INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet PARIS

Programmation des systèmes parallèles et distribués pour la simulation numérique à grande échelle

Rennes

THÈME 1A

Table des matières

1. Composition de l'équipe

Le projet PARIS *a été reconnu projet commun de l'*IRISA *et de l'Antenne Bretagne de l'ENS Cachan par une convention signée en novembre 2001. L'activité du projet est examinée par les partenaires lors d'un Comité de suivi annuel. Pour 2002, ce comité s'est réuni le 22 mars à l'IRISA.*

Responsable scientifique
Thierry Priol [DR INRIA]

Assistante
Huguette Béchu [TR INRIA, jusqu'au 30 novembre 2002]
Maryse Auffray [TR INRIA, depuis le 1er décembre 2002]

Personnel Inria
Gabriel Antoniu [CR depuis le 1er septembre 2002]
Yvon Jégou [CR]
Christine Morin [DR INRIA depuis le 1er octobre 2002]
Christian Pérez [CR]

Personnel Université de Rennes 1
Jean-Pierre Banâtre [professeur, Université de Rennes 1, depuis le 1er octobre 2002]
Christine Morin [maître de conférence (en détachement de l'INRIA jusqu'au 30 septembre 2002), Université de Rennes 1]

Personnel Insa de Rennes
Jean-Louis Pazat [maître de conférence, INSA de Rennes]

Personnel ENS Cachan
Luc Bougé [professeur, Antenne de Ker Lann]

Ingénieurs experts Inria
Benoît Hubert [depuis le 1er septembre 2002, contrat ACI GRID *RMI*]
Guillaume Mornet [depuis le 1er octobre 2002, contrat RNTL *CasPer*]

Ingénieur associé Inria
Viet Hoa Dinh [jusqu'au 6 septembre 2002]

Chercheur post-doctorant
Renaud Lottiaux [bourse PDI INRIA co-financé par EDF]

Chercheurs doctorants
Alexandre Denis [Allocataire moniteur normalien]
Pascal Gallard [bourse INRIA]
Sébastien Lacour [bourse INRIA à partir du 1er octobre]
André Ribes [bourse INRIA-Région]
Louis Rilling [normalien, à partir du 1er octobre]
Gaël Utard [bourse INRIA]
Geoffroy Vallée [bourse Cifre EDF]

Autre personnel
Ramamurthy Badrinath [Chercheur invité de l'IIT Kharagpur, depuis le 10 mai 2002]

2. Présentation et objectifs généraux

2.1. Introduction

Le projet étudie la programmation des grappes de calculateurs pour des applications utilisant des techniques de simulation numérique distribuée. Il a pour ambition de construire des mécanismes systèmes et des

environnements logiciels en vue de faciliter la mise en œuvre de telles applications. Il s'agit notamment d'étudier les mécanismes logiciels permettant de faciliter la conception et l'expérimentation d'applications ayant pour cible des architectures qui sont, par nature, à la fois parallèles et distribuées. Les recherches menées par le projet sont organisées selon deux grands thèmes : *la programmation de grappes de calculateurs homogènes* et *la programmation de grappes hétérogènes de calculateurs*. Pour atteindre ces objectifs, nos travaux s'appuient sur une plate-forme, en cours de montage à l'IRISA, constituée d'un ensemble conséquent de grappes de calculateurs. Le projet a également pour ambition de « valider » le résultat de ses recherches en prenant en compte des applications par une participation active à des actions de transfert technologique.

2.1.1. Le parallélisme pour calculer plus vite

L'émergence de nouvelles technologies dans le domaine des architectures de processeurs, de calculateurs et des réseaux permet d'entrevoir de nouvelles applications fondées sur une utilisation intensive de la *simulation numérique*. Le parallélisme, qu'il soit à grain fin au sein d'un processeur ou bien à gros grain au sein d'un calculateur parallèle, a permis de réduire considérablement les temps de calcul des applications qui utilisent des techniques de simulation numérique. Un simple PC muni de quelques processeurs permet désormais de réaliser une simulation complexe en quelques heures dans des domaines très variés tels que la déformation de structure, l'écoulement des fluides, la propagation des ondes, la compatibilité électromagnétique ou bien encore dans le domaine de la finance. Ces temps de calcul continueront de décroître avec l'arrivée prochaine d'une nouvelle génération de processeurs pour les PC. La simulation numérique n'est donc plus synonyme de supercalculateurs coûteux réservés uniquement aux grandes industries (aéronautique, automobile, nucléaire, etc.).

Cette évolution va rendre les techniques de simulation numérique accessibles à un plus grand nombre d'acteurs économiques, dont notamment des PME/PMI. Les contraintes, fixées par un marché de plus en plus mondialisé, vont imposer aux acteurs économiques, quelle que soit leur taille, de réduire de façon très significative les délais et les coûts de conception. La simulation numérique sera un outil incontournable et prendra un réel essor dans les années futures.

L'adoption de cette technologie posera des problèmes nouveaux et qui ne pourront pas être résolus par le *parallélisme*. En effet, l'utilisation croissante de la simulation va faire naître le besoin non plus de simuler un seul aspect d'un problème mais un plus grand nombre de phénomènes physiques. Jusqu'à maintenant, et malgré l'évolution rapide de la technologie des processeurs, la performance des machines ne permet pas de simuler complètement le comportement d'un système physique car cela met en jeu un grand nombre de modèles mathématiques dont la résolution est trop coûteuse en temps de calcul. Par conséquent, la simulation concerne le plus souvent un seul aspect physique d'un problème (la déformation de structure ou bien l'écoulement d'un fluide).

2.1.2. La distribution pour résoudre de nouveaux problèmes

L'utilisation d'une grappe de calculateurs interconnectés via un réseau à très haut débit offre désormais un niveau de performance *simulation numérique distribuée*. Il s'agit non plus d'utiliser un seul code mais un ensemble de codes, qui collaborent entre eux, permettant ainsi d'améliorer la qualité de la simulation en prenant en compte un plus grand nombre de phénomènes physiques. La contrainte de performance n'est pas la seule qui rend nécessaire la simulation numérique distribuée. Dans un système économique mondialisé, la réalisation de grand projets industriels nécessite de mettre en commun un ensemble d'expertises qui sont apportées par différentes sociétés, chacune ayant ses propres outils de simulation numérique.

La simulation numérique de plusieurs phénomènes physiques nécessite l'utilisation d'environnements logiciels qui permettent d'intégrer et de coupler plusieurs codes de simulation. Cette intégration nécessite à la fois des techniques relevant du parallélisme et du distribué. Le *parallélisme* permet de répondre aux contraintes de performance alors que le *distribué* est imposé pour satisfaire les exigences en ressources et prendre en compte la localisation géographique des équipements ou des expertises.

Par exemple, l'exécution simultanée de plusieurs codes de simulation sur une même machine peut être rendue inefficace, voire impossible, par le manque de ressources mémoire. La distribution s'impose donc

afin de pouvoir exploiter un plus grand nombre de ressources disponibles sur un réseau. Deux codes peuvent s'exécuter plus rapidement sur deux machines même s'il faut échanger des données car les mécanismes de pagination sont souvent plus coûteux (accès disque) que des communications sur les réseaux à très haut débit actuellement disponibles.

La nécessité de distribution peut être imposée par l'application elle-même. Deux sociétés qui participent à la conception d'un produit (un avion par exemple) peuvent avoir à simuler des parties différentes du produit (une antenne et le fuselage). Aucune des sociétés n'étant disposée à communiquer son savoir faire (modélisation des objets par exemple), celles-ci souhaitent effectuer la simulation en utilisant leurs propres ressources de calcul connectées par l'Internet. La visualisation des résultats d'une simulation peut également imposer la distribution des codes de simulation en fonction de la localisation des ressources graphiques (centre de réalité virtuelle par exemple).

2.1.3. Les problèmes abordés par le projet

La conception d'une application, fondée sur des techniques de simulation numérique distribuée, pose de nombreux problèmes à la fois *du point de vue de la compatibilité des méthodes numériques couplage de codes*) et du point de vue des concepts informatiques nécessaires pour en faciliter le déploiement. Le projet PARIS contribue à ce deuxième aspect en étudiant les mécanismes logiciels nécessaires à la simulation numérique distribuée sur des grappes de calculateurs. Le problème est difficile à la fois par la complexité de l'architecture matérielle qui est la cible du projet PARIS et par la nature des applications traitées. Une application de simulation numérique distribuée peut être vue comme un ensemble de codes de simulation indépendants qui peuvent être de nature séquentielle ou parallèle. L'objectif des travaux du projet PARIS est de proposer des services système et des environnements logiciels qui permettent d'exécuter ce type d'applications, sous forme de composants logiciels, à la fois sous des contraintes de performance et de facilité d'utilisation. La figure 1 résume l'approche suivie par le projet PARIS. Une application de simulation numérique distribuée peut alors être vue comme un ensemble de composants logiciels interconnectés par un bus logiciel. Un composant est implanté soit sous forme d'un seul processus (composant séquentiel) ou de plusieurs processus (composant parallèle). Il s'agit de permettre l'exécution de ces composants sur une grappe de calculateurs telle que celle présentée dans la figure 1. Une grappe de calculateurs homogènes peut être vue comme un seul calculateur permettant l'exécution d'un composant logiciel parallèle.

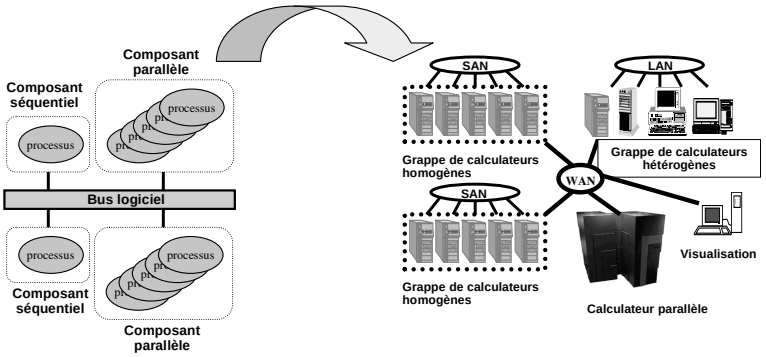

Figure 1. Programmation d'une grappe de calculateurs à l'aide de composants logiciels.

La programmation d'une grappe de calculateurs est rendue complexe par la distribution physique des ressources (processeurs, mémoires, disques). Chaque machine dispose de son propre système d'exploitation en charge de gérer ses ressources matérielles. Notre objectif est d'offrir à l'utilisateur une vision unique de l'ensemble de la grappe de calculateurs. Il s'agit notamment de masquer la distribution des ressources (processeurs, mémoires, disques, réseau) d'une grappe de calculateurs à la fois parallèle (plusieurs processeurs qui partagent une mémoire) et distribués (un ensemble de machines qui disposent chacune de leur propre mémoire). La distribution des ressources n'est pas le seul problème à prendre en compte. Le caractère

homogène ou hétérogène (au sens des processeurs, des réseaux et des systèmes d'exploitation) d'une grappe de calculateurs implique des approches différentes pour atteindre les objectifs que se fixe le projet PARIS.

Dans le cas des grappes homogènes, où les calculateurs de la grappe sont identiques, nous adoptons une approche système en concevant des services de gestion unifiée des ressources processeurs, mémoires et disques de la grappe ainsi que des outils pour les exploiter. Ces services sont destinés à supporter l'exécution de composants logiciels parallèles. Cette approche suppose l'utilisation d'un système d'exploitation dont les sources sont librement accessibles, en l'occurrence Linux. Pour la programmation d'une grappe constituée d'éléments homogènes et hétérogènes, comme celle présentée dans la figure 1, une autre approche est nécessaire car la grappe est constituée de calculateurs différents, chacun ayant son propre système d'exploitation, éventuellement fonctionnant sur des architectures de processeurs différents. Dans ce cas, nous adoptons une approche de type *middleware* [1] pour concevoir des services de gestion de ressources. Il s'agit donc de concevoir des environnements logiciels distribués ou des machines virtuelles qui s'appuieront sur les services offerts par les différents systèmes d'exploitation des calculateurs et sur ceux que nous aurons conçus pour les grappes homogènes. Ces deux approches constituent les deux axes de recherche du projet PARIS.

Bien que le projet PARIS s'intéresse essentiellement aux applications de simulation numérique, nous n'excluons pas d'appliquer les résultats de nos recherches à d'autres domaines applicatifs (traitement d'images, synthèse d'images, traitement des données) en collaboration étroite avec d'autres d'équipes de recherche spécialistes de ces domaines.

2.2. Programmation d'une grappe homogène de calculateurs

Mots clés : *mémoire virtuelle partagée*, *système de gestion de fichiers parallèles*, *Java*, *OpenMP*.

La gestion efficace des ressources au sein d'une grappe de calculateurs est essentielle pour l'obtention de la haute performance. Nous étudions les concepts qui permettent de gérer la mémoire (mémoire virtuellement partagée), les disques (système de gestion de fichiers parallèles) et les processeurs (migration de processus). Des études sont en cours sur des grappes de PC. Nous nous intéressons notamment à l'intégration de ces différents concepts au sein d'un système distribué pour grappes de PC. Nous utilisons ces mécanismes de gestion de ressources pour concevoir des exécutifs pour des langages parallèles.

Il s'agit d'étudier, de concevoir et de réaliser des *mécanismes système pour une meilleure gestion des ressources* dans une grappe de calculateurs homogènes interconnectés par un réseau à très haut débit. Ce type d'architecture permet d'offrir, pour certaines applications, des performances équivalentes à des multiprocesseurs beaucoup plus coûteux. En effet, le coût d'une grappe de calculateurs évolue linéairement par rapport au nombre de processeurs, ce qui n'est pas le cas des architectures multiprocesseurs offrant un système à image unique pour lesquels il est nécessaire de concevoir des mécanismes d'interconnexion, sur bus mémoire, très performants et donc coûteux.

Notre objectif est de permettre à des composants logiciels parallèles d'exploiter les ressources d'une grappe de calculateurs. Dans ce domaine, nous souhaitons adopter une *approche globale* dans la gestion des ressources (mémoire, disque, processeur). Nous pensons, en effet, que le succès des grappes de calculateurs dépend fortement des capacités à en gérer globalement les ressources. L'utilisateur ne doit voir qu'une seule machine plutôt qu'un ensemble de machines ayant chacune ses propres ressources. Il ne s'agit donc pas de gérer un réseau de calculateurs en machine parallèle à mémoire distribuée comme le font la plupart des projets actuels. Bien que la gestion mémoire (pagination à distance, mémoire virtuellement partagée), la gestion des disques (RAID, systèmes de gestion de fichiers distribués ou parallèles), la gestion des processeurs (placement et migration de processus) sont des mécanismes qui ont fait l'objet de nombreuses études, par notamment les membres de ce projet, très peu d'efforts ont été à ce jour consentis pour concevoir une *gestion de ressources globale et intégrée* dans laquelle l'ensemble de ces mécanismes doivent coopérer pour offrir la meilleure performance. Ceci constitue une des originalités des recherches qui sont menées par le projet PARIS. D'autre part, nous prenons en considération une caractéristique importante associée à l'utilisation de grappes de

[1] Un *middleware* (en français, *intergiciel*) est un logiciel se situant entre un processus client et un processus serveur et qui offre des services supplémentaires

calculateurs : la défaillance d'un constituant de la grappe. Il s'agit notamment de satisfaire à la fois les contraintes de *performance et de disponibilité*.

