcert_request.pem

Please e-mail the request to the Globus Simple CA seidali.rehab@lri-annaba.net

You may use a command similar to the following:

cat /etc/grid-security/hostcert_request.pem | mail seidali.rehab@lri-annaba.net

Only use the above if this machine can send AND receive e-mail. if not, please

mail using some other method.

Your certificate will be mailed to you within two working days.

If you receive no response, contact Globus Simple CA at seidali.rehab@lri-annaba.net

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

-rw-r--r-- 1 root root 0 jui 21 10:20 hostcert.pem

-rw-r--r-- 1 root root 1406 jui 21 10:20 hostcert_request.pem

-r-------- 1 root root 887 jui 21 10:20 hostkey.pem

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

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

Le résultat sera :

To sign the request

please enter the password for the CA key:

The new signed certificate is at: /home/globus/.globus/simpleCA//newcerts/05.pem

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

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

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

De même, nous demandons le certificat pour les autres utilisateurs de la machine 'poste'

On lance la commande suivante sous l'utilisateur globus de la seconde machine :

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

Résultat:

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

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

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

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

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

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

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

Le résultat sera:

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

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

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

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

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

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

Résultat:

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

De même, on demande le certificat pour le deuxième utilisateur ‘seidali’ :

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

Le résultat:

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

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

Visualisation des trois fichiers '.pem' :

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

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

```
-r-------- 1 seidali seidali  963 jui 21 10:25 userkey.pem
```

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

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

Et nous lançons la commande de signature:

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

Le résultat sera:

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

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

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

Visualisation des propriétés du fichier usercert.pem :

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

Résultat:

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

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

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

Résultat:

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

```
Your proxy is valid until: Mon Jul 21 22:55:09 2008
```

1.4.4 Ajout des autorisations

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

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

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

Et pour l'utilisateur seidali comme suit

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

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

1.4.5 Ajout des autorisations inter-postes

Cette étape se traduit par l'ajout des entrées pour chaque utilisateur d'un poste i dans un autre j, on donne un exemple qui permet à l'utilisateur 'globus' de la machine 'poste' d'être reliée avec le poste 4, on ajoute au map du poste 4 l'entrée composée du sujet de certificat de l'utilisateur globus de l'hote 'poste' et de l'utilisateur globus de l'hote 'poste4'

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

De l'autre coté, on lance :

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

1.5 Service 'gridFTP'

1.5.1 Configuration du service 'gridFTP'

Nous configurons le service gridFTP pour qu'il se lance au démarrage, et pour se faire, nous utilisons xinetd ou inetd, dans notre cas, nous avons utilisé xinetd, si nous n'avons pas xinetd installé sur la machine, nous lançons son installation comme suit:

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

Résultat:

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

```
Installed: xinetd.i386 2:2.3.14-14.fc8

Complete!
```

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

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

Et nous écrivons le contenu suivant:

```
service gsiftp
{
instances = 100
socket_type = stream
wait = no
user = root
env += GLOBUS_LOCATION = /usr/local/globus-4.0.6
env += LD_LIBRARY_PATH=/usr/local/globus-4.0.6/lib
server = /usr/local/globus-4.0.1/sbin/globus-gridftp-server
server_args = -i   root     70 jui 15 14:59 gridftp.conf
log_on_success += DURATION
nice = 10
disable = no
}
```

Nous lançons ou nous relançons xinetd comme suit:

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

Résultat:

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

Ou avec

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

Résultat:

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

Nous ajoutons le service gsiftp à l'ensemble des services locaux, au niveau du fichier '/etc/services' :

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

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

Une visualisation avec la commande 'tail' est possible:

```
[root@poste4 ~]# tail /etc/services
iqobject        48619/udp                  # iqobject
vboxd           20012/udp
```

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

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

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

Et nous mettrons le contenu suivant:

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

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

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

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

Le résultat sera:

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

1.5.2 Lancement du service 'gridFTP'

De même, nous pouvons lancer manuellement le serveur gridFTP comme suit:

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

Résultat:

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

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

Exemple:

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

-Maintenant, nous pouvons utiliser le protocole gsiftp pour l'envoi des fichiers:

Exemple :

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

1.6 Lancement du container des services web

Nous créons le script permettant le lancement du container, sous globus, nous exécutons la commande suivante:

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

Et nous plaçons le contenu suivant:

```
#!/bin/sh
set -e
export GLOBUS_OPTIONS="-Xms256M -Xmx512M"
sh $GLOBUS_LOCATION/etc/globus-user-env.sh
cd $GLOBUS_LOCATION
case "$1" in
start)
$GLOBUS_LOCATION/sbin/globus-start-container-detached -p 8443
;;
stop)
$GLOBUS_LOCATION/sbin/globus-start-container-detached
;;
*)
echo "Usage : globus start|stop">&2
exit 1
;;
esac
exit 0
```

Nous donnons le droit d'exécution pour ce fichier avec la commande :

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

Sous root, nous créons un deuxième script sous '/etc/init.d/' qui va être utilisé et qui va faire appel au premier.

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

Et nous écrivons le contenu suivant:

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

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

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

Maintenant, nous lançons le container comme suit:

```
[root@poste4 ~]# /etc/init.d/globus-4.0.6 start
Starting Globus container. PID: 3554
```

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

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

Le résultat sera :

```
2008-07-13 11:35:37,100 ERROR service.ReliableFileTransferImpl [main,<init>:69] Unable to setup database driver with pooling.Connection refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections.
```

2008-07-13 11:35:37,550 WARN service.ReliableFileTransferHome [main,initialize:97] All RFT requests will fail and all GRAM jobs that require file staging will fail.Connection refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections.

Starting SOAP server at: https://192.168.0.103:8443/wsrf/services/

With the following services:

[1]: https://192.168.0.103:8443/wsrf/services/AdminService

[2]: https://192.168.0.103:8443/wsrf/services/AuthzCalloutTestService

[3]: https://192.168.0.103:8443/wsrf/services/CASService

[4]: https://192.168.0.103:8443/wsrf/services/ContainerRegistryEntryService

[5]: https://192.168.0.103:8443/wsrf/services/ContainerRegistryService

[6]: https://192.168.0.103:8443/wsrf/services/CounterService

[7]: https://192.168.0.103:8443/wsrf/services/DefaultIndexService

[8]: https://192.168.0.103:8443/wsrf/services/DefaultIndexServiceEntry

[9]: https://192.168.0.103:8443/wsrf/services/DefaultTriggerService

[10]: https://192.168.0.103:8443/wsrf/services/DefaultTriggerServiceEntry

[11]: https://192.168.0.103:8443/wsrf/services/DelegationFactoryService

[12]: https://192.168.0.103:8443/wsrf/services/DelegationService

[13]: https://192.168.0.103:8443/wsrf/services/DelegationTestService

[14]: https://192.168.0.103:8443/wsrf/services/InMemoryServiceGroup

[15]: https://192.168.0.103:8443/wsrf/services/InMemoryServiceGroupEntry

[16]: https://192.168.0.103:8443/wsrf/services/InMemoryServiceGroupFactory

[17]: https://192.168.0.103:8443/wsrf/services/IndexFactoryService

[18]: https://192.168.0.103:8443/wsrf/services/IndexService

[19]: https://192.168.0.103:8443/wsrf/services/IndexServiceEntry

[20]: https://192.168.0.103:8443/wsrf/services/ManagedExecutableJobService

[21]: https://192.168.0.103:8443/wsrf/services/ManagedJobFactoryService

[22]: https://192.168.0.103:8443/wsrf/services/ManagedMultiJobService

[23]: https://192.168.0.103:8443/wsrf/services/ManagementService

[24]: https://192.168.0.103:8443/wsrf/services/NotificationConsumerFactoryService

