Partage de mémoire à très grande échelle sur des réseaux pair-à-pair

Gabriel Antoniu, Luc Bougé, Thierry Priol

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Partage de mémoire à très grande échelle sur des réseaux pair-à-pair

Gabriel Antoniu — Luc Bougé — Thierry Priol

N° 4410

Mars 2002

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--4410--FR

Partage de mémoire à très grande échelle sur des réseaux pair-à-pair

Gabriel Antoniu, Luc Bougé , Thierry Priol

Thème 1 — Réseaux et systèmes
Projet PARIS

Rapport de recherche n° 4410 — Mars 2002 — 13 pages

Résumé : Cet article explore une direction nouvelle dans le domaine de la gestion de données à très grande échelle: le partage de mémoire à l'aide de techniques pair-à-pair (Peer-to-Peer). Les développements actuels dans ce domaine se sont focalisés sur le partage de fichiers. Nous montrons l'intérêt de partager les espaces mémoire des noeuds, et non plus seulement leurs fichiers. Cette idée conduit à repenser complètement la problématique de la mémoire virtuellement partagée dans le contexte du calcul global sur l'Internet ou des grilles de calcul, permettant de rendre la gestion des données aussi transparente que celle des calculs. Nous proposons l'environnement JXTA comme plate-forme de développement de cette approche.

Mots-clés : peer-to-peer, mémoire virtuellement partagée, middleware, JXTA, grille de calcul

Unité de recherche INRIA Rennes

Large-scale Memory Sharing on Peer-to-Peer Networks

Abstract: This paper explores a new research direction in the field of large-scale data management: it addresses memory sharing using peer-to-peer techniques. The current developments in this field have essentially focused on file sharing. We show why it is interesting to share the memory address space of the nodes, not only their files. Most research issues intenssively studied during the last 15 years within the context of distributed shared memory systems for small- and medium-sized clusters of workstations need to be revisited when considering a large-scale distributed execution on the Internet, in the context of grid computing. The idea is to provide a mechanism for transparent data management. We propose to use the JXTA environment as an implementation platform for our approach.

Key-words: peer-to-peer, distributed shared memory, middleware, JXTA, grid computing

1 Introduction

Le fort développement de l'Internet durant les toutes dernières années a mené à l'apparition de nombreuses initiatives ayant pour objectif l'exploitation des ressources de calcul et de stockage disponibles sur ce gigantesque réseau. Quelles que soient leurs formes, ces initiatives ont souvent en commun l'idée que les traitements de calcul intensif réalisés il y a 20 ans sur des supercalculateurs spécialisés, et plus récemment sur des grappes à haute performance, se feront de plus en plus de manière transparente sur des ressources distribuées à une très grande échelle, typiquement à l'échelle mondiale. On peut identifier plusieurs approches de ce type.

Calcul global *Global computing.* Cette approche consiste à récupérer les ressources inutilisées de nœudsde calcul (PC) distribués sur le réseau pour une tâche définie, le plus souvent de manière complètement transparente pour l'utilisateur habituel de ces nœuds. Dès que le nœuddétecte que son utilisateur est inactif, par exemple la nuit, il signale à un serveur central qu'elle est disponible pour effectuer des calculs sous son contrôle.

Calcul sur grille *Grid Computing.* Cette approche consiste à conduire de très gros calculs, typiquement des simulations numériques de plusieurs heures, en utilisant les ressources lourdes de plusieurs centres de calcul distribués sur l'ensemble de la planète. Le problème des procédures administratives imposées par les centres de calcul (inscription, réservation des ressources, facturation, etc.) et de confidentialité des données sont souvent prépondérants dans ce contexte. Contrairement au modèle *client/serveur* du calcul global, il s'agit plutôt ici d'une coopération entre serveurs selon un schéma en général statique, par exemple un pipeline: les données sont acquises en un lieu de la planète, traitées en un autre lieu, et finalement visualisées dans un troisième lieu. Le principal environnement de gestion de ce type de grille est *Globus* (`www.globus.org`). Certains systèmes de calcul distribué interactif comme Scilab // [4] permettent de mettre en œuvredes schémas relativement dynamiques de manière transparente.