En parallèle à cette activité système, nous étudions la *conception d'exécutifs pour le support de langages parallèles* sur des grappes de calculateurs homogènes. Un exécutif est un mécanisme logiciel offrant des services spécifiques pour l'exécution de programmes écrits dans un langage particulier. Son objectif est de spécialiser des mécanismes systèmes généraux (gestion mémoire, communication, ordonnancement des tâches, etc.) afin d'atteindre la meilleure performance de l'architecture cible et de son système d'exploitation. L'originalité de notre approche est d'utiliser le concept de mémoire virtuelle partagée comme mécanisme de base pour la communication au sein de ces exécutifs. Nous nous intéressons en particulier à deux langages : Fortran (avec ses extensions parallèles OpenMP [61]) et Java. Le langage Fortran est traditionnellement très utilisé dans les applications de simulation que nous visons. Le langage Java quant à lui commence à être utilisé dans ce domaine en particulier pour des applications irrégulières où les structures de données ne sont pas des tableaux. Il s'agit un peu d'un « pari sur l'avenir » mais qui nous paraît réaliste. En effet, de nombreux travaux sur Java permettent de lever les principales limitations de performance [60].

Pour les deux langages cités, nous nous appuyons sur les mécanismes systèmes étudiés dans le cadre du projet, et principalement sur les mécanismes de gestion mémoire (mémoire virtuelle partagée). En ce qui concerne les langages OpenMP et Java, l'exécution de programmes sur une grappe de calculateurs pose de nombreux problèmes dus à la façon d'exprimer le parallélisme. Dans les deux cas, elle nécessite l'utilisation d'un mécanisme d'adressage global qui n'est normalement pas présent sur une grappe de calculateurs. Là encore, nous nous appuyons sur les mécanismes systèmes, mémoire virtuelle partagée et migration de processus, afin de concevoir des exécutifs spécialisés pour les langages OpenMP et Java.

2.3. Programmation d'une grappe hétérogène de calculateurs

Mots clés : *Programmation parallèle et distribuée*, *metacomputing*, *CORBA*, *grilles de calcul*, *environnement de couplage de codes*, *gestion de données à grande échelle*.

L'accroissement rapide de la performance des calculateurs et des réseaux permet d'envisager des techniques de simulation numérique par couplage de codes. Il s'agit non plus de simuler un seul aspect physique d'un problème mais plusieurs phénomènes physiques simultanément. Ainsi, par exemple, on pourra simuler le comportement d'un satellite dans l'espace en y intégrant la dynamique, la thermique, la déformation de structure et l'optique. L'enjeu est de réduire les temps de conception d'objets manufacturés. Pour permettre cette simulation multi-physique, il est nécessaire d'utiliser un ensemble de ressources de calcul et de stockage de données disponibles sur un réseau afin de permettre l'exécution simultanée, et de façon coordonnée, de plusieurs codes de simulation. L'objectif de ce thème de recherche est de concevoir des technologies qui permettent la construction d'environnements logiciels qui permettent de supporter efficacement l'exécution d'applications de simulation numérique distribuée. Les activités de recherche sont regroupées selon deux thèmes principaux : les grilles de calcul et les grilles de données.

En ce qui concerne cet axe, nous contribuons à la conception d'une infrastructure logicielle, ou *Problem Solving Environment (PSE)*. Ce type d'environnement logiciel est rendu nécessaire par l'utilisation intensive de la simulation numérique lors des étapes de conception d'objets manufacturés, comme les avions ou les automobiles. À titre d'illustration, la conception de la prochaine génération d'avions nécessitera l'utilisation d'une dizaine de milliers de programmes informatiques qui sont développés en interne ou bien par des sociétés éditrices de logiciels [62]. Beaucoup de ces programmes concernent la simulation physique du comportement de telle ou telle partie de l'objet. Cependant, la simulation n'est pas une finalité en soi, elle est partie intégrante du processus d'optimisation dans la conception d'un système physique. Il est donc nécessaire d'intégrer l'ensemble des logiciels de simulation au sein d'un PSE afin de permettre de modifier simplement les paramètres essentiels du système physique et d'observer le résultat de ces modifications par simulation.

Un PSE doit permettre d'exploiter les ressources (processeurs, mémoires, disques) d'une grappe hétérogène de calculateurs, dispersés géographiquement, en vue d'exécuter des applications de simulation numérique distribuée. Nous considérons un ensemble de grappes de calculateurs homogènes et de calculateurs hétérogènes

interconnectés par un réseau local. Un PSE correspond à l'intégration d'un ensemble d'outils et de codes de simulation pour résoudre un problème donné [62]. Ces outils (pour le pré-traitement ou la visualisation) et ces codes de calcul n'ont souvent pas été conçus dans une optique d'intégration. Ils ont été développés le plus souvent sur des architectures particulières (parallèles) ; dans certains cas, les sources ne sont pas disponibles. Ceci impose donc de prendre en compte l'hétérogénéité des codes.

Notre recherche dans ce domaine consiste à identifier des services génériques pour la conception de PSE. Il s'agit, notamment, de concevoir des services au dessus des systèmes d'exploitation qui permettent, comme précédemment, de masquer la distribution physique des ressources. La difficulté provient essentiellement de l'hétérogénéité des ressources (différents calculateurs) et de la difficulté à fournir des niveaux de performance élevés de par l'utilisation intensive des réseaux. L'hétérogénéité impose de prendre en compte les problèmes tels que la représentation de données différente entre calculateurs (et, dans certains cas, entre applications) ainsi qu'une administration et des règles de sécurité spécifiques à chaque machine.

2.3.1. Grilles de calcul

La conception de services pour faciliter la programmation des grilles de calcul est indissociable du modèle de programmation que nous avons choisi pour la conception d'applications en simulation numérique distribuée. Nous rappelons que ce type d'application est un ensemble de codes complexes couplés entre eux. Nous avons donc naturellement adopté une approche par *composants logiciels* comme modèle de programmation. Plus précisément, nous avons choisi d'utiliser une technologie à objet distribué de type CORBA (*Common Object Request Broker Architecture*). Cependant, l'adoption d'une telle technologie pose de nombreux problèmes dus à la nature des applications de simulation numérique. Le concept de composant logiciel doit pouvoir *prendre en compte les particularités du parallélisme* comme par exemple les modèles d'exécution ou l'accès aux données (distribuées ou partagées). Un composant doit pouvoir encapsuler un code composé d'un ensemble de processus qui s'exécutent sur plusieurs processeurs d'une grappe de calculateurs homogènes tout en offrant une interface unique. Il est en effet essentiel de conserver l'intérêt du concept de composant : c'est-à-dire cacher à l'utilisateur les détails de l'implémentation du composant. Pour aborder ce problème, nous avons proposé des extensions à CORBA qui restent toutefois compatibles avec le standard actuel. L'idée que nous proposons est fondée sur le concept de collection d'objets pour implémenter un composant parallèle. Cette démarche offre l'avantage de conserver une approche totalement fondée sur les objets distribués.

La *communication entre composants* est également un point que nous prenons en compte. Dans le domaine de la simulation numérique distribuée, le volume de données transféré entre les différents codes est très élevé (de l'ordre de plusieurs centaines de Méga-octets voire quelques Giga-octets). Il s'agit donc de concevoir des protocoles de communication rapides entre objets ou bien de réaliser des services de gestion de données (répertoire de données) dédiés qui permettent aux composants de s'échanger efficacement des données. Bien entendu, la conception de protocoles ou de répertoires de données doit s'appuyer sur des technologies réseaux qui offrent des performances élevées. Le projet PARIS tient compte des dernières technologies dans le domaine des réseaux de type SAN (System Area Network) ou LAN (Local Area Network) avec pour objectif d'exploiter de nouveaux concepts de communication dans la réalisation de protocoles de communication entre objets distribués. Nous évaluons, par exemple, le concept d'adressage à distance offert dans les réseaux de type SAN (SCI, Memory Channel, Synfinity) et LAN (futur standard *Virtual Interface* de Compaq, Intel et Microsoft).

L'exécution d'une application de simulation numérique distribuée sous forme de composants pose le problème du placement des composants sur les différents calculateurs de la grappe en vue d'utiliser au mieux les ressources de calcul disponibles. Dans une approche fondée sur le concept d'objet distribué, ce problème peut être abordé par des techniques de migration dynamique des objets. Dans notre modèle, un composant peut être vu comme un seul objet (composant séquentiel) ou une collection d'objets (composant parallèle). Pour des raisons d'efficacité, nous nous limitons au cas du traitement de la collection sur une grappe homogène. Dans ce cas, le problème de la migration d'objet, appartenant à une collection, peut être abordé par l'utilisation et l'adaptation des mécanismes de migration de processus issus des recherches du premier axe du projet (cf. section 2.2).

2.3.2. Grilles de données

Si les grilles de calcul ont fait l'objet de nombreux travaux ces dernières années, en revanche, peu d'efforts ont été faits pour faire avancer les aspects liés à la *gestion des données à très grande échelle*, pourtant non triviales. Paradoxalement, on dispose à ce jour d'infrastructures complexes de calcul permettant d'ordonnancer des calculs répartis sur plusieurs sites, alors que le transfert des données vers ces sites est laissé à la charge de l'utilisateur, où, au mieux, des fonctionnalités rudimentaires de type transfert de fichiers (FTP) sont proposées. Ceci devient un facteur limitant dans l'exploitation efficace des grilles de calcul.

Dans ce contexte, il nous semble important de complémenter les recherches sur les mécanismes de gestion des calculs sur la grille par des recherches sur une gestion efficace des données utilisées par ces calculs. L'objectif est de revoir les techniques classiquement utilisées pour la gestion des données partagées à l'échelle d'une grappe à la lumière des nouvelles problématiques liées au passage à l'échelle. Est-ce que les modèles de cohérence et les protocoles utilisés classiquement à l'échelle d'une grappe sont adaptés pour une utilisation à très grande échelle ? Est-ce que les hypothèses fondatrices de ces modèles restent valides ? Est-ce que les contraintes de mise en oeuvre sont les mêmes ? Les réponses semblent négatives aujourd'hui et il est clairement nécessaire de revoir de très près, dans ce domaine, tout un ensemble de principes directeurs et de techniques qui en sont issues.

Le projet PARIS a démarré une étude sur les principes de conception d'un service de partage de mémoire à grande échelle, permettant la mise en place de mécanismes d'accès *transparents* aux données. Un tel service doit être responsable de la gestion de l'allocation, de la localisation, de la réplication, de la cohérence et de la distribution des données. Une application typique pour un tel système est le couplage de code : deux applications s'exécutant sur deux grappes distribuées pourront communiquer à travers une mémoire partagée à grande échelle, qui prendra en charge de manière transparente le transfert et la redistribution des données. Dans ce contexte, les réseaux *pair-à-pair* apparaissent comme en cadre qui illustre bien les caractéristiques d'un environnement réparti à grande échelle. Ces réseaux ont jusqu'à présent été étudiés pour des applications de *partage de fichiers*. Ce type de partage est un cas particulier de partage de données qui correspond à une réplication en lecture seule. Nous nous intéressons au cas plus général du partage de mémoire vive, qui nécessite l'intégration de mécanismes de gestion de la *cohérence* des données répliquées, *modifiables*.

3. Fondements scientifiques

3.1. Introduction

Les activités de recherche du projet PARIS s'appuient sur des bases issues de plusieurs domaines scientifiques : système d'exploitation, intergiciel (*middleware*), programmation par objets distribués ou par composants. Nous avons choisi de présenter ici brièvement quelques fondements de nos recherches : les principes et défis liés à la gestion de ressources (mémoire virtuellement partagée) sous contraintes de performance et de tolérance aux fautes et la programmation de grilles de calcul.

3.2. Mémoire virtuellement partagée

Mots clés : *Mémoire virtuellement partagée*, *gestion de mémoire*.

La gestion des données dans une architecture parallèle à mémoire distribuée est rendue complexe par la distribution physique des mémoires. Ce caractère distribué de la mémoire oblige l'utilisateur ou un compilateur « intelligent » à distribuer les données de l'algorithme devant s'exécuter en parallèle. Cette distribution des données est une opération complexe qui demande une très bonne connaissance de l'application à paralléliser. Cette distribution explicite peut être évitée grâce à des mécanismes de gestion de données permettant la migration de celles-ci en fonction des calculs effectués par chaque processeur. La mémoire virtuelle partagée (MVP) [59] est un exemple de mécanisme de gestion de données. Un tel concept offre un espace d'adressage global pour une architecture parallèle ayant un ensemble d'espaces d'adressage disjoints. L'implémentation d'un mécanisme de MVP s'appuie sur des mécanismes de gestion mémoire virtuelle au sein ou au dessus

d'un système d'exploitation. L'espace d'adressage global est un ensemble de régions de mémoire virtuelle composée de pages migrant à la demande entre les processeurs selon les accès mémoire. Chaque mémoire locale agit comme un grand cache, ou mémoire attractive, contenant les pages précédemment accédées. Comme tout dispositif fondé sur l'utilisation de caches, le problème de la cohérence de ces caches se pose.

Le concept de MVP fournit une vision globale de la mémoire dans laquelle les calculateurs peuvent lire ou écrire. Vis à vis de l'utilisateur, il offre également un modèle mémoire qui caractérise le comportement de la mémoire lorsque plusieurs calculateurs effectuent des accès simultanés. De façon intuitive, l'utilisateur souhaite que la mémoire fournisse toujours le dernier résultat qui a été écrit dans la mémoire. Cependant, dans un système parallèle, la notion de *dernier accès* est ambigüe. Il oblige à définir un ordre total sur tous les accès mémoire, ce qui n'est pas souvent nécessaire. Le modèle de cohérence séquentielle est un exemple de modèle mémoire dont les accès sont consistants avec un ordre total. Un système mémoire possède la propriété de cohérence séquentielle si tous les processus voient les accès mémoire comme si ils avaient été exécutés sur un calculateur séquentiel multiprogrammé. Du point de la vue de la mise en œuvre d'une MVP, un tel modèle impose de nombreuses communications (accès aux pages, invalidation, etc.). Plusieurs travaux ont été réalisés afin de concevoir de nouveaux modèles mémoire à cohérence relâchée pouvant être implémentés plus efficacement sur des systèmes parallèles.

Parmi ceux-ci, le modèle de cohérence à la libération [54] a été l'un des plus étudiés. Le principe de ce modèle mémoire repose sur le constat que les accès aux données, effectués par un programme parallèle, sont souvent synchronisés. Le modèle mémoire à consistance à la libération est fondé sur l'utilisation de deux classes d'opérations sur la mémoire. La première classe regroupe les opérations classiques de lecture et d'écriture tandis que la deuxième classe contient les opérations de synchronisation : libération et acquisition. Le rôle de ces deux opérations est de propager les modifications qui ont été réalisées par les opérations d'écriture. Une opération de libération indique qu'un processeur a effectué des modifications et que celles-ci doivent être communiquées à tout processeur qui effectuera une opération d'acquisition. De même, une opération d'acquisition indique qu'un processeur va exécuter des opérations qui nécessitent la connaissance des modifications effectuées par les processeurs ayant exécuté une opération de libération. Deux formes de cohérence à la libération ont été proposées [56]. La première forme est appelée cohérence à la libération impatiente. Les modifications, réalisées depuis la dernière opération d'acquisition, sont propagées à tous les autres processeurs lors de la libération. La deuxième forme, appelée cohérence à la libération paresseuse, diffère de la précédente par le moment choisi pour diffuser les modifications. Plutôt que de le faire à la libération, les modifications sont propagées lors de l'opération d'acquisition. Lors de l'acquisition, le processeur détermine quelles sont les modifications valides dont il a besoin en fonction de la définition du modèle de cohérence à la libération. Cette approche permet ainsi de réduire fortement le nombre de messages nécessaires au maintient de la cohérence. Une première mise en œuvre de la cohérence à la libération paresseuse a été effectuée au sein de la MVP TreadMarks [57]. Koan et Myoan implémentent une autre forme de cohérence permettant la modification simultanée par de multiples écrivains d'une même page [58].