[25]: https://192.168.0.103:8443/wsrf/services/NotificationConsumerService

[26]: https://192.168.0.103:8443/wsrf/services/NotificationTestService

[27]: https://192.168.0.103:8443/wsrf/services/PersistenceTestSubscriptionManager

[28]: https://192.168.0.103:8443/wsrf/services/ReliableFileTransferFactoryService

[29]: https://192.168.0.103:8443/wsrf/services/ReliableFileTransferService

```
[30]: https://192.168.0.103:8443/wsrf/services/RendezvousFactoryService
[31]: https://192.168.0.103:8443/wsrf/services/ReplicationService
[32]: https://192.168.0.103:8443/wsrf/services/SampleAuthzService
[33]: https://192.168.0.103:8443/wsrf/services/SecureCounterService
[34]: https://192.168.0.103:8443/wsrf/services/SecurityTestService
[35]: https://192.168.0.103:8443/wsrf/services/ShutdownService
[36]: https://192.168.0.103:8443/wsrf/services/SubscriptionManagerService
[37]: https://192.168.0.103:8443/wsrf/services/TestAuthzService
[38]: https://192.168.0.103:8443/wsrf/services/TestRPCService
[39]: https://192.168.0.103:8443/wsrf/services/TestService
[40]: https://192.168.0.103:8443/wsrf/services/TestServiceRequest
[41]: https://192.168.0.103:8443/wsrf/services/TestServiceWrongWSDL
[42]: https://192.168.0.103:8443/wsrf/services/TriggerFactoryService
[43]: https://192.168.0.103:8443/wsrf/services/TriggerService
[44]: https://192.168.0.103:8443/wsrf/services/TriggerServiceEntry
[45]: https://192.168.0.103:8443/wsrf/services/Version
[46]: https://192.168.0.103:8443/wsrf/services/WidgetNotificationService
[47]: https://192.168.0.103:8443/wsrf/services/WidgetService
[48]: https://192.168.0.103:8443/wsrf/services/gsi/AuthenticationService
[49]: https://192.168.0.103:8443/wsrf/services/mds/test/execsource/IndexService
[50]: https://192.168.0.103:8443/wsrf/services/mds/test/execsource/IndexServiceEntry
[51]: https://192.168.0.103:8443/wsrf/services/mds/test/subsource/IndexService
[52]: https://192.168.0.103:8443/wsrf/services/mds/test/subsource/IndexServiceEntry
```

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

1.6 Configuration du RFT

1.6.1 Création du fichier pg_hba.conf

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

Nous créons le chemin suivant '/var/lib/pgsql/data'

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

On édite le fichier pg_hba.conf

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

Nous ajoutons la ligne suivante :

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

1.6.2 Création d'un utilisateur globus sous postgres

Nous lançons le serveur postgressql :

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

Le résultat est le suivant:

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

Nous utilisons la base de données test créée précédemment :

```
[postgres@poste4 ~]$ /usr/local/pgsql/bin/psql test
Welcome to psql 8.2.6, the PostgreSQL interactive terminal.
Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit
test=#
```

Nous lançons la création de l'utilisateur globus sous postgres en lançant la requête suivante :

```
test=# CREATE USER globus WITH PASSWORD 'globus' CREATEDB;
```

Résultat :

```
CREATE ROLE
```

1.6.3 Création de la base de données rftDatabase

Après la création de l'utilisateur, nous allons lancer la création de la base de données rftDatabase

```
[globus@poste4 ~]$ /usr/local/pgsql/bin/createdb rftDatabase
```

Résultat :

```
CREATE DATABASE
```

Nous exécutons le script rft_schema.sql qui nous permet de créer les schémas de la base de données

```
[globus@poste4 ~]$  /usr/local/pgsql/bin/psql -d rftDatabase -f
/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql
```