Calcul pair-à-pair (*Peer-to-Peer*, P2P). Un autre paradigme de calcul global a récemment focalisé l'intérêt de la communauté scientifique: le calcul *pair-à-pair* [5]. Cette approche généralise les idées du calcul global. Le modèle complémente le modèle classique *client/serveur*, aujourd'hui à la base de la plupart des traitements sur l'Internet, en *symétrisant* la relation des nœudsqui interagissent. La relation client-serveur n'est plus associée aux nœuds, mais aux transactions: chaque nœudpeut être client dans une transaction et serveur dans une autre. Pour l'instant, cette technique a essentiellement été utilisée pour le partage de fichiers entre millions d'utilisateurs, connectés de manière intermittente et imprévisible.

Cet article n'a pas pour but de présenter des travaux déjà réalisés, mais plutôt de discuter en détail d'une nouvelle direction de recherche: l'utilisation des techniques *pair-à-pair* pour partager l'espace mémoire des nœudsd'un réseau de très grande taille, fortement hétérogène, de topologie complètement dynamique, sans point central, où les nœudssont présents de manière intermittente (section 2). L'objectif est que chaque nœudpuisse ainsi accéder de

manière uniforme et transparente à l'ensemble de l'espace d'adressage ainsi constitué. Cette vision renouvelle donc totalement le concept classique de *mémoire virtuellement partagée* (MVP) conçu pour des grappes de petite taille, statiques, homogènes et stables (section 3). Nous proposons ensuite un cadre de développement pour un tel système: *JXTA* (section 4). Il s'agit d'une plate-forme *Open Source* qui propose une infrastructure logicielle *générique* de type intermédiaire (*middleware*) pour les réseaux pair-à-pair. Nous montrons comment cette plate-forme peut être utilisée pour la mise en œuvred'un système de partage de mémoire à très grande échelle.

2 Pourquoi les réseaux *pair-à-pair*?

La plupart des services disponibles sur l'Internet utilisent aujourd'hui le modèle traditionnel *client/serveur*. Dans ce modèle, les clients se connectent à un serveur central à travers un protocole donné (par exemple HTTP, FTP, Telnet, etc.) afin d'accéder à une ressource. Ce modèle ne passe pas à l'échelle: la puissance du serveur et la bande passante disponible sont critiques. De plus, ce modèle exige en général que le serveur reste connecté en permanence sur le réseau, et qu'il soit doté d'une identité fixe (numéro IP, par exemple). Un nœudprésent de manière intermittente et doté d'une adresse IP dynamiquement allouée (DHCP) ne convient pas. Or, ces nœudsreprésentent une puissance de calcul et une capacité de stockage particulièrement importantes, largement sous-utilisées.

C'est l'exploitation de ces ressources qui a motivé l'apparition des réseaux *pair-à-pair* (*peer-to-peer*). Cette approche permet à des nœudsde fournir des services et d'utiliser les services fournis par les autres nœuds, dans une relation de parité. Ces nœudssont en général connectés de manière intermittente, et ils reçoivent leurs identités de manière dynamique. Les traitements ne sont plus centralisés sur un serveur, mais répartis sur l'ensemble des pairs. La réplication des services sur les nœudspermet d'assurer leur disponibilité continue malgré l'instabilité des connexions et des désignations.

Le premier exemple de logiciel qui illustre cette approche a été le réseau *Napster*, mis en place pour le partage de fichiers MP3: chaque nœudmet a disposition sur le réseau des documents et en même temps a accès à des millions de documents disponibles sur les autres nœudsdu réseau. Un exemple récent est le réseau *KaZaA* (`www.kazaa.com`) qui peut regrouper simultanément 500.000 nœudsdonnant accès à 75 millions de fichiers contenant 400 téraoctets de données.

Indépendamment des aspects légaux, ce type d'approche offre une véritable solution dans le contexte du partage d'une grande quantité de documents et présente d'importants avantages si on la compare à des solutions classiques basées sur une approche client-serveur. Ceci a motivé leur utilisation pour les systèmes d'information de grandes compagnies multi-sites, telles qu'IBM et Intel.