3.3. Grilles de calcul

Mots clés : *Grille*, *metacomputing*.

Depuis plusieurs années, le développement d'infrastructures logicielles pour la simulation numérique distribuée est une activité très importante dans les grands laboratoires nationaux et les universités aux États-Unis. Cette activité est connue sous le terme de *grilles de calcul*. La plupart des projets de recherche liés aux *grilles de calcul* ont pour objectif de construire un supercalculateur virtuel composé d'un très grand nombre de calculateurs et de supercalculateurs interconnectés par l'Internet. L'idée fédératrice de ces projets procède d'une certaine analogie avec les réseaux fournissant de l'énergie électrique [53]. Il s'agit d'offrir à l'utilisateur un accès le plus transparent possible aux ressources de calcul, quelle que soit la localisation de celles-ci. La nature fortement distribuée d'une *grille de calcul* pose de nombreux problèmes tels que l'allocation conjointe de ressources (co-allocation), la communication entre calculateurs et/ou supercalculateurs, la gestion de données distribuées, la sécurité des accès et des données ainsi que la tolérance aux défaillances. Parmi les

projets de *grilles de calcul* les plus importants, on peut citer Globus [52] et Legion [55] qui proposent des solutions à ces problèmes. D'autres projets moins ambitieux proposent des approches orientées client/serveur telles que NetSolve [51]. Le projet Ninf [63] au Japon adopte une approche similaire à celle de NetSolve. En Europe, les travaux sur le domaine sont peu nombreux. Le Centre Suisse de Calcul Scientifique (CSCS) à Zürich a conçu plusieurs environnements tels que RCS [50] et plus récemment le système ISCN fondé sur le concept d'objet distribué.

3.4. Systèmes pair-à-pair

Mots clés : *Peer-to-Peer*, *gestion de données*, *grande échelle*, *versatilité*, *JXTA*, *cohérence*.

En parallèle avec les développements institutionnels centrés sur le *grid computing*, un autre paradigme de calcul global a récemment focalisé l'intérêt de la communauté scientifique : le calcul *pair-à-pair* (*peer-to-peer computing*). Ce modèle complémente le modèle classique *client-serveur* qui est aujourd'hui à la base de la plupart des traitements sur Internet, en *symétrisant* la relation des machines qui interagissent. La relation client-serveur n'est plus associée aux machines, mais aux transactions : chaque machine peut être client dans une transaction et serveur dans une autre. De ce fait, ce modèle permet d'exploiter d'innombrables ressources de calcul et de stockage situées à la périphérie d'Internet. À titre d'exemple, le réseau *KaZaA* (http://www.kazaa.com/), l'un des réseaux les plus récents de ce type, centré sur le partage de fichiers à grande échelle, regroupe en moyenne, à chaque instant, 3 millions de machines, 500 millions de fichiers contenant 5 péta-octets de données. Il s'agit typiquement de PC connectés à Internet avec intermittence, jusqu'ici remplissant uniquement le rôle de clients vis-à-vis des serveurs centralisés classiques.

En plus du goulot d'étranglement qui rendrait inefficace l'utilisation d'un serveur central accessible par plusieurs machines clientes, il se pose un problème de coût du stockage, lorsqu'il s'agit de plusieurs péta-octets de données. Un autre avantage se situe au niveau de la disponibilité du service : la défaillance d'un serveur centralisé rend automatiquement tous les documents indisponibles pour l'ensemble des machines clientes. Dans un réseau pair-à-pair, la défaillance d'un noeud ne met généralement pas en cause le bon fonctionnement du réseau, grâce à une distribution largement redondante des documents sur les différentes machines.

Il est important de noter que le paradigme pair-à-pair a été centré dès le départ sur la *gestion de larges quantités de données réparties à très grande échelle*. Même s'il a été utilisé jusqu'à présent de manière prépondérante pour le partage de fichiers (*Gnutella*, *Freenet*, *KaZaA*), de nombreux projets de recherche s'y intéressent aujourd'hui pour d'autres types de partage, notamment le partage de puissance de calcul. Une initiative unificatrice pour ce type de projets a été prise par Sun à travers le projet *JXTA* (http://www.jxta.org/), qui met les fondations d'une *infrastructure générique* pour des applications *peer-to-peer*.

3.5. Haute disponibilité

Mots clés : *Disponibilité*, *tolérance aux fautes*.

« Un système informatique réparti est un système où la défaillance d'une machine dont vous ignoriez jusqu'à l'existence peut rendre votre propre machine inutilisable » (L. Lamport).
La *disponibilité* d'un système est définie comme étant la fraction de temps pendant laquelle il fournit le service pour lequel il a été conçu c'est-à-dire qu'il se comporte conformément à ses spécifications. On dit que le système est *défaillant* lorsqu'il ne se comporte pas selon ses spécifications. Une *erreur* est la manifestation d'une *faute* quand la partie fautive du système est activée. Elle peut conduire à la défaillance du système. En vue de fournir des systèmes à haute disponibilité, des techniques de *tolérance aux fautes* fondées sur de la redondance peuvent être mises en œuvre. Elles peuvent être décomposées en quatre étapes. La *détection d'erreur* est à la base de toute technique de tolérance aux fautes. Le *traitement d'erreur* a pour objectif d'éviter que l'erreur conduise à la défaillance du système. Le *traitement de faute* consiste à éviter que la faute soit réactivée. Deux classes de techniques de traitement de faute peuvent être employées : la *réparation* du système qui consiste à remplacer l'élément défectueux et la *reconfiguration* qui consiste à transférer la charge de l'élément défectueux sur les composants valides.

Le traitement d'erreur peut prendre deux formes : la *compensation* d'erreur ou le *recouvrement* d'erreur. La compensation d'erreur est fondée sur des techniques de redondance matérielle ou logicielle utilisées pour masquer l'erreur afin de permettre au système de continuer à fournir le service en dépit de l'erreur. Le recouvrement d'erreur consiste à rétablir un état sain à partir de l'état erroné. Ceci peut être fait par *poursuite* c'est-à-dire par transformation de l'état erroné en un état sain ou par *reprise* c'est-à-dire en substituant un état sain préalablement sauvegardé en mémoire stable, appelé point de reprise, à l'état erroné.

Une *mémoire stable* est un support de stockage qui garantit trois propriétés en présence de défaillances :

(i) non altérabilité : Les données rangées en mémoire stable ne sont pas altérées par les défaillances.
(ii) accessibilité : Les données rangées en mémoire stable restent accessibles en dépit des défaillances.
(iii) atomicité des mises à jour : La mise à jour des données rangées en mémoire stable est une opération effectuée en tout ou rien. En cas de défaillance pendant la mise à jour d'un groupe de données rangées en mémoire stable, soit toutes les données restent dans leur état initial, soit elles prennent toutes leur nouvelle valeur.

4. Domaines d'application

4.1. Simulation numérique distribuée

Mots clés : *simulation numérique distribuée*, *Metacomputing*, *couplage de codes*.

Nos travaux induisent le développement de prototypes logiciels (cf. 5.6, 5.2, 5.3, 5.7, 5.5, 5.4) dont les domaines d'applications sont essentiellement le *calcul haute performance* : image, calcul scientifique, etc.

Le projet s'intéresse prioritairement aux applications de simulation numérique distribuée. Une application de ce type est constituée d'un ensemble de codes parallèles de simulation numérique qui sont distribués géographiquement pour des raisons de disponibilité des ressources ou bien pour des raisons de confidentialité. Dans le premier cas, il s'agit d'utiliser un ensemble de ressources de calcul distribuées sur un réseau, pouvant être à grand échelle, afin de réduire les temps de calcul. Dans le deuxième cas, il s'agit de prendre en compte des contraintes de confidentialité qui imposent qu'un code de simulation soit exécuté sur une machine donnée (chez un partenaire industriel) quelque soit l'impact sur les performances. Plusieurs applications de ce type sont actuellement étudiées par le projet PARIS dans le cadre de contrats.

5. Logiciels

5.1. Introduction

Le projet PARIS développe de nombreux prototypes logiciels de recherche a des fins d'expérimentation. Certains d'entre eux font l'objet d'un dépôt à l'APP et dans ce cas sont disponibles sur le serveur Web du projet. Nous présentons ici *Gobelins*, MOME, PACO ++, PADICOTM, quatre logiciels conséquents développés au sein du projet. Ces quatre logiciels ont fait l'objet d'évolution notable en 2002. Le projet continue d'assurer la distribution de deux autres logiciels (*Do !* et PACO) qui ne font plus l'objet de modifications mais sont cependant encore utilisés par plusieurs équipes de recherche. Ils sont mentionnés brièvement à la fin de ce paragraphe. On en trouvera une présentation détaillée dans les rapports d'activité des années passées.

5.2. Gobelins : un système d'exploitation distribué pour des grappes de PC

Participants : Viet Hoa Dinh, Pascal Gallard, Renaud Lottiaux, Christine Morin, Louis Rilling, Gaël Utard, Geoffroy Vallée.

Mots clés : *système à image unique*, *gestion globale et dynamique des ressources*, *mémoire partagée répartie*, *migration de processus*, *ordonnancement global*, *haute disponibilité*, *tolérance aux fautes*, *recouvrement arrière*.

Contact : Christine Morin

Statut : déposé à l'APP sous le numéro IDDN.FR.001.480003.00.S.C.2000.000.10100, disponible sur le serveur Web du projet (http://www.kerrighed.org/).

Gobelins est un système d'exploitation distribué mettant en œuvre une gestion globale et dynamique des ressources (mémoires, disques et processeurs) d'une grappe de calculateurs pour l'exécution d'applications à haute performance. Gobelins offre aux applications la vision d'une machine unique à haute performance et haute disponibilité. Ainsi, une application *multithreadée* conçue pour une machine multiprocesseur à mémoire partagée peut être exécutée sans modification sur le système Gobelins. Ce système est construit autour du concept de *conteneur* qui est à la base de la gestion globale de la ressource mémoire dans la grappe. Un conteneur est un ensemble de pages qui est identifié de manière unique dans la grappe. Le système Gobelins permet à tous les nœuds de la grappe d'accéder de façon transparente et cohérente aux pages d'un conteneur indépendamment de leur localisation physique. Les conteneurs sont intégrés au sein d'un système d'exploitation hôte grâce à des lieurs qui permettent de construire différents services distribués de haut niveau. Ainsi, Gobelins offre un système de gestion de fichiers distribué offrant une interface de projection et un système de caches coopératifs fondés sur les conteneurs. Le mécanisme de migration de processus utilisé pour l'ordonnancement global des processus sur la grappe s'appuie lui aussi sur les conteneurs grâce auxquels les accès aux fichiers ouverts sur le nœud d'origine d'un processus migré peuvent être réalisés efficacement sur son nouveau nœud d'exécution.

Enfin, le service de gestion dynamique des ressources mis en œuvre par le système Gobelins permet d'ajouter ou de retirer des nœuds à la grappe sans arrêter le système et par conséquent sans perturber les applications en cours de fonctionnement. Ce service pilote également la reconfiguration des services distribués en cas de défaillance pour assurer la haute disponibilité du système malgré la gestion globale des ressources.

Le système Gobelins est mis en œuvre sous forme de modules d'extension au noyau Linux. Quelques fonctions du noyau Linux ont été détournées pour permettre la liaison entre les segments mémoires du système Linux et les conteneurs. En outre, nous avons été amenés à concevoir un système de communication efficace doté d'une interface de niveau noyau et indépendant de la technologie du réseau d'interconnexion sous-jacent pour la mise en œuvre des services distribués du système Gobelins. Ce système de communication appelé GIMLI (Gobelins Interaction Message LIbrary) met en œuvre le concept de port et offre une interface de type envoyer/recevoir ainsi que des messages actifs.

Un prototype du système d'exploitation Gobelins est opérationnel. Le logiciel Gobelins est disponible sur le serveur Web du projet (http://www.kerrighed.org/) depuis le 15 novembre 2002 sous forme d'un logiciel libre portant le nom *Kerrighed*, qui a fait l'objet d'un dépôt de marque. Un logo a été réalisé par le service de l'INRIA pour être associé au logiciel. Une démonstration du logiciel Kerrighed a été présentée sur le stand de l'INRIA lors de la conférence Supercomputing 2002 qui s'est tenue du 16 au 22 novembre 2002 à Baltimore aux États-Unis.

5.3. Mome : une mémoire virtuelle partagée pour des langages parallèles

Participant : Yvon Jégou.

Mots clés : *mémoire virtuelle partagée*.

Contact : Yvon Jégou

La MVP MOME permet la communication par partage de mémoire sur des architectures à mémoires distribuées. MOME met en œuvre un modèle de cohérence relâchée avec écrivains multiples. Le modèle de cohérence relâchée implémenté dans MOME permet à un processeur de demander à voir toutes les modifications qui ont été apportées à une section de mémoire partagée avant un événement de synchronisation, comme une barrière ou un verrou. La mise en œuvre de ce modèle est fondée sur l'utilisation d'une horloge globale à laquelle les requêtes de consistance font référence. L'exploitation de ce modèle de cohérence est rendue possible par les techniques de compilation utilisées dans le domaine du calcul haute performance. Ces techniques sont en effet fondées sur l'analyse à la compilation des fonctions d'accès aux données. Il est de

ce fait possible d'insérer automatiquement dans le code généré des requêtes de consistance sur les données accédées. L'implémentation courante de la MVP *Mome* permet la sélection dynamique par chaque nœud de la grappe du modèle de cohérence, séquentiel ou relâché, qui doit être appliqué. Pour les besoins du projet IST POP (FET), un troisième modèle de cohérence intermédiaire qui garantit une vision identique de l'espace partagé par les nœuds sur les points de synchronisation a été étudié et devrait être intégré dans un avenir proche. Ce nouveau modèle doit permettre la compilation de programmes OpenMP sans faire appel à des techniques complexes d'analyse statique des accès aux données partagées.

La MVP MOME intègre à la fois un exécutif pour langages parallèles (opérations collectives de diffusion, de réduction et de synchronisation ainsi que des verrous) et un support permettant le couplage d'applications parallèles en vue d'une exécution sur des grilles de calcul. Pour le couplage d'applications, MOME fournit un espace d'adressage et de nommage uniforme sous forme de segments que les applications peuvent lier à leur espace d'adressage local par un mécanisme de projection. Deux formes de couplage peuvent être utilisées : le couplage de MVP et le partage de MVP.