Le résultat sera:

```
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:6: NOTICE:  CREATE TABLE /
PRIMARY KEY will create implicit index "requestid_pkey" for table "requestid"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:11: NOTICE:  CREATE TABLE /
PRIMARY KEY will create implicit index "transferid_pkey" for table "transferid"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:30: NOTICE:  CREATE TABLE /
PRIMARY KEY will create implicit index "request_pkey" for table "request"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:65: NOTICE:  CREATE TABLE /
PRIMARY KEY will create implicit index "transfer_pkey" for table "transfer"
CREATE TABLE
psql:/usr/local/globus-4.0.6/share/globus_wsrf_rft/rft_schema.sql:71: NOTICE:  CREATE TABLE /
PRIMARY KEY will create implicit index "restart_pkey" for table "restart"
CREATE TABLE
CREATE TABLE
CREATE INDEX
```

Nous vérifions les headers du fichier jndi-config.xml, et pour cela nous lançons la commande :

```
[globus@poste4 globus_wsrf_rft]$  grep -C 3 password
$GLOBUS_LOCATION/etc/globus_wsrf_rft/jndi-config.xml
```

Le résultat sera :

```
</parameter>
        <parameter>
          <name>
```

```
password
</name>
<value>
foo
```

1.6.4 Test de fonctionnement du RFT

Avant de tester le bon fonctionnement du rft, nous devons relancer le container des web services avec l'option restart s'il est déjà lancé, sinon le lancer de nouveau avec l'option start.

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

Le résultat sera :

```
Stale pid file detected. It will be removed
Starting Globus container. PID: 3554
```

Nous testons le bon fonctionnement du rft, nous lançons la commande qui fait référence au fichier runtests.xml

```
[globus@poste4 ~]$ ant -Dtests.jar=/usr/local/globus-
4.0.6/lib/globus_wsrf_rft_test.jar -f /usr/local/globus-
4.0.6/share/globus_wsrf_rft_test/runtests.xml
```

Le résultat sera comme suit:

```
Buildfile: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/runtests.xml
init:
  [delete] Deleting directory /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/classes
   [mkdir] Created dir: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/classes
   [unjar] Expanding: /usr/local/globus-4.0.6/lib/globus_wsrf_rft_test.jar into /usr/local/globus-
4.0.6/share/globus_wsrf_rft_test/tests/classes
runServer:
runTests:
_runCustomTests:
   [junit] Running org.globus.transfer.reliable.service.test.PackageTests
   [junit]     GSSException: Expired credentials detected)
   [junit] Tests run: 18, Failures: 0, Errors: 17, Time elapsed: 11,096 sec
   [junit] Test org.globus.transfer.reliable.service.test.PackageTests FAILED
   [junit] Running org.globus.transfer.reliable.service.test.client.PackageTests
   [junit] Tests run: 2, Failures: 2, Errors: 0, Time elapsed: 0,322 sec
   [junit] Test org.globus.transfer.reliable.service.test.client.PackageTests FAILED
   [junit] Running org.globus.transfer.reliable.service.test.connection.PackageTests
```

```
    [junit] Tests run: 81, Failures: 0, Errors: 81, Time elapsed: 1,065 sec
    [junit] Test org.globus.transfer.reliable.service.test.connection.PackageTests FAILED
BUILD SUCCESSFUL
Total time: 15 seconds
```

Nous pouvons exécuter un nouveau test qui fait référence au fichier mentionné précédemment, en lançant la commande suivante :

```
[globus@poste2 ~]$ ant -f /usr/local/globus-
4.0.6/share/globus_wsrf_rft_test/runtests.xml generateTestReport
```

Le résultat sera:

```
Buildfile: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/runtests.xml
generateTestReport:
   [delete] Deleting directory /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/test-reports-html
    [mkdir] Created dir: /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/test-reports-html
[junitreport] Processing /usr/local/globus-4.0.6/share/globus_wsrf_rft_test/tests/test-reports/TESTS-
TestSuites.xml to /tmp/null6835105
[junitreport] Loading stylesheet jar:file:/usr/local/outils/apache-ant-1.7.0/lib/ant-
junit.jar!/org/apache/tools/ant/taskdefs/optional/junit/xsl/junit-frames.xsl
[junitreport] Transform time: 1050ms
[junitreport] Deleting: /tmp/null6835105
BUILD SUCCESSFUL
Total time: 1 second
```