Extensibilité à faible coût. En plus du goulot d'étranglement qui rend inefficace l'utilisation d'un serveur central accédé par un très grand nombre de nœudsclients, le coût du stockage centralisé devient significatif lorsqu'il s'agit de plusieurs téraoctets de données. L'approche pair-à-pair est naturellement appropriée au passage à l'échelle à la

fois en terme de puissance de traitement, de capacité de stockage et de ressources réseau. Elle permet d'utiliser notamment la bande passante disponible à la périphérie du réseau, souvent sous-utilisée.

Haute disponibilité. Dans une architecture traditionnelle, la défaillance d'un serveur centralisé rend indisponibles tous les services pour l'ensemble des nœudsclients. Dans un réseau pair-à-pair, la défaillance d'un nœudne met généralement pas en cause le bon fonctionnement du réseau, grâce à une distribution largement redondante des ressources sur les différentes nœuds.

Fiabilité des répertoires de ressources Pour les opérations de recherche de ressources, les moteurs de recherche traditionnels utilisent des bases de données centralisées mises à jour périodiquement. À cause du très grand volume d'information à référencer, le maintien à jour de ces index centraux est difficile. Par conséquent, ils peuvent souvent fournir des informations obsolètes. En revanche, dans un contexte pair-à-pair, la responsabilité de l'indexation peut être répartie, chaque nœudpouvant maintenir un index à jour sur ses propres ressources. S'agissant d'une quantité très limitée de ressources pour chaque nœud, le maintien à jour des index est dans ce cas beaucoup plus facile. La réponse à une requête peut alors être satisfaite par la mise en commun de ces méta-données.

Les réseaux pair-à-pair présentent aussi des désavantages. De par la nature du réseau, l'exécution des services sur ces réseaux n'est pas nécessairement déterministe. Par exemple, la réponse à une même requête dépend de la localisation du demandeur sur le réseau et des ressources disponibles dans son voisinage. Par ailleurs, une ressource peut disparaître du réseau si le nœudqui la fournit se déconnecte du réseau. Néanmoins, du fait de la réplication implicite des ressources les plus demandées par le protocole, ce problème de disponibilité ne se pose que dans le cas des ressources rarement utilisées. Nous remarquerons que plus la taille de ces réseaux augmente, plus haute est la probabilité de trouver une ressource.

Jusqu'à présent, l'application la plus typique ayant tiré profit des réseaux pair-à-pair a été le *partage de documents* en mode lecture, typiquement des fichiers [6, 5]. *Napster* était basé sur une approche hybride: le stockage et les transferts de fichiers étaient distribués, mais les répertoires étaient centralisés. *Gnutella*, l'un de ses successeurs, implémente un protocole complètement distribué. Un autre successeur est *Freenet*, qui se focalise sur l'anonymat des sources et sur l'intégrité des documents. Aujourd'hui, des systèmes de partage de fichiers de deuxième génération, telles que *KaZaA*, offrent des fonctionnalités plus complexes: transferts parallèles de plusieurs sources, possibilités d'arrêt et de reprise d'un transfert, etc. Une autre application typique pour les réseaux pair-à-pair est la *messagerie instantanée* proposée par *ICQ* ou par *Yahoo!Messenger*. Enfin, des projets sont actuellement en cours pour utiliser les réseaux pair-à-pair comme support pour le calcul distribué, par exemple *XtremWeb* [3].

3 Partage de mémoire en contexte pair-à-pair

Un grand nombre de travaux ont été consacrés au problème du partage de fichiers en lecture sur des réseaux pair-à-pair. Par contre, aucun, à notre connaissance, ne s'est intéressé à la possibilité d'utiliser cette approche pour le partage d'espace mémoire en lecture *et* en écriture, généralisant ainsi à grande échelle les techniques de *mémoire virtuellement partagée* (MVP) bien connues pour les grappes. La contribution de cet article est d'analyser les problèmes posés par la conception d'un tel système.