Dans la première forme qui met en œuvre un couplage lâche, des transferts de données peuvent être déclenchés directement entre deux MVP à l'initiative des applications. L'interface de couplage fournie par la MVP MOME ne se limite pas à une simple copie de mémoire à mémoire et permet de définir les objets à transférer par des fonctions d'adressage linéaire de type Fortran90. La bibliothèque de couplage se charge également des conversions nécessaires lorsque les grappes à coupler sont hétérogènes. MOME a été utilisée dans le cadre du projet RNRT VTHD pour montrer l'intérêt d'un tel mécanisme pour exploiter des réseaux vraiment à très haut débit (2,5 Gb/s) reliant des grappes de calculateurs. La présence de la MVP simplifie l'organisation des flux de données lorsqu'il faut répartir la charge de transférer les données sur un nombre élevé de machines (chaque machine ayant une capacité de communication nettement plus faible que le réseau), lorsque les deux grappes couplées disposent d'un nombre différent de machines (un couplage point à point ne permet pas de répartir équitablement les transferts) ou lorsque les fonctions d'adressage sont complexes.

Dans la deuxième forme, deux ou plusieurs applications (par ailleurs parallèles) s'exécutent sous le contrôle d'une même MVP et peuvent partager des segments de données. L'exécutif associé à MOME permet de définir dynamiquement des groupes de machines et d'exprimer toutes les opérations collectives sur ces groupes (barrières de synchronisation, opérations de diffusion et de réduction). La mise en œuvre des protocoles de cohérence de la MVP doit également tenir compte de ces regroupements pour exploiter au mieux les phénomènes de localité qui apparaissent au sein des applications couplées ou au sein des hiérarchies de grappes. Une nouvelle mise en œuvre de MOME plus *« grid-aware »* est en cours d'intégration. Elle permet une gestion hiérarchisée des pages de la MVP tenant compte des regroupements formés par les applications et par les grappes.

L'exploitation de la MVP MOME dans un contexte distribué a fait apparaître un certain nombre de besoins nouveaux comme la dynamicité des machines supportant la MVP ou la persistance des données de la MVP. La dynamicité implique de pouvoir ajouter et retirer des machines supportant la MVP pendant l'exploitation. La persistance permet aux applications de venir se connecter dynamiquement à la MVP, d'exploiter et de mettre à jour les données présentes au moment du couplage et d'y stocker des données pour une exploitation future. L'ajout de la dynamicité et de la persistance dans la MVP MOME a été étudié dans le contexte du projet RNRT e-Toile pour la mise en œuvre d'un *répertoire de données à très grande échelle*.

5.4. PadicoTM : une plate-forme d'intégration d'intergiciels et d'exécutifs communicants

Participants : Alexandre Denis, Christian Pérez, Thierry Priol.

Mots clés : *partage de ressources réseaux*, *ORB haute performance*.

Contact : Christian Pérez
Statut : déposé à l'APP sous le numéro IDDN.FR.001.260013.000.S.P.2002.000.10000.

PADICOTM est une plate-forme de recherche dont le but est d'explorer les problèmes d'intégration de plusieurs intergiciels et exécutifs communicants qui sont nécessaires pour la programmation des grilles de calcul. Il vise en particulier les applications de couplage de code basé sur le concept des objets CORBA parallèles. PADICOTM (*Padico Task Manager*) a pour rôle de fournir une infrastructure haute performance permettant de *brancher* des intergiciels ou des exécutifs tels que CORBA, MPI (*Message-Passing Interface*), des DSM (*Distributed Shared Memory*), ou des JVM (*Java Virtual Machine*), etc.

L'architecture de PADICOTM est dérivée de la technologie des composants logiciels. PADICOTM est composé d'un ensemble de modules, chaque module est accompagné d'un fichier de description en XML qui décrit les services et les besoins du module. PADICOTM est logiquement composé de modules de type *noyaux* et de modules de type *services*. Les modules *noyaux* implémentent la gestion des modules, le multiplexage des communications et la gestion des processus légers. PADICOTM utilise Marcel comme bibliothèque de processus légers et Madeleine comme bibliothèque de communications. Marcel et Madeleine appartiennent à l'environnement PM2 développés par le projet ReMaP. Les modules *services* sont basés soit sur les modules *noyaux* soit sur d'autres modules *services* existants. Nous disposons actuellement de plusieurs modules supportant CORBA basé sur OmniORB 3 et 4, MICO 2.3, Orbacus 4.0, d'un module MPI, dérivé de MPI-CH/Madeleine, d'un module fournissant une machine virtuelle Java, basé sur la JVM Kaffe, d'un module fournissant une interface socket. Ce dernier module, utilisé par les modules CORBA et par le module Kaffe, permet de router les messages soit au travers de Madeleine soit au travers de TCP/IP.

Deux nouveaux modules VRP et ParallelStreams ont été ajoutés. VRP est un module qui offre une interface d'envoi de messages mais permettant de contrôler la perte de données. Ce module est destiné à mieux exploiter les réseaux longue distance. Le deuxième module, ParallelStreams, offre une interface socket mais implémente des communications multiflots. Ce module est plutôt destiné à exploiter les réseaux longue distance à haut débit.

PADICOTM a été déposé à APP en juin 2002. Il est diffusé sur Internet via le site du projet PARIS. Fin octobre 2002, on compte une trentaine de téléchargement.

5.5. PaCO ++ : mise en œuvre du concept d'objet CORBA parallèle portable

Participants : André Ribes, Christian Pérez, Thierry Priol.

Mots clés : *CORBA*.

Contact : Christian Pérez
Statut : Prototype en cours de réalisation.

PACO ++ fait suite au développement de PACO. Il s'agit d'une mise en œuvre *portable* du concept d'objet CORBA parallèle. Il s'agit de pouvoir instancier des objets CORBA parallèles sur des ORB standards. La spécification de la distribution n'est plus ajoutée dans l'interface IDL d'un service CORBA mais via un fichier XML associé à celle-ci. Ceci offre l'avantage de ne plus avoir à modifier la syntaxe du langage IDL.

Un prototype est en cours de développement. Un compilateur IDL prend en entrée un fichier IDL ainsi que le fichier XML associé, et génère plusieurs fichiers afin de gérer le parallélisme coté client et coté serveur. Les souches et les squelettes CORBA sont générés avec un compilateur IDL standard mais à partir d'un fichier IDL généré par notre compilateur.

Les redistributions de données sont gérées à partir d'un système de plug-in basé sur un mécanisme d'interface abstraite permettant de supporter simultanément plusieurs bibliothèques de redistribution. Nous avons développé une bibliothèque de redistribution supportant uniquement la distribution bloc à une dimension. Cependant, notre objectif est d'ajouter d'autres bibliothèques de redistribution comme celle de Scalapack ou celle définie par le *DARPA Data Reorganization Effort*.

PACO ++ est basé sur le compilateur de compilateur SableCC ainsi que sur plusieurs scripts en Python. Il supporte actuellement les ORB OmniORB et MICO. Mais, son architecture nous laisse penser que d'autres ORB peuvent être ajoutés sans nécessiter un travail de portage conséquent.

La version courante de PACO ++ contient un certain nombre de limitations que nous prévoyons de lever l'année prochaine. D'une part, elle ne supporte pas des opérations avec des arguments distribués selon différentes distributions, ni ne gère les exceptions parallèles.

Le prototype actuel permet néanmoins de montrer la faisabilité de l'approche. Les premières mesures de bande passante, réalisées dans le cadre du projet RNRT VTHD, indiquent que le prototype est capable d'exploiter des réseaux à vraiment très haut débit (2,5 Gb/s).

5.6. Do ! : Générateur automatique de code Java réparti

Participant : Jean-Louis Pazat.

Mots clés : *framework*, *objets*, *transformation de programme*.

Contact : Jean-Louis Pazat
Statut : Déposé à l'APP sous le numéro IDDN.FR.001.270020.00.R.P.1998.000.10600, disponible sur le serveur Web du projet. Logiciel ne subissant plus d'évolution.

Le logiciel Do ! réalise une génération automatique de code réparti à partir de code Java parallèle centralisé. Le modèle de programmation parallèle est exprimé par un framework, qui permet de limiter l'expression du parallélisme sans modification du langage Java. Le placement des tâches et des données sur les processeurs est dérivé d'indications du programmeur sur les caractéristiques de distribution de son application. Le code généré s'appuie sur le RMI Java et un exécutif (classes Java) permettant la création distante d'objets.

5.7. PaCO : mise en œuvre du concept d'objet CORBA parallèle

Participants : Christian Pérez, Thierry Priol.

Mots clés : *CORBA*.

Contact : Thierry Priol
Statut : déposé à l'APP sous le numéro IDDN.FR.001.480004.00.S.C.2000.000.00000, disponible sur le serveur Web du projet. Logiciel ne subissant plus d'évolution.

PACO est une mise en œuvre du concept d'objet CORBA parallèle au sein de MICO, une implémentation de CORBA réalisée par l'université de Francfort. Un objet CORBA parallèle se présente sous la forme d'une collection d'objets CORBA identiques, décrite par des extensions au langage de spécification d'interface (IDL) Ces extensions se présentent sous la forme de nouveaux mots-clés qui permettent de spécifier le parallélisme et la distribution de données au sein de la collection. PACO est évalué par le CEA et plusieurs équipes de recherche en France et à l'étranger.

6. Résultats nouveaux

6.1. Systèmes d'exploitation pour grappes de PC

Participants : Ramamurthy Badrinath, Viet Hoa Dinh, Pascal Gallard, Renaud Lottiaux, Christine Morin, Louis Rilling, Gaël Utard, Geoffroy Vallée.

Mots clés : *grappe de calculateurs*, *système d'exploitation distribué*, *système à image unique*, *mémoire partagée répartie*, *système de gestion de fichiers parallèle*, *migration de processus*, *ordonnancement global*, *haute disponibilité*, *tolérance aux fautes*, *recouvrement arrière*, *OpenMP*.

6.1.1. Panorama général

Du fait de l'augmentation continue de la puissance des microprocesseurs et de l'évolution de la technologie des réseaux d'interconnexion, les grappes de calculateurs sont devenues des architectures attrayantes pour l'exécution d'applications de calcul et/ou d'accès aux données intensifs. Un problème clé des grappes de

calculateurs est de combiner haute performance et haute disponibilité afin de pouvoir satisfaire les exigences des applications parallèles de longue durée. L'une de nos activités de recherche porte sur la conception et la mise en œuvre d'un système d'exploitation distribué pour grappe, appelé Gobelins. Notre objectif est de construire un système à image unique grâce à des mécanismes de gestion globale et dynamique des ressources pour donner l'illusion qu'une grappe constitue une seule machine à haute performance et à haute disponibilité [25].

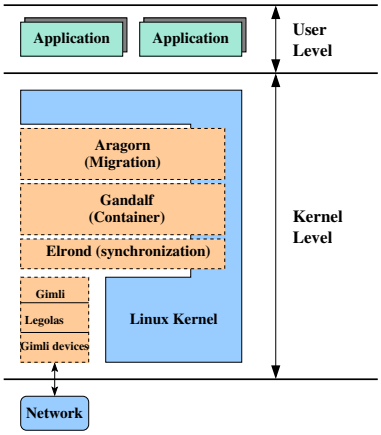

Figure 2. Architecture du système Gobelins.

Les travaux effectués en 2002 portent plus particulièrement sur les aspects suivants : la pagination en mémoire distante, l'ordonnancement global des processus, la gestion dynamique des flux de données, les entrées-sorties à haute performance, le recouvrement arrière des applications parallèles, le support des applications OpenMP, la conception d'un serveur Web fondé sur Gobelins.

L'effort entrepris l'an dernier pour améliorer la robustesse du prototype du système Gobelins en vue de valider nos résultats de recherche sur des applications industrielles d'envergure a été poursuivi cette année grâce à la présence d'un ingénieur associé et d'un post-doctorant industriel.

Cette année, un effort très important a été consenti pour permettre la diffusion du système Gobelins sous forme de logiciel libre. Le système a été expérimenté intensivement à l'aide d'un spectre élargi d'applications parallèles dont Volrend, application de rendu volumique, qui est utilisée à des fins de démonstration et des applications fournies par EDF. Des applications OpenMP et MPI ont également fait l'objet de tests.

Le système Gobelins a été packagé pour permettre son installation sur tout type de grappe de PC. Des scripts d'installation ont été mis au point. Les manuels d'installation et d'utilisation ont été rédigés. Un serveur Web de documentation et de diffusion a été mis en place (http://www.kerrighed.org/). Il permet le téléchargement du logiciel Gobelins disponible depuis le 20 novembre 2002 sous licence GPL.

6.1.2. Pagination en mémoire distante

Participant : Renaud Lottiaux.

Le concept de *conteneur* permet la gestion globale des pages dans une grappe de calculateurs. Ce concept permet de mettre en œuvre le partage des pages de mémoire vive entre processus ou threads s'exécutant sur des nœuds distincts ainsi que le partage des données de fichiers sous la forme d'un système de caches coopératifs [24][41]. Cette année, nous avons étudié et mis en œuvre un mécanisme de pagination en mémoire distante s'appuyant sur les conteneurs. Couplé à un algorithme de remplacement global, ce mécanisme permet de réduire le nombre des accès aux disques en conservant le plus longtemps possible en mémoire les données utiles.

6.1.3. Ordonnancement global de processus

Participants : Christine Morin, Louis Rilling, Geoffroy Vallée.

Compte tenu de la quantité de ressources disponibles sur l'ensemble d'une grappe, il est important de définir une politique d'utilisation de ces ressources qui permette de tirer au mieux parti de toute la puissance disponible. La mise en place de cette gestion des ressources passe par la mise en œuvre d'un ordonnanceur global des processus sur l'ensemble des processeurs de la grappe.

Nous avons conçu et mis en œuvre dans le système Gobelins un ordonnanceur global dont l'architecture est décentralisée. Tous les composants de l'ordonnanceur sont instanciés sur chaque nœud de la grappe et jouent le même rôle. L'architecture de l'ordonnanceur est constituée au plus bas niveau de moniteurs qui analysent le déroulement de l'exécution au sein du système d'exploitation et qui peuvent envoyer des alarmes à l'ordonnanceur lorsqu'une dégradation des performances est observée. Le niveau intermédiaire de l'architecture est constitué de composants mettant en œuvre des politiques d'ordonnancement primitives traitant chacune d'aspects différents de l'exécution. L'ensemble des composants primitifs ont une interface commune qui permet de coupler différentes politiques primitives et de définir des priorités entre ces politiques.

La décision de migration d'un processus est prise au niveau supérieur de l'ordonnanceur global qui interagit avec les différents composants primitifs grâce à leur interface générique. Nous avons étudié plusieurs politiques primitives intéressantes : l'équilibrage de la charge processeur, l'activation d'un *co-scheduling* pour optimiser les applications utilisant un grain fin de synchronisation et de communication, la diminution du coût généré par le partage de mémoire sur la grappe et le placement des threads des applications parallèles.

Cette architecture permet d'étudier différentes politiques d'ordonnancement global. Chaque composant de l'architecture peut être chargé dynamiquement. L'architecture proposée permet aussi de construire simplement un système de batch léger pour limiter la charge globale de la grappe en différant l'exécution de certaines tâches ou en suspendant l'exécution de tâches grâce au mécanisme d'arrêt et reprise d'applications mis en œuvre dans le système Gobelins.