1.7 Configuration du service GRAM

1.7.1 Édition du fichier '/etc/sudoers'

Après avoir installer gridFTP et le RFT, il est maintenant possible de lancer l'installation du gestionnaire des ressources GRAM, pour se faire, nous faisons l'édition du fichier '/etc/sudoers' :

```
[root@poste4 ~]# vim /etc/sudoers
```

Nous ajoutons :

```
globus ALL=(user1,user2) NOPASSWD: /usr/local/globus-4.0.6/libexec/globus-gridmap-and-execute
-g /etc/grid-security/grid-mapfile /usr/local/globus-4.0.6/libexec/globus-job-manager-script.pl *
globus  ALL=(user1,user2) NOPASSWD: /usr/local/globus-4.0.6/libexec/globus-gridmap-and-execute
-g /etc/grid-security/grid-mapfile /usr/local/globus-4.0.6/libexec/globus-gram-local-proxy-tool *
```

1.7.2 Test du fonctionnement de GRAM

Avec la modification apportée précédemment, nous pouvons lancer GRAM et soumettre des jobs, nous lançons la commande suivante :

```
[globus@poste4 ~]$ globusrun-ws -submit -c /bin/true
```

Le résultat sera donc:

```
Submitting job...Done.
Job ID: uuid:b39456d8-526b-11dd-a4f2-001ec92c2a43
Termination time: 07/16/2008 12:44 GMT
Current job state: Active
Current job state: CleanUp
Current job state: Done
Destroying job...Done.
```

1.8 Configuration de gridFTP, RFT et GRAM sous les autres machines

Après l'installation du certificat sur les autres machines (machine1, machine2 et machine3) voir (section 2.4.4) nous procédons à la configuration des autres services (gridFTP, lancement des services Web, RFT, GRAM) de la même façon que pour la machine4 (serveur du CA).

Remarque : Il faut toujours mettre à jour le fichier 'grid-mapfile' dans chaque nœud de la grille en lui ajoutant le sujet et le nom de chaque nouvel utilisateur qui accède à la grille et même les utilisateurs des autres machines qui appartiennent à la même autorité.

2. La mise en œuvre de Sakai sous Linux

Sakai nous offre la possibilité de télécharger une installation précompilée ou source.

2.1 Mise en œuvre précompilée

La version précompilée offre une façon rapide et simple pour mettre sur pieds une instance de Sakai, mais elle assume qu'aucune modification au code ne sera nécessaire. Nous devons néanmoins avoir les droits d'administration sur notre poste et être à l'aise avec la ligne de commande. Les commandes que nous devrons entrer sont inscrites en italique dans les étapes suivantes.

1. Décompresser l'archive tout d'abord sur le disque dur. On assumera ici qu'on décompresse sur /usr/local par défaut.
2. Si on choisit un autre emplacement que /usr/local pour y placer Sakai, on édite les fichiers start-sakai.sh et stop-sakai.sh à la racine du dossier Sakai. Modifier la variable SAKAI_HOME en y inscrivant l'emplacement précis où on a décompressé l'archive.
3. Télécharger et installer MySQL 4.1 en tant que service. L'installation par défaut est suggérée, mais il faut préciser toutefois un mot de passe pour l'utilisateur "root".
4. Lancer un terminal.
5. On assure d'avoir Java 1.5 sur notre poste en entrant *java -version* dans le terminal. Si ce n'est pas le cas, installer la JDK si on prévoit éventuellement développer du nouveau code ou simplement la JRE si on ne prévoit qu'exécuter Sakai.
6. Dans le terminal, taper *mysql -uroot -p* pour accéder au client MySQL. Fournir à ce moment le mot de passe.
7. Entrer dans MySQL la commande *source /usr/local/SakaiVanille/initscript.sql*. Si on a décompressé Sakai Vanille à un endroit différent, spécifier un chemin d'accès approprié. Entrer ensuite *quit* pour quitter MySQL.
8. Toujours en ligne de commande, lancer Sakai au moyen de la commande */usr/local/SakaiVanille/ start- sakai .sh*
9. Sakai est démarré lorsqu'est affiché dans la console le message: *INFO: Server startup in xxxxx ms*
10. Pour accéder à Sakai, utiliser un navigateur internet et aller à l'adresse: *http://localhost:8080/portal*, on pourra créer un compte dans SAKAI ou bien entrer en tant qu'administrateur en utilisant le login: *admin* et le mot de passe: *admin*