Un tel service de partage de mémoire s'apparente aux systèmes de *mémoire virtuellement partagées* (MVP). Il est responsable de la gestion de l'allocation, de la localisation, de la réplication et, en plus, de la distribution des données. Ces problématiques, largement étudiées pour le cas des grappes de stations, doivent être ré-évaluées lors du passage à l'échelle. Les réseaux pair-à-pair sont bien adaptés à ce nouveau cadre, mais le problème est de passer d'une vision du réseau comme espace de *stockage* distribué à une vision du réseau comme espace d'*adressage global* distribué. Il faut intégrer des mécanismes de gestion de la *cohérence* des données répliquées, qui peuvent être modifiées par un sous-ensemble des nœudset lues par un autre sous-ensemble. Ce problème ne se posait pas dans le cas du partage de fichiers, qui correspond à un schéma de partage de type réplication très simple, en lecture seule.

Utiliser un réseau pair-à-pair comme support pour le partage de mémoire nécessite de revisiter les hypothèses des MVP classiques et d'en repenser les mécanismes afin de prendre en compte les caractéristiques de ce contexte original.

Dynamicité du réseau sous-jacent. Les réseaux pair-à-pair supportent des milliers, voire des millions de nœudsqui peuvent se connecter et se déconnecter du réseau de manière dynamique pendant l'exécution d'un traitement. Les études sur les systèmes à MVP tolérants aux pannes sont donc particulièrement intéressantes dans ce cadre. Cependant, la problématique est plus générale ici, car c'est en fait le *degré* de disponibilité des pairs qui varie: l'espace mémoire offert au partage, la bande passante disponible, etc. Les déconnexions peuvent être vues comme des cas particuliers où la disponibilité d'un nœuddevient nulle.

Hétérogénéité. Les systèmes à MVP ont été conçus le plus souvent pour des grappes homogènes. Dans un réseau de très grande taille, l'hypothèse d'homogénéité des ressources matérielles n'est plus réaliste. Le cas le plus fréquent est celui où les processeurs, les systèmes d'exploitation, les réseaux et les protocoles de communication varient d'une zone du réseau à une autre. Les caractéristiques des différentes ressources mises en jeu (débit réseau, fiabilité, niveau de sécurité, degré de disponibilité, etc.) sont également hétérogènes, dans l'espace et dans le temps. La portabilité et l'intéropérabilité deviennent dans ce contexte des propriétés particulièrement importantes.

Architecture hiérarchique. À la différence des grappes, caractérisées le plus souvent par une architecture *plate*, le passage à l'échelle impose une architecture *hiérarchique*, par exemple des constellations de grappes. Les études récentes sur l'extension des modèles de cohérence classiques à de telles architectures sont particulièrement intéressantes dans ce cadre [2].

Hypothèse	MVP classiques	Mémoire partagée sur réseaux P2P
Nombre de nœuds	fixe, 10–100	variable, 10^3–10^5
Disponibilité des nœuds	fixe	variable dans le temps
Configuration matérielle	homogène, statique	hétérogène, dynamique
Modèle de mémoire	plat	hiérarchique
Partage	intensif, global	faible, local
Critères d'évaluation	efficacité	disponibilité

TAB. 1 – Comparaison d'hypothèses: MVP classiques vs systèmes pair-à-pair

Espace d'adressage. Les systèmes à MVP classiques ont été conçus pour des grappes de petite taille. Un espace d'adressage global codé sur 32 bits était suffisant. Si l'on veut maintenant pouvoir accéder de manière uniforme à la mémoire de centaines de milliers de nœuds, 64 bits d'adressage sont nécessaires. Il semble raisonnable de penser que les processeurs courants fourniront cette fonctionnalité d'ici quelques années, mais ce point ne doit pas être négligé.

Nommage des nœuds. Les systèmes à MVP classiques se placent dans un cadre statique, où les nœudspeuvent être désignés par des entiers consécutifs. Ceci permet une gestion très optimisées des tables. En présence de centaines de milliers de nœudsdispersés sur l'Internet, présents de manière intermittente, sans répertoire centralisé, une telle approche est clairement inadéquate. Il faut pouvoir s'appuyer sur un système auxiliaire de gestion dynamique de noms uniques. Dans l'approche proposée ici, ce service est fourni par JXTA.