Parallèlement à l'étude de l'ordonnancement global, nous avons poursuivi nos travaux portant sur la mise en œuvre efficace des mécanismes de base de gestion de processus à l'échelle d'une grappe [30]. Le mécanisme de migration de processus ou de threads d'une application parallèle fondée sur le modèle de communication par mémoire partagée est opérationnel [29]. Une étude préliminaire des performances du mécanisme de migration de processus a été menée. Le mécanisme de migration de processus du système Gobelins offre de meilleures performances que celui du système Mosix. Ceci s'explique par l'utilisation du concept de conteneur pour la migration de processus dans le système Gobelins.

6.1.4. Migration de flux de données

Participants : Pascal Gallard, Christine Morin.

Un système d'exploitation traditionnel gère les données sous deux formes essentielles : les pages (ou blocs) et les flux de données. Deux types de périphériques sont distingués : les périphériques blocs comme les disques pour lesquels les données sont gérées en pages et les périphériques caractères comme l'écran, le clavier ou l'interface réseau pour lesquels les données sont gérées sous forme de flux de données.

Le concept de conteneur proposé dans le système Gobelins permet de gérer globalement les pages et par conséquent l'ensemble des périphériques blocs. Nous nous sommes intéressés cette année à la gestion globale des flux de données. Un flux de données possède en général deux extrémités qui dans un système comme Gobelins peuvent se trouver sur deux nœuds distincts. L'une des extrémités d'un flux de données doit également pouvoir être déplacée d'un nœud à l'autre, par exemple lorsqu'un processus fait l'objet d'une migration sur décision de l'ordonnanceur global ou suite à un changement de configuration de la grappe.

Nous avons proposé un système de gestion dynamique de flux de données répondant à ces exigences. Ce système a été mis en œuvre au sein du prototype Gobelins. Il permet actuellement de migrer, au sein d'une grappe, les sockets Unix et les pipes Unix, efficacement et de manière transparente pour les applications. À des fins de validation, des expérimentations sont en cours pour des applications parallèles fondées sur MPI (communication par messages).

La mise en œuvre des concepts de conteneur et de flux dynamique dans le système Gobelins permet à son ordonnanceur global de gérer des charges de travail constituées d'applications variées, séquentielles ou parallèles fondées sur les paradigmes de programmation traditionnels (communication par mémoire partagée

ou par échange de message). Il n'existe pas à notre connaissance d'autres systèmes d'exploitation pour grappe doté d'un ensemble équivalent de fonctionnalités.

6.1.5. Entrées-sorties à haute performance

Participants : Renaud Lottiaux, Christine Morin, Gaël Utard.

L'utilisation optimale d'une grappe passe par la définition d'un système d'entrées-sorties à haute performance. Nous avons étudié un système de gestion de fichiers parallèle et distribué utilisant efficacement les disques de la grappe. Ce système s'appuie sur les conteneurs et en intègre les caractéristiques : cache coopératif et pré-chargement fondé sur de la prédiction. Une plate-forme assure le transfert de pages entre les nœuds, la cohérence du cache, le lien avec la projection mémoire, ainsi que le lien avec l'interface de lecture/écriture. Le système de fichier proprement dit vient se greffer au dessus de cette plate-forme.

Dans un premier temps, nous avons réalisé un prototype en adaptant des systèmes de fichiers existants et nous avons choisi d'assurer la haute disponibilité grâce à un système de RAID logiciel. Mais la plate-forme d'entrée-sortie permet également d'expérimenter des systèmes de fichiers totalement nouveaux, répondant mieux aux contraintes d'une grappe, et spécialisés selon le type d'application.

Dans le cadre de ce travail nous avons été amenés à développer un système de communication spécifique. Celui-ci permet le transfert très rapide d'une page mémoire d'un nœud à un autre, associée à quelques données de contrôle. Il permet également l'appel à distance de fonctions du noyau.

6.1.6. Support des applications OpenMP

Participants : Viet Hoa Dinh, Renaud Lottiaux, Christine Morin, Geoffroy Vallée.

OpenMP est le standard de fait des applications parallèles fondées sur le paradigme de programmation par mémoire partagée sur les machines multiprocesseurs à mémoire partagée. L'objectif du système à image unique Gobelins étant de donner d'une grappe l'image d'un multiprocesseur à mémoire partagée, nous avons tout naturellement étudié les mécanismes à intégrer dans ce système pour permettre l'exécution efficace d'applications OpenMP sur grappe.

Parmi les compilateurs OpenMP, le compilateur développé par Omni génère du code pour une interface Pthread Posix. Nos travaux ont donc porté sur la fourniture de cette interface par le système Gobelins car une telle approche présente l'avantage de limiter les modifications du compilateur OpenMP aux seules optimisations. Le concept de conteneur offre déjà le partage de données entre threads. Afin d'offrir une interface Posix complète, nous avons conçu et réalisé un système de synchronisation de processus efficace offrant des verrous, des sémaphores et des variables conditions pour des threads s'exécutant sur différents nœuds d'une grappe et susceptibles de migrer. Nous avons également été amenés à mettre en œuvre un mécanisme d'allocation dynamique de mémoire fonctionnant en environnement distribué. Les premières expérimentations effectuées ont permis de valider l'interface Pthread Posix du système Gobelins. Ces travaux devront être approfondis à des fins d'optimisation de l'exécution d'applications OpenMP sur grappe.

6.1.7. Haute disponibilité du système et des applications

Participants : Ramamurthy Badrinath, Pascal Gallard, Christine Morin, Geoffroy Vallée.

Les changements de configuration dans une grappe peuvent entraîner une variation de la quantité de ressources disponibles. Cette variation doit être prise en compte par les services de gestion globale des ressources. En outre, la suppression de ressources dans une grappe même suite à une défaillance ne doit pas nuire à la disponibilité de ces services.

Nous avons identifié plusieurs fonctionnalités de base nécessaires à la plupart d'entre eux pour la gestion des changements de configuration et avons proposé de les regrouper au sein d'un service commun de gestion dynamique des ressources, appelé *Legolas*, sur lequel s'appuient les autres services distribués du système pour le traitement des reconfigurations. Parmi les fonctionnalités du service Legolas, nous pouvons citer : un protocole d'intégration ou d'éviction d'un nœud, un protocole de détection de défaillance, un mécanisme permettant de modifier la répartition des gestionnaires de ressources sur les différents nœuds de la grappe en cas de reconfiguration et de localiser efficacement ces gestionnaires nomades [21][37]. Le service Legolas [22]

a été mis en œuvre au sein du système Gobelins et a servi de base à la mise en œuvre du service de gestion dynamique des flux de données.

Grâce au service Legolas, le système d'exploitation de la grappe reste disponible en dépit de la défaillance d'un nœud. Cependant, la défaillance d'un nœud entraîne l'arrêt de l'exécution des applications utilisant ses ressources. Nous étudions des techniques de recouvrement arrière d'applications parallèles à intégrer au système Gobelins afin de décharger les programmeurs de la gestion complexe de la tolérance aux fautes.

Les travaux que nous menons visent à permettre l'établissement de points de reprise pour n'importe quelle application s'exécutant au dessus du système Gobelins : application séquentielle, application parallèle communiquant par mémoire partagée ou par échange de messages. À partir de l'étude des protocoles traditionnels de sauvegarde de points de reprise d'applications parallèles et de leurs optimisations, nous avons identifié un ensemble de fondements qui leur sont communs pour garantir l'existence d'une ligne de recouvrement. Nous avons proposé une modélisation des entités du système, à savoir les processus, les pages partagées et les messages, et des informations qui doivent leur être associées en vue de la mise en œuvre des protocoles de sauvegarde et de restauration de points de reprise d'applications parallèles. Nous avons montré comment mettre en œuvre efficacement dans ce cadre les différents protocoles de la littérature, protocoles de sauvegarde coordonnée de points de reprise dits pessimistes, protocoles de sauvegarde indépendante de points de reprise dits optimistes et protocoles de sauvegarde de points de reprise induite par les communications [32].

Les mécanismes communs aux différents protocoles ont été mis en œuvre au sein du système Gobelins. En outre, le mécanisme de migration de processus réalisé antérieurement a été généralisé afin de permettre la sauvegarde de l'état privé d'un processus. Ainsi, il est possible d'extraire l'état d'un processus du système et de choisir sa destination : réseau pour une migration de processus ou mémoire distante ou disque local ou distant pour la sauvegarde d'un point de reprise.

Tous ces mécanismes ont été expérimentés dans un premier temps pour des applications séquentielles et des applications parallèles à mémoire partagée. Un protocole de sauvegarde coordonnée de point de reprise dérivé de celui étudié dans [16] est opérationnel et ses performances sont en cours d'évaluation.

Lorsque la mise en œuvre sera achevée, il sera possible de comparer expérimentalement sur une même plate-forme différentes techniques de recouvrement arrière pour les applications parallèles qu'elles soient fondées sur le modèle de communication par mémoire partagée (applications OpenMP, par exemple) ou par échange de message (applications MPI, par exemple). Ainsi, le programmeur pourra choisir le type de protocole de recouvrement arrière le mieux adapté à son application.

6.1.8. Système de communication fiable, portable et efficace pour la mise en œuvre de services systèmes distribués

Participants : Viet Hoa Dinh, Renaud Lottiaux.

Une première version d'un système de communication pour la mise en œuvre des services distribués du système Gobelins avait été réalisée. En 2002, l'architecture de ce système a été reconsidérée pour améliorer sa portabilité tout en conservant la possibilité d'optimiser ses performances pour certaines technologies de réseau.

Le nouveau système de communication est structuré en deux niveaux : le niveau supérieur offre l'interface envoyer/recevoir et les messages actifs aux services du système, le niveau inférieur est constitué de différents pilotes mis en œuvre dans des modules Linux, chacun adapté à une technologie de réseau particulière. Cette nouvelle version du système de communication présente l'avantage de pouvoir exécuter le système Gobelins sur tout type de grappe de PC sans recompilation du noyau. Le système de communication est auto-configurable.

Au niveau supérieur, nous avons introduit un protocole de contrôle de flux et amélioré le système d'acquittement des messages pour des raisons de performance.

Au niveau inférieur, nous avons conçu et mis en œuvre un pilote générique permettant d'assurer la portabilité du système Gobelins sur tout type de réseau, un pilote TCP développé initialement à des fins de mise au point

mais qui peut servir également dans le cadre de fédérations de grappes et un pilote optimisé pour des réseaux Ethernet ou Gigabit Ethernet.

Une évaluation des performances des différentes configurations du système de communication a été menée sur des réseaux Fast Ethernet, Gigabit Ethernet et Myrinet.

6.1.9. Weblins : un serveur Web fondé sur Gobelins

Participante : Christine Morin.

Ce travail a été fait dans le cadre de la co-direction par C. Morin de la thèse de Ahmad Faour, de l'Université libanaise,

Le système Gobelins offre des fonctionnalités attrayantes pour servir de support à un serveur Web implanté sur une grappe. En particulier, les serveurs Web ont des contraintes de haute disponibilité, d'extensibilité et de performance, la performance se mesurant en nombre de requêtes indépendantes traitées par seconde et volume d'entrées-sorties. Une spécialisation des algorithmes implantés dans le système doit être effectuée pour répondre aux besoins des serveurs de données sur Internet. En particulier, les algorithmes de remplacement et de pré-chargement de données spécifiques doivent être étudiés. En outre, le placement des données et leur réplication sur les disques de la grappe doivent être effectués judicieusement pour une bonne utilisation de l'ensemble des ressources de la grappe en présence de données dont la popularité est variable.

Ces considérations nous ont amené à définir l'architecture de serveur Web *Weblins* sur grappe qui exploite les caractéristiques du système Gobelins tout en spécialisant les algorithmes de gestion globale des ressources.

Nous avons, cette année, réalisé un simulateur d'architectures de serveurs Web qui nous a permis de comparer l'architecture Weblins à d'autres architectures de serveurs Web fondées sur des grappes : architecture simple avec répartition des requêtes en tourniquet, de manière aléatoire ou avec priorité aux nœuds les moins chargés, architecture LARD décentralisée et architecture fondée sur le système Gobelins non modifié. Ce simulateur a été utilisé pour évaluer différents algorithmes de gestion globale des caches de données : LRU, LFU, GDSF, CGDSF (implanté dans Weblins).

6.1.10. Support système pour la fédération de grappes

Participants : Christine Morin, Louis Rilling.

Les grappes occuperont une position dominante sur le créneau de la haute performance. On peut imaginer qu'une grande entreprise disposera sur un ou plusieurs sites de différentes grappes de taille raisonnable. Plusieurs motivations peuvent conduire à fédérer ces grappes sur un réseau à très haut débit : utiliser dans un mode pair à pair les ressources disponibles de ces grappes pour optimiser les investissements informatiques, construire à partir de plusieurs grappes une infrastructure permettant d'exécuter occasionnellement des applications nécessitant des ressources dépassant les capacités d'une seule grappe, déployer des applications distribuées constituées de plusieurs composants coopérants, les différents composants s'exécutant sur différentes grappes.

Dans le contexte actuel, une hypothèse fondamentale pour les travaux sur les grilles de calcul est que les machines qui en constituent les nœuds sont fortement hétérogènes et exploitées sous des systèmes d'exploitation propriétaires. Pourtant, les évolutions actuelles laissent présager d'une beaucoup plus grande homogénéité : PC omniprésents, domination de deux systèmes d'exploitation sur le marché des PC, Linux et Windows. Face à cette homogénéisation des nœuds de la grille et dans le contexte visé, gérer les ressources des applications à grande échelle au sein de *middleware* tend à perdre de l'intérêt. L'empilement de services de gestion de ressources introduit en effet de la redondance de mécanismes et la possibilité de décisions contradictoires dans les différents niveaux.

Partant de ce constat, nous avons initié une activité de recherche visant à concevoir un système d'exploitation *Grid-aware* c'est-à-dire intégrant les mécanismes permettant de mettre en place une infrastructure pour exploiter les ressources d'une fédération de grappes pour des applications à grande échelle.

Pour ces travaux, nous faisons l'hypothèse que chaque grappe dispose d'un système à image unique tel que Gobelins. Il s'agit d'étudier les extensions à apporter à un tel système pour faciliter l'intégration d'une grappe

au sein d'une fédération et satisfaire les exigences de performance des applications à grande échelle dans le cadre d'une fédération de grappe.

6.2. Mémoire virtuelle partagée comme support d'exécution de OpenMP

Participants : Yvon Jégou, Christian Pérez.

La programmation des grappes de PC se distingue de celle des autres architectures parallèles (de type SMP[2] ou bien cc-NUMA[3]) par l'absence d'un espace d'adressage global forçant ainsi le programmeur à utiliser des exécutifs de très bas-niveau (échange de messages). Si des efforts ont été réalisés, dans le passé, pour fournir des langages de programmation de plus haut niveau (comme HPF[4] pour la programmation des grappes), il faut cependant reconnaître que leur succès a été limité. Cet échec s'explique essentiellement par le spectre très étroit des applications pour lesquelles il était possible d'obtenir une bonne performance et, aussi en partie, par la diffusion rapide des architectures SMP et cc-NUMA. Une initiative pour la standardisation de directives de parallélisation pour ce type d'architectures a donné naissance à OpenMP. OpenMP spécifie un ensemble de directives pour les langages C/C++ et Fortran qui permettent d'indiquer notamment quelles sont les boucles qui peuvent être exécutées en parallèle. OpenMP prend pour hypothèse que les variables sont stockées dans un espace d'adressage partagé.