2.2 Mise en œuvre précompilée à partir du source

Avec ce type d'installation, on devra récupérer, compiler et déployer nous-mêmes Sakai Vanille. Cela suppose qu'on est à l'aise avec la ligne de commande et qu'on a un minimum d'expérience en développement sous Java. Noter qu'il est possible que ces procédures soient davantage simplifiées dans le futur. On doit Contacter d'abord support@sakaiquebec.org pour demander les codes d'accès à nos différents serveurs. Les commandes qu'on devra entrer sont inscrites en italique dans les étapes suivantes :

1. Créer un dossier nommé /usr/local/SakaiVanille et accéder à ce dossier en ligne de commande. On peut Récupérer le code en y inscrivant :
2. *svn co http://svn.sakaiquebec.org/sakai-quebec/sakaivanille/trunk src*

3. Dézipper ensuite dans /usr/local/SakaiVanille le fichier nommé *vanille_deploy.zip*, situé à la racine du dossier /usr/local/SakaiVanille/src nouvellement créé par SVN.
4. Réaliser les étapes 2 à 7 de l'installation précompilée sous Linux, telles que définies plus haut dans cette page.
5. Télécharger Tomcat 5.5.23 et décompresser-le sur le disque dans /usr/local/SakaiVanille.
6. Télécharger et installer Maven 2. Tel que mentionné dans les procédures d'installation, assurons-nous d'avoir la bonne version en lançant *mvn --version* sur la ligne de commande. Noter ici le double "-".
7. À la manière de la variable qu'on a eu à définir pour Maven, définir une variable d'environnement nommée CATALINA_HOME et pointant sur Tomcat, soit /usr/local/SakaiVanille/apache-tomcat-5.5.23
8. On doit Copier le fichier nommé settings.xml présent dans l'archive *vanille_deploy* dans le dossier .m2 de notre répertoire utilisateur. Sur Linux, ce répertoire est accessible par la variable d'environnement $HOME.
9. Éditer maintenant le fichier settings.xml qu'on a copié pour y modifier les trois paires username/password présentes. Inscrire-y les accès qui vont ont été envoyé par le support de Sakai-Québec.
10. On est maintenant prêts à construire Sakai Vanille. En ligne de commande, accéder au dossier /usr/local/SakaiVanille/src et inscrire *mvn clean install -Pvanille*
11. Une fois la compilation réalisée, on peut déployer notre Sakai vers Tomcat par la commande *mvn sakai:deploy -Dmaven.tomcat.home="$CATALINA_HOME"*
12. Lancer Sakai au moyen de la commande */usr/local/SakaiVanille/ start- sakai .sh*
13. Sakai est démarré lorsqu'est affiché dans la console le message: *INFO: Server startup in xxxxx ms*
14. Pour accéder à Sakai, on doit utiliser notre navigateur internet et aller à l'adresse: *http://localhost:8080/portal*, on pourra créer un compte dans SAKAI ou bien entrer en tant qu'administrateur en utilisant le login: *admin* et le mot de passe: *admin*