Schémas de partage. Les systèmes à MVP pour les grappes supposent le plus souvent qu'une donnée peut être accédée indifféremment par tous les nœudsà chaque instant. Dans un réseau pair-à-pair de grande taille, il semble raisonnable de poser une hypothèse de *localité* sur le partage: une donnée est traitée par un pair ou un groupe de pairs (une grappe, par exemple), puis par un autre groupe, etc. Par ailleurs, les schémas de partage ont également une structure hiérarchique. Par exemple, plusieurs applications qui coopèrent peuvent utiliser un ensemble de matrices stockées dans l'espace de mémoire partagée en y accédant suivant un schéma de type pipe-line. À un niveau hiérarchique inférieur, à chaque étape, un ensemble de sous-matrices sont utilisées par une application distribuée donnée suivant un certain schéma d'accès local à cette étape.

Critères d'évaluation. Le passage à l'échelle a aussi pour effet une remise en cause des priorités des critères d'évaluation des systèmes. La fonctionnalité, la transparence, la tolérance du support dynamique et l'intéropérabilité deviennent plus importantes que la performance, qui passe souvent au second plan.

4 JXTA: une infrastructure pour des services pair-à-pair

Une analyse des systèmes pair-à-pair existants permet de constater que ces réseaux partagent un grand nombre de fonctionnalités communes. Malheureusement, ces fonctionnalités sont souvent volontairement implémentées par des mécanismes propriétaires, fermés. Afin de mettre en œuvreun nouveau type de service pair-à-pair, il est souvent nécessaire de construire un système complet, alors qu'un grand nombre de composants du système reposent sur des mécanismes déjà implémentés précédemment. Par conséquent, un investissement important en temps et ressources matérielles et humaines est nécessaire lors de la validation de nouvelles techniques pair-à-pair, car cela nécessite à chaque fois la mise en œuvredes mécanismes de base.

En partant de ce constat, le projet JXTA [7] s'est proposé d'identifier plusieurs concepts et mécanismes de base *génériques* propres aux réseaux pair-à-pair. Initié par Sun dans une approche *Open Source*, ce projet propose plusieurs protocoles permettant de gérer un certain nombre de fonctionnalités communes, comme la gestion des pairs, la découverte de ressources, l'invocation de services sur le réseau, la communication inter-pair, la sécurité, etc. Ces protocoles peuvent servir de briques de base pour la mise en œuvrede services pair-à-pair. JXTA met ainsi les fondations d'une *infrastructure générique* de type *middleware* pour des applications pair-à-pair.

4.1 Concepts de base

Pair. Un *pair* est l'unité structurelle de base de tout réseau pair-à-pair. Un pair peut être toute ressource capable de ce connecter au réseau et de fournir un service: une station de travail, une grappe de stations, ou même un assistant personnel électronique, de type PDA. JXTA distingue plusieurs types de pairs.

Pairs minimaux. Ces pairs sont réduits à des fonctionnalités élémentaires de communication: envoi et réception de message. Il ne sont capables ni de stocker de l'information, ni de router les messages.

Pairs simples. Ces pairs peuvent envoyer et recevoir de messages et, en plus, stocker dans un cache local des informations sur les ressources du réseau, sous forme d'*annonces* (cf. définition d'annonce plus loin). Ces annonces leur permettent de répondre à des requêtes d'informations et de participer ainsi au protocole de découverte de ressources.

Pairs de rendez-vous. Ces pairs sont essentiels à la constitution du réseau. En plus des fonctionnalités décrites plus haut, ces pairs permettent de faire suivre les requêtes vers d'autres pairs connus (pairs simples ou pairs de rendez-vous).

Pairs de routage. Ces pairs stockent des informations de routage vers les autres pairs. En plus, ils permettent de transmettre les requêtes des pairs isolés du réseau par des mécanismes de pare-feu.

Groupe de pairs. Les *groupes de pairs* correspondent à des ensembles de pairs réunis par un intérêt commun: une application de collaboration, un ensemble de services ou un niveau de sécurité.

Service. Les *services* sont des traitements fournis au sein d'un réseau pair-à-pair (recherche, partage, etc.). Ce sont eux qui motivent la constitution du réseau. On distingue deux types de services: des services *individuels*, fournis par des pairs individuels et des services *de groupe*, fournis par un groupe à tous ses membres. Plusieurs membres du groupe peuvent contribuer, de manière redondante, à ce dernier type de service, qui reste disponible tant qu'au moins un membre du groupe reste connecté.