L'exécution de programmes OpenMP sur grappe de PC pose de nombreux challenges dus à l'absence d'un espace d'adressage global. L'objectif de nos recherches est d'étudier quels sont les mécanismes nécessaires pour une exécution efficace de programmes OpenMP sur ce type de machine. Nous étudions notamment l'utilisation du concept de mémoire virtuellement partagée comme mécanisme de base à un exécutif pour un compilateur OpenMP. Ce travail est réalisé dans le cadre du projet IST POP (FET) dans lequel les chercheurs du projet sont responsables de la partie *exécutif à base de mémoire virtuellement partagée*.

Durant la première année du projet, une analyse des besoins a été menée. Nous avons obtenu une spécification de l'interface entre l'exécutif proprement dit et les sous-modules. Le sous-module dans lequel nous sommes impliqué, à savoir le sous-module SDSM, possède ainsi une interface qui permettra aux partenaires du projet de tester différentes stratégies d'implémentation, allant d'une solution où tout (ou presque) est partagé à une solution où tout (ou presque) est privatisé. Dans le premier cas, il s'agit de simuler au plus près une machine à mémoire partagée : même le code est chargé via la DSM. L'avantage principal attendu est de pouvoir exécuter du code binaire compilé pour SMP. La question ouverte concerne les performances qu'il est possible d'obtenir. Dans le deuxième cas, il s'agit tout au contraire d'adapter le code au maximum à l'environnement distribué dans lequel il s'exécute. Ainsi, en ayant un contrôle le plus fin possible sur les communications, les performances devraient être au rendez-vous. S'il est envisageable de pouvoir traiter les codes de calcul purs programmés en OpenMP, il existe des contraintes sur les bibliothèques qui peuvent être utilisées ainsi que sur les opérations d'entrée-sortie.

Afin de permettre aux partenaires du projet POP de tester ces différentes solutions, nous avons ajouté de nouvelles fonctionnalités à la SDSM MOME. Ainsi, les possibilités de MOME pour l'envoi et la réception de messages a été étendue. Il est maintenant possible d'envoyer et de recevoir des messages de tailles quelconques. De même, les identifiants de verrou peuvent maintenant être pris dans l'espace d'adressage du processus, ce qui permet d'en créer dynamiquement. Enfin, les piles des processus légers peuvent être en mémoire partagée. Les défauts de pages sur la pile sont correctement gérés.

Un prototype de l'exécutif d'OpenMP développé dans le projet POP basé sur la SDSM MOME a été testé. Il nous permet ainsi de valider les différents choix effectués et de mieux comprendre les problèmes restants comme le support des variables globales.

6.3. Grilles de calcul

Participants : Alexandre Denis, Benoît Hubert, Guillaume Mornet, Christian Pérez, Jean-Louis Pazat, Thierry Priol, André Ribes.

[2] Symmetric Multi-Processing
[3] Cache-Coherent Non-Uniform Memory Access
[4] High-Performance Fortran

Mots clés : *metacomputing*, *CORBA*, *Java*, *framework*, *objets*, *composants*, *transformation de programme*, *couplage de codes*.

L'accroissement des performances des calculateurs et des réseaux permet d'envisager de nouvelles applications dans le domaine de la simulation. Il est ainsi possible de coupler plusieurs codes de calcul afin d'améliorer la qualité des résultats en prenant en compte un plus grand nombre de phénomènes physiques. L'utilisation de réseaux à haut débit permet également d'envisager la visualisation des résultats produits par un supercalculateur quelque soit la distance qui sépare le supercalculateur du système de visualisation. Cette activité a pour objectif de contribuer au développement de technologies qui permettent la conception d'environnement de *metacomputing*. Nos travaux portent principalement sur des extensions au concept d'objets distribués en y intégrant le parallélisme, la génération automatique de code réparti, la conception de mécanismes de communication efficace entre objets distribués, les répertoires de données et la conception d'environnements logiciels pour la programmation par composants logiciels.

6.3.1. Une plate-forme d'intégration d'intergiciels et d'exécutifs communicants

Participants : Alexandre Denis, Benoît Hubert, Christian Pérez, Thierry Priol.

Mots clés : *partage de ressources réseaux*, *intergiciels*, *exécutifs*, *ORB haute performance*.

L'exécution d'applications de simulation numérique distribuée sur des grilles de calculs nécessite l'utilisation de plusieurs intergiciels et/ou exécutifs ayant accès aux ressources réseaux. Il se pose alors un problème nouveau : la cohabitation de plusieurs intergiciels et/ou exécutifs au sein d'un même programme. Par exemple, les objets CORBA parallèles requièrent l'utilisation *simultanée* de l'intergiciel CORBA et de l'exécutif MPI. C'est à dire, qu'un ORB doit cohabiter avec un exécutif MPI. Cette cohabitation soulève en particulier des problèmes d'accès aux réseaux. Non seulement les différents systèmes doivent coopérer alors qu'ils n'ont a priori aucune connaissance les uns des autres mais en plus cette coopération doit être obtenu sous des contraintes de hautes performances. Le second problème avec les grilles de calculs est qu'il faut être capable de transporter les communications d'un intergiciel, comme par exemple CORBA sur *tous* les réseaux, allant des SAN au WAN, disponibles au sein d'une grille.

Nous avons proposé de bâtir une infrastructure permettant de brancher des intergiciels et/ou des exécutifs. Cette infrastructure est basée sur un modèle d'exécution qui garantit une cohabitation efficace des différents intergiciels et exécutifs. Le prototype PADICOTM a permis de démontrer la pertinence de la proposition en obtenant plus de 96 % de la bande passante d'un réseau Myrinet 2000, aussi bien avec MPI qu'avec CORBA.

En 2002, nous avons généralisé nos travaux sur les mécanismes d'intégration d'intergiciels et d'exécutifs communicants. Nous avons proposer une architecture d'un environnement de communication pour les grilles de calcul. Cette architecture ambitionne d'étendre l'usage des grilles de calcul en autorisant l'exécution d'applications distribués et/ou parallèles sans contraindre l'utilisation d'une couche de communication particulière. Ainsi, les deux principaux paradigmes de programmation, *parallèles* et *distribués*, peuvent être utilisés quelque soit les technologies réseaux disponibles dans une grille : CORBA peut être transporté sur des réseaux Myrinet via une couche de communication telles que BIP ou bien MPI peut être déployé sur des réseaux longue distance.

Le principe de notre architecture, présentée en figure 3 repose sur la séparation des fonctionnalités en trois couches : une couche *d'arbitrage*, une couche *d'abstraction* et un ensemble de personnalités.

La couche basse est la couche *d'arbitrage* dont le rôle est d'offrir un accès multiplexé, réentrant et efficace pour toutes les technologies de réseaux. Cette couche n'offre pas une interface uniforme mais utilise chaque technologie réseau avec le paradigme le plus approprié. Ces multiples interfaces reposent cependant sur le principe du *callback* pour assurer l'efficacité, la réentrance et le multiplexage. La couche *d'arbitrage* est donc le seul client des ressources réseaux.

Au dessus de la couche *d'arbitrage*, la couche *d'abstraction* fournit des services de plus haut niveau, indépendemment des ressources. Son rôle est de fournir des interfaces adaptées aux intergiciels et/ou exécutifs. Elle fédère les services de la couche *d'arbitrage* en services abstraits. Ainsi, un intergiciel et/ou un exécutif utilisant ces services abstraits ne sait pas s'il utilise un réseau Myrinet ou un réseau longue distance : il

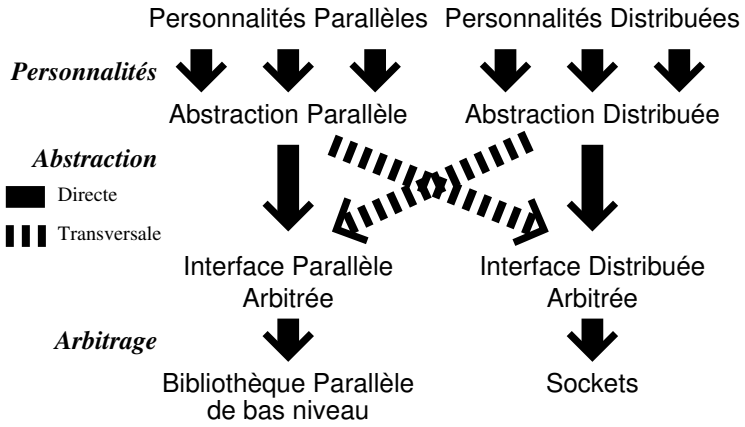

Figure 3. Architecture d'un environnement de communication à trois couches pour les grilles de calcul.

utilise toujours la même interface de programmation. La couche *d'abstraction* choisit automatiquement et dynamiquement le *meilleur* service de la couche d'arbitrage en fonction des ressources disponibles.

La couche d'abstraction fournit des interfaces abstraites, qui ne sont pas forcément des interfaces standards des paradigmes de programmation parallèles ou distribués. Afin de prendre en compte les intergiciels et/ou exécutifs existants, des personnalités, qui offrent des interfaces standards, sont placées au dessus de la couche d'abstraction. Les personnalités sont de fins adaptateurs entre une interface standard et un service abstrait particulier. Elles ne sont pas sensées réaliser une adaptation d'un protocole

Ces notions ont été intégrées dans PADICOTM [14][20][18][12], notre plate-forme d'intégration d'intergiciels et d'exécutifs communicants. Chaque interface abstraite, personnalité et support de protocole a été implémenté dans son propre module dynamiquement chargeable. Actuellement, PADICOTM supportent les personnalités offrant les interfaces socket, les entrées-sorties asynchrones (AIO), Madeleine et Fast Messages. Les exécutifs supportés par PADICOTM sont MPI-CH, OmniORB 3 et 4, Orbacus, MICO et Kaffe. Les performances obtenues montrent que les exécutifs ne faisant pas de recopie de données arrivent à exploiter environ 240 Mo/s sur les 250 Mo/s disponible sur Myrinet 2000. PADICOTM n'ajoute qu'une demi-microseconde à la latence, ce qui permet à MPI-CH d'avoir les mêmes performances avec ou sans PADICOTM.

De nouvelles fonctionnalités sont en cours d'ajout dans PADICOTM. Ainsi, si la plate-forme supporte bien les réseaux locaux, rien de particulier n'avais été fait pour les réseaux haut débit longue distance comme VTHD. Joel Daniel, étudiant américain financé par le *International Research Opportunities Program* de l'UNH, a pendant trois mois développé une interface multisocket dans PADICOTM. En effet, du fait de la gestion de la perte de paquet avec le protocole TCP/IP, il apparaît intéressant de transporter les données via plusieurs sockets. L'interface utilisateur ne changeant pas, le service reconstruit les données. Les résultats préliminaires montrent une augmentation significative de la bande passante. Il serait cependant intéressant de reconsidérer ce problème avec le nouveau protocole *Stream Control Transmission Protocol* qui commence à être déployé. En effet, SCTP offre des mécanismes de gestion de multiflot.

Une deuxième fonctionnalité concerne également les réseaux longues distance. L'objectif est de supporter les protocoles à perte de données ajustable. Le principal intérêt de ce genre de protocole est qu'il permet d'améliorer très nettement la bande passante sur les réseaux chargés tout en offrant un modèle de programmation raisonnable. En effet, ces protocoles permettent de spécifier quelles sont les parties d'un message qui ne peuvent pas être perdu, comme le contrôle, et quel est le pourcentage de perte toléré sur les données.

Le logiciel PADICOTM a été déposé à l'APP en juin 2002. Il est diffusé sur Internet via le site du projet PARIS. Fin octobre 2002, on compte une trentaine de téléchargement.

6.3.2. Objets CORBA parallèles portables

Participants : Alexandre Denis, Christian Pérez, Thierry Priol, André Ribes.

Ayant proposé et validé le concept d'objets CORBA parallèle portable [13], nous avons, cette année, réalisé une mise en œuvre (logiciel PACO ++). Notre approche consiste à spécifier ce qui ressort de la gestion du parallélisme dans un fichier annexe [15] sous un format XML.

Durant l'année 2002, nous avons conçu un compilateur capable de lire la spécification IDL et le fichier d'annotations XML. Le compilateur génère des talons et des squelettes parallèles qui sont quasiment indépendants d'une implémentation de CORBA. Les bibliothèques de redistribution sont gérées comme des *plug-ins*. Le code généré est ainsi indépendant de la bibliothèque de redistribution effective. À terme, un avantage attendu est de pouvoir supporter simultanément plusieurs bibliothèques de redistribution et de pouvoir en changer dynamiquement. La chaîne de compilation pour les objets CORBA parallèles est présentée en figure 4.

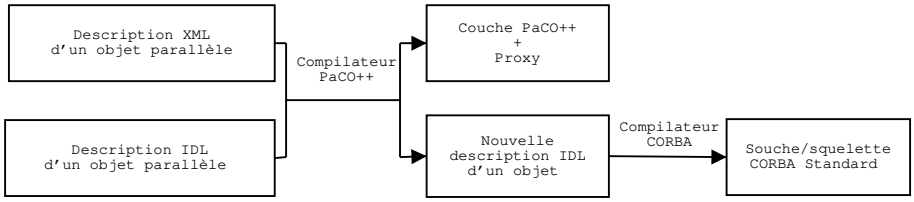

Figure 4. Chaîne de compilation des objets CORBA parallèles portables.

Les talons et les squelettes utilisent également quelques primitives de communications, principalement pour s'identifier et réaliser des opérations de synchronisations. Afin de n'être pas dépendant d'une bibliothèque de communications, les communications sont aussi virtualisées. L'implémentation actuelle supporte deux bibliothèques de communication, à savoir MPI et MPCircuit. MPCircuit a été développé dans PADICOTM pour circonvenir à quelques limitations du modèle MPI, dont notamment de ne pas pouvoir créer des communicateurs en mode MIMD. En effet, comme il n'y a pas de synchronisation lors de la création de deux objets CORBA parallèles, nous ne pouvons pas garantir que la primitive sera appelée dans le même ordre sur tous les nœuds.

Une application d'EADS, obtenue dans le cadre de l'ACI GRID RMI, a servi de base pour une démonstration de la pertinence des objets CORBA parallèles à SuperComputing 2002. Cette application est constituée de deux ordonnanceurs locaux, chacun gérant une succession de code MPI. Les deux ordonnanceurs doivent se synchroniser et surtout échanger des données produites par les codes MPI. À l'origine, l'échange se faisait via un système de fichier commun, ce qui empêchait l'application d'être déployée sur une grille. Nous avons introduit un échange des données via les objets CORBA parallèles. En plus de permettre d'assurer le transport de donnée efficace, cette solution a l'avantage d'éviter la sérialisation lors des phases de lecture et d'écriture.

6.3.3. Concept de composant CORBA parallèle

Participants : Christian Pérez, Thierry Priol, André Ribes.

En juillet 2002, la spécification de CORBA 3.0 est sortie. De fait, les composants logiciels sont la spécification courante de CORBA. Notre extension sur les composants logiciels peut donc maintenant s'appuyer sur une spécification finale.