Annonce. Une annonce est une représentation structurée d'une entité présente sur le réseau. En particulier, toutes les entités décrites jusqu'ici (pair, groupe de pairs, service) ou toute autre ressource peut être décrite par une annonce.

Protocoles de base. Plusieurs *protocoles* définissent la structure des échanges d'informations pour un ensemble d'interactions fréquentes dans un réseau pair-à-pair. JXTA définit 6 protocoles, que nous décrivons dans la section suivante.

4.2 Protocoles JXTA

Les protocoles JXTA sont définis sous forme de séries de messages XML et correspondent aux principaux types d'interaction présents dans un réseau pair-à-pair. Tous les protocoles sont asynchrones et reposent sur un modèle requête/réponse. Typiquement, un pair envoie une requête à un ou plusieurs pairs de son groupe et peut recevoir zéro, une, ou plusieurs réponses. Il n'est pas requis que tous les pairs implémentent tous les protocoles.

Protocole de découverte. Ce protocole (PDP: *Peer Discovery Protocol*) est utilisé par les pairs pour faire connaître leurs propres ressources et pour découvrir les ressources fournies par les autres pairs. Le protocole utilise les pairs de rendez-vous pour transmettre une requêtes vers tous les pairs connus. Parmi ces pairs, les pairs simples peuvent éventuellement répondre aux requêtes, alors que les pairs de rendez-vous permettent, en plus, de faire suivre la requête. Dans l'implémentation de référence de JXTA, ce protocole repose sur des communications de type multicast sur des réseaux locaux, combinés avec des retransmissions effectuées par les *pairs de rendez-vous*.

Protocole d'information. Ce protocole (PIP: *Peer Information Protocol*) permet d'obtenir des informations sur les pairs (état, traffic, etc.) identifiés à l'aide du protocole précédent. Ces informations peuvent servir pour guider le déploiement interne des applications JXTA.

Protocole d'invocation de service. Ce protocole (PRP: *Peer Resolver Protocol*) permet de transmettre des requêtes génériques sur le réseau. Les requêtes peuvent être adressées à un pair spécifique ou bien peuvent être propagées à tous les membres d'un groupe via les *pairs de rendez-vous*. Les deux protocoles précédents spécialisent ce protocole qui permet l'échange de tout type d'information: dans le cas de PDP le contenu est constitué par des annonces, alors que PIP l'utilise pour communiquer des informations d'état.

Protocole de communication par canaux. Le protocole PBP (*Pipe Binding Protocol*) permet d'établir des canaux virtuels de communication (*pipes*) entre 2 ou plusieurs pairs. Un canal est une file abstraite de messages qui supporte des opérations de création, destruction, connexion à un ou plusieurs pairs, déconnexion, envoi et réception de messages. Le message peut s'adresser à un pair spécifique ou à l'ensemble des pairs connectés au canal.

Protocole de routage. Ce protocole (ERP: *Endpoint Routing Protocol*) de plus bas niveau est utilisé par les pairs pour déterminer la route d'accès à une destination donnée, sous forme de séquence de pairs de routage. Un pair qui doit envoyer un message recherche d'abord ces informations de routage dans son cache local. Si l'information ne s'y trouve pas ou si le pair n'a pas de cache de routage, une requête est envoyée à un *pair de routage*, qui répond ou fait suivre la requête.

Protocole de rendez-vous. Le protocole RVP (*Rendezvous Protocol*) sert à propager des messages au sein d'un *groupe de pairs*. Il permet aux pairs de se *connecter* au service (et donc d'être capables de communiquer sur le réseau) et contrôle la propagation des messages (gestion de la durée de vie du message et détection des boucles). Les protocoles d'invocation de services (PRP) et de communication par canaux (PBP) reposent sur ce protocole.

L'implémentation de référence de JXTA supporte l'ensemble de ces protocoles. Néanmoins, un pair peut redéfinir ses propres protocoles si les comportements par défaut ne sont pas jugés satisfaisants. Ces six protocoles offrent une couche de base de type *middleware* pour l'implémentation de services pair-à-pair spécialisés qui peuvent être intégrés dans des applications pair-à-pair.

5 Définition d'un service de partage de mémoire dans JXTA

Notre objectif est de définir un service de partage de mémoire pair-à-pair. L'idée est de construire un espace d'adressage virtuel global au-dessus d'un ensemble de zones mémoires mises à disposition par les pairs. Comme nous l'avons montré, la conception d'un tel service introduit les problématiques des systèmes à mémoire virtuellement partagée dans le domaine des réseaux pair-à-pair et nécessite la prise en compte de nouvelles hypothèses, que nous avons discutées dans la section 3: dynamicité du réseau, hétérogénéité, etc. Or, JXTA prend déjà en compte une partie significative de ces nouveaux aspects et fournit des éléments génériques qui peuvent servir de couche de base. C'est au-dessus de ces éléments génériques que les mécanismes d'un système de partage de mémoire virtuelle peuvent être construites.

L'architecture que nous proposons est illustrée sur la figure 1. Le système de partage de mémoire est défini comme un service JXTA, situé entre l'application distribuée et le noyau JXTA, sur lequel il s'appuie. Ce noyau prend en compte des traitements génériques nécessaires dans tout réseau pair-à-pair. En particulier, le service de partage de mémoire utilise les mécanismes de gestion des groupes, de monitoring et de gestion des canaux virtuels

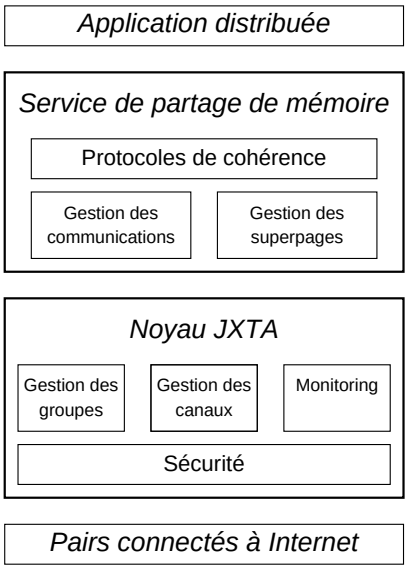

FIG. 1 – Architecture générale.

de communication. Le noyau JXTA intègre également des mécanismes permettant la mise en œuvrede politiques de sécurité: cryptage des données, authentification par certificats.

De manière interne, le service de partage est structuré comme un système à mémoire virtuellement partagée classique. Nous considérons qu'un tel système doit respecter une structure modulaire et doit supporter l'implémentation de différents modèles et protocoles de cohérence, selon une structure similaire à celle implémentée dans le système DSM-PM2 [1]. Au niveau haut, le service comporte un module de *gestion des protocoles de cohérence*. Ces protocoles s'appuieront sur les fonctionnalités offertes par deux modules spécialisés: un module de *gestion des communications* et un module de *gestion des superpages*.

Gestion des communications Ce module gère les opérations de communication typiques des systèmes à mémoire virtuellement partagée tout en les adaptant à JXTA: requêtes de superpages, transfert des superpages, requêtes d'invalidation, etc. Dans le cadre des MVP pour grappes telles que DSM-PM2, ces opérations peuvent être implémentées à l'aide de mécanismes d'appel de procédure à distance (RPC). De manière analogue, sur un réseau pair-à-pair, ces opérations peuvent être implémentées en utilisant le *protocole d'invocation de services* de JXTA (PRP), qui fournit un mécanisme similaire en s'appuyant à son tour sur le *protocole de rendez-vous* (RVP). Le *protocole de routage* est également sollicité lors du transfert des messages. Ce protocole permet notamment de communiquer avec les pairs isolés du réseau par un pare-feu.