Les travaux menés dans PACO ++ ont pour but de servir de fondation aux composants logiciels parallèles. En effet, à l'exécution les composants CORBA sont des objets CORBA avec des propriétés particulières. Les extensions parallèles définies pour un objet CORBA sont transposables à une facette d'un composant parallèle. Les réflexions ont concerné les liens entre un composant parallèle et ses facettes. Le problème, similaire à celui entre un objet CORBA parallèle et ses opérations, est la localisation de l'exécution d'une facette (resp. opération) par rapport à un composant (resp. objet) parallèle. Nous pensons qu'il est souhaitable de pouvoir définir des facettes (resp. opérations) séquentielles à l'intérieur d'un composant (resp objet) parallèle. De plus ces facettes (resp opération) peuvent être contraindre, par exemple toujours sur le même servant ou non.

Le développement de GRIDCCM, bien que basé sur PACO ++ est fortement lié à la disponibilité d'une implémentation C++ des composants CORBA. Notre approche est d'enrichir les implémentations de CCM de manière la plus portable possible. C'est pourquoi notre effort s'est principalement porté sur PACO ++.

Durant l'année 2002, nous avons principalement contribué à la diffusion du concept de composant CORBA parallèle [11][27][26]

6.3.4. Framework applicatif pour la simulation en Java

Participants : Guillaume Mornet, Jean-Louis Pazat.

En 2002, nous avons utilisé le prototype Do ! conçu dans le cadre de la thèse de Pascale Launay.

Lors du stage d'été de Maxime Glaizot (étudiant en Maîtrise IFSIC), nous avons pu mettre en évidence un certain nombre de difficultés d'utilisation du prototype Do ! pour la distribution d'un logiciel de modélisation de croissance des arbres réalisé à partir des travaux du CIRAD. La transformation automatique d'un programme Java parallèle en programme distribué pose un certain nombre de problèmes que nous n'avions pas identifié et il semble intéressant de pouvoir décider de la réplication de certaines classes ou de leur localisation de manière explicite ce que le framework de Do ! ne permet pas.

Plutôt que de poursuivre le développement de la plate-forme Do ! il nous semble plus intéressant de reprendre les idées développées dans le cadre de Do ! , c'est à dire la description du parallélisme et de la distribution par des collections pour les appliquer à d'autres environnements. Nous avons donc évalué la possibilité d'utiliser les collections avec la bibliothèque *ProActive* réalisée à l'INRIA Sophia par Denis Caromel. Une première étude réalisée lors du stage de DEA de Taoufik Korchi (Université d'Orléans) a permis de montrer la faisabilité de l'approche et la possibilité d'introduire des constructions structurées dans *ProActive*.

6.4. Grilles de données

6.4.1. Couplage de codes via un espace d'adressage global

Participant : Yvon Jégou.

La communication au sein d'une grappe hétérogène de calculateurs peut être effectuée par d'autres mécanismes que ceux actuellement utilisés (*RPC*, courtier d'objets ou échange de messages). Nous avons entrepris une étude sur le couplage de mémoires virtuellement partagées. Le système distribué que nous visons est une grille de calcul constituée d'une collection de grappes et de machines parallèles ayant une forte hétérogénéité. Chaque grappe ou machine parallèle est dotée d'une mémoire virtuellement partagée. L'objectif est d'offrir un espace d'adressage global sur l'ensemble de ces grappes et de ces machines parallèles. Plusieurs problèmes se posent : la gestion de l'hétérogénéité des machines (représentation des données, taille de page, placement des données en mémoire, système d'exploitation), la cohérence des données (sachant que chaque mémoire virtuellement partagée peut avoir son propre protocole de cohérence), le transfert de données entre plusieurs machines en exploitant les ressources réseaux disponibles, l'ajout et le retrait de calculateurs de la grille, la durée de vie et la persistance des données du répertoire (possibilité offerte aux applications de venir se connecter à une MVP existante, conservation des données par la MVP après l'arrêt d'une application). Les développements en cours sur la MVP MOME prennent en compte un certain nombre de besoins : support d'exécution de langages parallèles (OpenMP dans le projet IST POP), couplage direct d'applications parallèles (lancement de plusieurs applications sur une même MVP), hétérogénéité des couches de communication (grappes de grappes, grappes reliées par un réseau longue distance à haut débit) par la mise en œuvre d'un protocole de gestion hiérarchisé, l'hétérogénéité dans les représentations des données par le couplage de MVP (expérimentations dans le cadre du projet RNRT VTHD), la persistance des données (projet RNTL e-Toile). Les expériences réalisées dans le cadre du projet RNRT VTHD ont montré qu'il était réaliste de coupler efficacement deux applications parallèles s'exécutant sur deux grappes distantes (Rennes et Sophia-Antipolis) reliées par un réseau haut débit (débit supérieur à 920 Mb/s, proche de la limite de 1 Gb/s, [23]), même lorsque le nombre de machines supportant les applications parallèles est différent sur les deux sites (cas qui génère un schéma de communication irrégulier).

6.4.2. Gestion de données dans des systèmes vraiment à grande échelle

Participants : Gabriel Antoniu, Luc Bougé, Sébastien Lacour, Thierry Priol.

Notre objectif est de concevoir un premier prototype qui permette de partager des données au niveau d'une constellation hétérogène de grappes, à l'échelle d'un pays comme la France, fondé sur une approche *Peer to Peer*. Les grappes sont réparties à Rennes, Paris, Grenoble, Lyon et Nice. Elles sont interconnectées par le nouveau réseau VTHD à 2,5 Gb/s.

Comme première étape, nous avons commencé une étude sur les problématiques nouvelles liée à la conception d'un système de partage de mémoire à grande échelle. Nous avons identifié plusieurs nouvelles hypothèses, dont notamment les suivantes.

Dynamicité du réseau sous-jacent. Les réseaux pair-à-pair supportent des milliers de noeuds qui peuvent se connecter et se déconnecter du réseau de manière dynamique pendant l'exécution d'un calcul. Cette variabilité fait apparaître des problématiques proches de celles des systèmes distribués tolérants aux pannes. Les études sur les systèmes à mémoire virtuellement partagée tolérants aux pannes peuvent s'avérer très utiles pour ce type d'étude. Par ailleurs, il y a également une variabilité dans le temps du *degré* de disponibilité des pairs. Les pannes deviennent alors des cas particuliers où la disponibilité d'un noeud devient nulle. Le projet *JXTA* (http://www.jxta.org/) propose une *infrastructure générique* pour des services *peer-to-peer* qui peut constituer un bon support pour la conception et la mise en œuvre d'un système de partage de mémoire pair-à-pair. Nous avons réalisé une première étude sur ce thème et nous avons commencé la réalisation d'un premier prototype sur JXTA (stage de maîtrise de Parkshit Machwe, étudiant du IIT, Inde).

Hétérogénéité. Dans un réseau de très grande taille, l'hypothèse habituelle d'homogénéité des ressources matérielles n'est plus réaliste. Les caractéristiques des différentes ressources mises en jeu (débit réseau, fiabilité, niveau de sécurité, degré de disponibilité) seront également hétérogènes. La portabilité et l'interopérabilité deviennent particulièrement cruciales.

Architecture hiérarchique. À la différence des grappes, caractérisées le plus souvent par une architecture *plate*, le passage à l'échelle impose le plus souvent une architecture hiérarchique, que l'on peut modéliser par des grappes de grappes. Pour exploiter efficacement ce type d'architecture, il est nécessaire de mettre en œuvre des protocoles de cohérence multi-niveaux, qui prennent en compte la structure hiérarchique des réseaux : interconnexions rapides au sein des grappes et beaucoup plus lentes entre les grappes. Nous avons conçu un protocole de cohérence qui montre que la prise en compte de la hiérarchie peut produire des gains significatifs en efficacité. Ce protocole constitue une étape préliminaire de notre travail de conception d'un système de partage de mémoire pair-à-pair à grande échelle. Il tient compte de la hiérarchie de l'architecture sous-jacente des nœuds grâce à deux mécanismes :

- priorité des threads et des nœuds locaux pour l'acquisition des verrous,
- libération partielle des verrous.

Le protocole a été implémenté et évalué sur la plate-forme DSM-PM2 [49], en environnement multithread. Les mesures effectuées montrent un très bon comportement du protocole hiérarchique en termes de performances et de passage à l'échelle.

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'interopérabilité deviennent au moins aussi importantes que la performance.

Depuis le 1er octobre 2002, nous avons mis en place un séminaire interne au projet PARIS, intitulé *Gestion de données pair-à-pair à grande échelle*. Ce séminaire a pour objectif la présentation et la discussion des principaux résultats des recherche publiés sur ce thème. Gabriel Antoniu est responsable de l'organisation de ce séminaire.

7. Contrats industriels

7.1. Projet RNRT VTHD

Participants : Yvon Jégou, Christian Pérez, Thierry Priol.

L'objectif du projet RNRT VTHD est la construction d'un réseau à très haut débit (2,5 Gb/s) reliant plusieurs centres de recherche dont l'INRIA. Le projet PARIS participe activement au sous-projet 5 qui a pour but d'expérimenter le réseau avec des applications dans le domaine du calcul scientifique et de la réalité virtuelle. Notre contribution à ce projet est de montrer qu'on peut exploiter un réseau à très haut débit pour faire communiquer plusieurs applications s'exécutant sur plusieurs grappes de calculateurs interconnectées par VTHD [47]. Nous expérimentons notamment le concept d'objet CORBA parallèle afin d'exploiter la totalité de la bande passante du réseau lors de la communication entre deux applications parallèles encapsulées dans des objets CORBA parallèles. Ainsi, un débit soutenu de 826 Mb/s (103 Mo/s) a été obtenu entre Rennes et Sophia en utilisant 11 machines dans chaque centre, soit un débit moyen de 75 Mb/s (9.43 Mo/s) obtenu des cartes Ethernet 100 Mb/s. Nous menons également des expérimentations avec la MVP MOME qui autorise le couplage d'applications fonctionnant sur deux instances de la MVP MOME placées sur deux grappes distinctes. Ces expériences de couplage montrent qu'il est possible d'exploiter la totalité de la bande passante offerte par le réseau VTHD, même lorsque les schémas de communication deviennent irréguliers. La présence de la MVP facilite l'équilibrage de la charge de transfert sur l'ensemble des machines des grappes.

Les expériences menées par le projet montrent qu'il était réaliste de coupler sur une longue distance des applications parallèles s'exécutant sur des grappes de PC même lorsque le débit potentiel de chaque calculateur est limité comme c'est fréquemment le cas pour les grappes. Ces expériences montrent également qu'il est possible d'exploiter simultanément une puissance de calcul élevée pour les besoins des simulations et un débit en communication élevé pour les besoins du couplage.

Le projet PARIS a participé, avec les autres partenaires du projet (France-Télécom, l'ENST, L'ENSTBr, l'INT et EURECOM), à plusieurs campagnes de mesure sur le comportement du réseau expérimental et des applications face à des situations de congestion et en présence de divers modèles de QOS. Ces campagnes ont montré que les techniques de couplage de code mises en œuvre par le projet pouvaient exploiter la totalité de la bande passante disponible sur un réseau non congestionné ; mais la faible capacité d'adaptation au comportement effectif du réseau ne permettaient pas d'obtenir un niveau de performance acceptable en situation de congestion. Cette sensibilité à la qualité de service du réseau est probablement liée au nombre élevé de flux de données mis en œuvre dans le couplage de codes sur grappes. D'autres campagnes de mesure sont prévues dans l'avenir.

7.2. Projet RNRT VTHD++

Participants : Yvon Jégou, Christian Pérez, Thierry Priol.

Le projet RNRT VTHD++ lancé au printemps 2002 fait suite au projet RNRT VTHD. La contribution du projet PARIS dans le premier projet a été de montrer qu'il était possible de faire communiquer des applications parallèles s'exécutant sur des grappes de calculateurs interconnectés par un réseau haut débit comme VTHD. Mais les techniques mises en œuvre ont finalement été validées dans un contexte assez favorable : nombre peu élevé mais suffisant de calculateurs, schémas de communication simples, absence de congestion sur le réseau, etc...

Dans le projet VTHD++, il s'agit pour le projet PARIS de montrer que les techniques de couplage de codes utilisées restent performantes dans des contextes moins favorables : nombre élevé de calculateurs, déséquilibre dans la répartition des données à transférer sur les mémoires, grappes plus hétérogènes, congestion sur le réseau, QOS, contrôle d'admission, etc... Les mises en œuvre des couches basses de communication devront évoluer pour tenir compte du comportement dynamique du réseau. Pour le projet VTHD++, le débit potentiel du routeur d'accès de l'IRISA a été porté de 1 Gb/s à 2.5 Gb/s, débit de la dorsale du réseau. Cette augmentation

des capacités d'échange permettra d'évaluer plus finement le comportement des techniques de couplage en présence de congestion.

7.3. Projet RNTL e-Toile

Participants : Gabriel Antoniu, Yvon Jégou.

Le projet RNTL e-Toile a pour objectif la mise en place d'une infrastructure de type grille de calcul à l'échelle nationale. Ce projet se déroule dans le cadre du Réseau National de recherche et d'innovation en Technologies Logicielles (RNTL) mis en place par les ministères chargés de la Recherche et de l'Industrie et réunit des partenaires de recherche (le CNRS et l'INRIA), des partenaires industriels (France Télécom R&D, CS-SI, Sun France) et des utilisateurs (CEA, EDF). Le projet a démarré le 1er janvier 2002 pour une durée de 2 ans.

Le projet PARIS s'est engagé dans le cadre du projet RNTL e-Toile sur l'axe *répertoire de données à très grande échelle*. L'idée est de déployer une MVP *persistante* sur la grille : les applications pourront se connecter à la DSM préexistante et accéder à des données qui y sont stockées. Une version évoluée de la MVP Mome sera utilisée à cette effet. Les procédures d'initialisation et de déploiement de Mome seront adaptées en conséquence. En 2002, nous avons contribué à la spécification de l'architecture de la grille e-Toile et nous avons finalisé la conception de la MVP persistante. Nous avons également entrepris l'installation de l'environnement Globus sur la grappe de PC.

7.4. Projet RNTL CasPer

Participants : Jean-Louis Pazat, Guillaume Mornet.

Le développement de la simulation numérique permet d'appréhender le comportement des produits futurs dès les phases de conception. En réduisant les cycles et les coûts de conception, ainsi que le nombre de prototypes physiques, le prototypage virtuel sert la compétitivité de secteurs fortement concurrentiels, notamment le domaine des transports. la pleine utilisation de la simulation numérique dans les entreprises se heurte à deux écueils :

- Dans les petites et moyennes entreprises, la simulation numérique reste faiblement utilisée car les coûts d'investissement tant matériels que logiciels restent élevés surtout dès que l'on considère l'utilisation de logiciels reconnus comme étant des standards dans leur domaine d'application. Quand ils sont faits, ces investissements sont souvent longs à rentabiliser. De plus, ces entreprises ne peuvent souvent pas dédier une personne spécifiquement pour gérer ces ressources et préfèrent donc souvent limiter l'utilisation de la simulation numérique.
- Pour les grandes sociétés, la situation est différente car la simulation est en général largement utilisée. Malgré tout la diffusion des résultats se restreint, le plus souvent, à un support papier empêchant toute analyse constructive entre les demandeurs de la simulation et les services l'ayant effectivement réalisée. En outre, les moyens de simulation sont souvent dimensionnés pour pouvoir effectuer les simulations les plus courantes ce qui empêche la simulation ponctuelle de phénomènes dont la complexité est de grande ampleur.