Gestion des superpages Une *superpage* est l'unité qui définit la granularité du partage, à la manière des pages dans les MVP classiques. Une *superpage* peut correspondre à une ou plusieurs pages. Le rôle de ce module correspond au rôle d'un gestionnaire des pages dans une MVP classique: localisation des données, gestion des droits d'accès sur les différents nœuds(pairs). À la différence des gestionnaires classiques, ce module doit gérer une configuration dynamique des nœuds, car les pairs peuvent se connecter ou se déconnecter avec intermittence. Ces événements sont captés par le *gestionnaire des groupes* du noyau JXTA, qui les signale au *gestionnaire des superpages*, lui permettant d'exécuter les actions nécessaires pour préserver la sécurité et la cohérence des données. Nous étudions actuellement une approche basée sur l'utilisation d'une correspondance entre un ensemble de nœudslogiques toujours disponibles et un ensemble variable de nœudsphysiques choisis parmi les pairs connectés, membres du groupe. Le choix des pairs est guidé par des informations fournies par le module de *monitoring* du noyau JXTA.

Connexion et déconnexion des pairs La participation des pairs au service se fait de la manière suivante. Les pairs-membres du groupe se déclarent automatiquement au service lors de la connexion et instancient le service de partage. Ils sont à la fois fournisseurs et utilisateurs de la mémoire. Lors de cet enregistrement, il déclarent une zone de mémoire utilisable par JXTA. L'ensemble de ces zones de mémoire est géré par le *gestionnaire des superpages*. Lorsqu'un pair se déconnecte, ce gestionnaire doit mettre à jour ses répertoires afin de supprimer les correspondances vers la mémoire du pair. Si des données non répliquées sont stockées uniquement dans la mémoire de ce nœud, ces données sont migrées vers d'autres pairs.

6 Conclusion

Cet article explore une direction nouvelle dans le domaine de la gestion de données partagées à très grande échelle. Nous avons montré que ce problème n'a pas été considéré jusqu'à présent par les environnements de calcul à grande échelle comme Globus, ou que des solutions *ad-hoc* peu satisfaisantes ont été proposées comme dans Scilab //.

Nous proposons une approche générale de *partage de mémoire* entre les nœudsfondée sur le principe de parité. Les données sont ainsi accédées par un adressage global de manière transparente par rapport à leur localisation, comme dans les systèmes à MVP classiques, conçus pour des grappes. Mais le contexte est ici complètement différent: le réseau est dynamique, les nœudssont présents de manière intermittente et imprévisible, fortement hétérogène, hiérarchique, avec des schémas de partage localisés. Alors que l'efficacité était l'objectif majeur des MVP, la haute disponibilité semble ici devenir prépondérante.

Nous proposons de développer une telle infrastructure sur l'environnement de gestion de réseau pair-à-pair JXTA. Nous avons montré que les fonctionnalités de JXTA permettent d'implémenter les services de partage de mémoire et d'adressage global à très grande échelle, en permettant de contrôler la disponibilité. Bien sûr, la performance d'une telle plate-forme reste à évaluer. Nous travaillons actuellement à la conception d'un premier prototype.

Références

[1] Antoniu (Gabriel). – *DSM-PM2: une plate-forme portable pour l'implémentation de protocoles de cohérence multithread pour systèmes à mémoire virtuellement partagée.* – Thèse de doctorat, LIP, ENS Lyon, Novembre 2001.

[2] Arantes (Luciana). – *A Distributed Shared Memory Runtime Support for Loosely-coupled Clusterd-based Platforms.* – Thèse de doctorat, LIP6, Univ. Paris 6, Décembre 2000.

[3] Cappello (Franck). – *Modèles d'exécution parallèles pour les architectures hautes performances et les grands systèmes distribués.* – Habilitation à diriger les recherches, LRI, Univ. Paris Sud, Octobre 2001.

[4] Desprez (Frédéric). – *Calcul numérique: des bibliothèques aux environnements de metacomputing.* – Habilitation à diriger les recherches, LIP, Univ. Claude Bernard, Lyon, Juillet 2001.

[5] Oram (Andy) (édité par). – *Peer-to-Peer, harnessing the power of disruptive technologies.* – O'Reilly & Associates, Mars 2001.

[6] Shirky (Clay). – What is P2P... and what isn't, Novembre 2000. Disponible à l'URL `www.openp2p.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html`.

[7] Sun Microsystems, Inc. – *Project JXTA: Java programmer's guide*, 2001. Disponible à l'URL `www.jxta.org/project/www/jxtaprogguide_final.pdf`.

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399