L'objectif du projet RNTL CasPer est de répondre a ces besoins en proposant le développement d'une plate-forme logicielle qui prendra la forme d'un ASP (*Application Service Provider*) accessible à travers l'intranet (et Internet) et qui permette à la fois d'utiliser des ressources déportées pour effectuer une simulation numérique et de partager les informations issues de ces simulations entre différents intervenants situés soit dans la même entreprise (intranet) soit dans des sociétés coopérant sur un même projet (extranet/Internet).

Le projet PARIS a pour rôle la définition complète de l'architecture logicielle de CasPer ainsi que l'évaluation et le choix des technologies pour la mettre en œuvre. Une très grande importance est accordée à cette architecture en raison du choix du modèle Open Source pour mutualiser la maintenance et l'évolution fonctionnelle de CasPer et du besoin d'intégration avec des codes de simulations existants.

Nous assurerons également le suivi du développement du logiciel et l'aide au déploiement des applications sur la plate-forme.

Ce projet de nature précompétitive met a profit notre expertise dans le domaine du GRID et des technologies sous-jacentes, y compris de celles développées dans notre projet mais nous permet également de mieux confronter les technologies existantes aux besoins industriels. Ceci nous permet de mieux préciser certaines pistes de recherches, voire d'en découvrir de nouvelles en particulier autour des problèmes de gestion de ressources et de gestion de données.

7.5. Projet IST POP

Participants : Yvon Jégou, Christian Pérez.

Le projet ISP POP numéro IST-2000-29245 porte sur la portabilité des performances des applications écrites en OpenMP. C'est un projet de trois ans qui a débuté le premier décembre 2001. Les partenaires du projet sont le *European Center of Parallelism of Barcelona* (CEPBA-UPC, Barcelone, Espagne), l'*Istituo di Cibernitica* (IC-CNR, Naples, Italie), le *High Performance Information System Laboratory* (LHPCA-UP, Patras, Grèce) et l'INRIA.

Le langage OpenMP est en phase d'adoption par l'industrie comme un standard de programmation de mémoire partagée. Cependant, ce standard n'est actuellement disponible que sur des machines à mémoire partagée. L'objectif du projet POP est de construite est environnement qui, à partir d'un programme OpenMP, soit capable de générer du code efficace sur tout type d'architecture. Ainsi, on vise à diminuer le besoin d'utiliser un modèle de programmation différent pour chaque architecture. Les architectures visées sont les machines à mémoire partagée, les machines multithreadées ainsi que les grappes de machines.

Plus particulièrement, le projet se focalise sur :

- Les extensions d'expressivité à apporter à OpenMP.
- L'ajout de fonctionnalités pour permettre l'adaptation de l'exécution en fonction du comportement de l'application.
- Les modifications architecturales à apporter à la sémantique fournie par les environnements de mémoire virtuellement répartie ainsi que par les architectures multithreadées.

Ce projet se base sur les résultats du projet européen *Nanos* qui avait permis le développement d'un environnent d'exécution de programme OpenMP pour les machines à mémoire partagée Origin2000.

La contribution du projet Paris est de contribuer à la remise en cause de la sémantique des environnements existant de mémoire virtuellement partagée afin de fournir le support nécessaire à l'exécution efficace de programme OpenMP sur des grappes de machines. Le projet implémentera les propositions dans la MVP MOME développée dans le projet Paris.

Durant l'année 2002, nous avons contribué à l'analyse des limitations des MVP existantes ainsi qu'à la définition de fonctionnalités importantes des MVP afin de satisfaire aux objectifs du projet POP. Nous avons également commencé à incorporer ces fonctionnalités dans la MVP MOME. Un premier prototype de l'exécutif Nanos a été mis en œuvre sur la MVP MOME.

7.6. Network of Excellence CoreGRID

Les technologies de grille et de calcul pair-à-pair sont considérées comme les principes de base des futures infrastructures de calcul haute performance. Celles-ci seront constituées d'un ensemble de ressources largement distribuées sur Internet. Cependant, la programmation de telles infrastructures sera certainement très difficile. De nouveaux paradigmes de programmation, environnements logiciels, algorithmes et applications seront nécessaires. Ce réseau d'excellence a pour objectif de définir un programme coordonné de recherche entre les différentes équipes européennes actives dans ce domaine, avec une vision à moyen et long terme de la recherche.

La proposition de réseau est pilotée par l'ERCIM. Le responsable général en est Michel Cosnard (UR INRIA Sophia-Antipolis), et le responsable exécutif T. Priol. La proposition a été soumise en juin 2002 comme *Expression of Interest*.

Les partenaires du réseau comprennent la plupart des institutions européennes les plus actives dans le domaine du calcul sur grille, et plusieurs partenaires industriels ont exprimé leur intérêt.

7.7. Contrat Alcatel

Participants : Yvon Jégou, Christine Morin.

La participation du projet dans le contrat INRIA-Alcatel numéro 101C01010031329012 porte sur l'utilisation d'une mémoire virtuellement partagée dans les routeurs de l'Internet afin de faire face à l'accroissement du volume des tables de routage et de permettre la parallélisation des codes de routage, [44][45]. Pour respecter les contraintes de tolérance aux fautes, nous avons, dans le cadre de cette collaboration, intégré au système de mémoire virtuelle partagée Mome un support de points de reprise fondé sur la gestion de copies multiples des pages de données sur l'ensemble du système.

7.8. Contrats EDF

Participants : Renaud Lottiaux, Christine Morin, Geoffroy Vallée.

L'objectif de la convention de recherche avec EDF (contrat n° 101C01700031329011) est de concevoir et réaliser un environnement et des outils pour la gestion et l'utilisation de grappes de PC pour l'exécution d'applications à haute performance. Les travaux effectués dans ce contexte s'intègrent à l'activité de recherche Gobelins et portent plus particulièrement sur la gestion globale des ressources processeur dans une grappe de calculateurs. Il s'agit d'une part de définir des politiques d'ordonnancement global de processus sur une grappe et de les mettre en œuvre dans le système Gobelins. D'autre part, nous étudions les techniques de reprise d'applications parallèles visant à décharger les programmeurs de la gestion des défaillances.

EDF a également co-financé le post-doctorat industriel (PDI INRIA) effectué par Renaud Lottiaux pendant toute l'année 2002 (contrat). L'objectif du post-doctorat est d'une part d'améliorer la robustesse du prototype du système Gobelins afin de permettre l'exécution d'applications de grande envergure. Il s'agit d'autre part de compléter les fonctionnalités du système Gobelins par un mécanisme de pagination en mémoire distante (pour améliorer les performances du système pour les applications gourmandes en mémoire), par un mécanisme d'allocation dynamique de mémoire (pour ne pas limiter le spectre des applications pouvant s'exécuter sur le système Gobelins) et par un système de gestion de fichiers distribué exploitant le système de caches coopératifs offert par les conteneurs.

Le système Gobelins a été installé sur une grappe expérimentale dans les locaux d'EDF R&D à Clamart. Des expérimentations ont été effectuées avec des applications d'envergure fournies par EDF.

8. Actions régionales, nationales et internationales

8.1. Actions régionales

Le projet s'est vu attribuer une subvention du Conseil Régional de Bretagne pour l'extension de la grappe de PC. La bourse de thèse d'A. Ribes fait également l'objet d'un contrat avec le Conseil Régional de Bretagne (financement à 50 %).

8.2. Actions nationales

8.2.1. ACI GRID RMI

Participants : Alexandre Denis, Yvon Jégou, Jean-Louis Pazat, Christian Pérez, Thierry Priol, André Ribes.

L'objectif de cette action est de promouvoir un modèle de programmation pour les grilles de calcul combinant à la fois des modèles du calcul parallèle et du calcul distribué. Ce modèle s'appuie sur le concept d'objet

distribué et de composant logiciel pour la programmation répartie. L'action vise à concevoir et expérimenter PADICOTM, une infrastructure logicielle de communication haute performance, permettant à la fois la communication efficace entre objets ou composants ainsi que la programmation parallèle. Cette action, d'une durée de deux ans, est coordonnée par le projet PARIS (C. Pérez).

8.2.2. ACI GRID HydroGrid

Participants : Christian Pérez, André Ribes.

L'action *HydroGrid* a pour but de modéliser et simuler des transferts de fluides et de transport de solutés dans des milieux géologiques souterrains. C'est un projet pluridisciplinaire de trois ans qui a débuté en septembre 2002. Cette action se propose d'utiliser les grilles de calcul en se servant notamment des résultats de l'ACI GRID RMI. Notre contribution est d'apporter l'expertise des grilles de calcul ainsi que les outils tels que les objets CORBA parallèles et l'environnement d'exécution PADICOTM.

L'un des points forts du projet HydroGrid est de regrouper des équipes spécialisées dans les domaines applicatifs, des équipes de numériciens, et des équipes d'informaticiens. Les partenaires de l'action sont l'équipe *Hydrodynamique et transferts en milieux poreux* de l'IMFS Strasbourg, l'équipe *Transferts physiques et chimiques en géosciences* de Rennes, les projets Aladin, Estime et PARIS de l'INRIA et l'équipe ASCII de l'IRISA.

8.2.3. ACI GRID DataGRAAL

Participants : Gabriel Antoniu, Luc Bougé, Sébastien Lacour, Thierry Priol.

L'objectif de cette action est de comprendre le lien existant entre d'une part le calcul sur grille et d'autre part les systèmes d'information distribués. Le cadre dans lequel cette question est considérée est celui du passage à la vraiment très grande échelle, qui représente un défi ardent, à la fois pour le calcul sur grille et pour les systèmes d'information distribués. Cette proposition unifie les propositions d'ACI AnimaGrid et VTGE qui sont respectivement des actions de type animation et projet logiciel. Les motivations pour la proposition de cette action sont donc doubles.

Animation : Soutenir les travaux autour des problématiques pair-à-pair et gestion de données en faisant se rencontrer deux communautés, bases de données et système. Servir de lien entre les différents projets français dans le domaine des grilles de données.

Pépinière logicielle : Permettre de mûrir des projets logiciels autour de la gestion des données dans les grilles à grande échelle par réalisation de maquettes préliminaires.

Cette action a une durée de deux ans et se déroule en collaboration avec les partenaires suivants : équipe *Clusters et grille* (LRI), projet ReMaP (LIP), équipe SRC (LIP6), équipe *Systèmes d'Information Communicants* (LISI, INSA de Lyon), projet APACHE (ID-IMAG), projet Caravel (INRIA Rocquencourt).

8.2.4. ACI GRID GRID2

Participants : Jean-Louis Pazat, Christian Pérez.

Jean-Louis Pazat est le responsable de ce projet qui regroupe 10 laboratoires et est financé par le ministère de la recherche pour 3 ans à hauteur de 150 000 Euros.

Cette ACI vise à assurer la formation de jeunes chercheurs, la diffusion d'information et plus généralement les rencontres entre les chercheurs impliqués dans des travaux sur les grilles de calcul. Les principaux domaines abordés sont les suivants.

Architecture des logiciels et langages : architecture des systèmes distribués à grande échelle, gestion globale et dynamique de ressources, langages, expression pour les grilles ;

Supports d'exécution et *middleware* : communications et *middleware*, observation et analyse de performances ;

Modèles et algorithmique : ordonnancement et algorithmique ;

Algorithmique et applications hautes performances : manipulation de données intensives, algorithmique et Simulation intensive, aspects applicatifs généraux et interdisciplinaires.

L'école GRID 2002 qui a lieu à Aussois au mois de décembre 2002 a été organisée dans le cadre de ce projet pour faciliter l'intégration des jeunes chercheurs dans cette communauté en leur donnant les éléments d'une culture commune. De plus chaque groupe thématique organise des réunions thématiques portant sur des domaines scientifiques et techniques pointus visant à une bonne diffusion des connaissances entre les chercheurs. Une autre école sera organisée durant ce projet. Enfin deux conférences seront également organisées afin d'assurer une dissémination rapide des résultats des travaux dans la communauté scientifique.

8.3. Relations bilatérales internationales

8.3.1. Europe

Univ. nouvelle de Lisbonne, Portugal. J.-L. Pazat est responsable d'une action de coopération INRIA/ICCTI avec le département informatique de la faculté des sciences et de technologie de l'Université nouvelle de Lisbonne. Le responsable du coté Portugal est le Prof. Dr. José C. Cunha. Le projet est de définir une infrastructure qui permette la description d'une application de simulation numérique répartie et son adaptation (reconfiguration) dynamique pour des problèmes de régulation de charge, de tolérance aux fautes ainsi que sa spécialisation pour des applications spécifiques de simulation. Cet environnement est fondé sur trois concepts : l'encapsulation de services à l'intérieur de composants auto-adaptables, le monitoring du système et la coordination globale de composants au travers d'un framework.

Microsoft Research Center, Cambridge, UK. Le projet PARIS collabore avec le centre de recherche Microsoft de Cambridge (UK). Cette collaboration porte sur la conception de mécanismes de tolérance aux fautes pour l'exécution d'applications parallèles sur une grappe de calculateurs.

Univ. Passau, Allemagne. Le Prof. Christian Lengauer, de l'Université de Passau, Allemagne, a visité le projet PARIS en janvier 2002.

8.3.2. Amérique du nord

Équipe associée à l'étranger Hyperion, Univ. New Hampshire, USA. Nous avons obtenu, en partenariat avec le projet ReMaP de l'INRIA Rhône-Alpes, un soutien *Équipe associée à l'étranger* à la suite de l'appel à projets de l'été 2001. Il concerne notre coopération avec l'équipe de Phil Hatcher et Bob Russell, professeurs à l'*Université du New Hampshire*, Durham, NH, USA, où ils animent le groupe de recherche *Compilation parallèle*. Cette collaboration a déjà été soutenue par deux contrats NSF/INRIA successifs, respectivement en 1998 et en 2001. En ce qui concerne le projet PARIS, l'objet de la collaboration est essentiellement le projet *Hyperion*, un environnement de compilation Java au-dessus de DSM-PM2 pour les grappes de PC hautes performances. Cette ligne de recherche était déjà bien engagée grâce à la collaboration étroite entre Phil Hatcher et Gabriel Antoniu dans le cadre de son travail de doctorat. Une partie de sa thèse y est consacrée. Un premier prototype est maintenant opérationnel, et plusieurs publications communes ont été écrites. Dans le cadre de cette collaboration, Philip Hatcher a effectué un séjour de deux semaines au sein du projet PARIS en mai 2002, soutenu par le programme *Hyperion*.

Joel Daniels, étudiant de Phil Hatcher à UNH, a effectué un stage de 2 mois dans le projet PARIS sous la direction de C. Pérez. Ce stage a été en partie soutenu par un bourse accordée par le *International Research Opportunities Program* (IROP) de UNH (http://www.unh.edu/urop/discoverirop.html), l'autre partie étant prise en charge par l'équipe associée.

Univ. Maryland, USA. Le projet entretient des relations suivies avec Liviu Iftode de Maryland University sur le thème de la conception de systèmes d'exploitation pour les grappes de calculateurs.

Le projet a invité Liviu Iftode le 10 juin 2002. Un projet NSF portant sur la migration de flux de données est en cours de dépôt.

Visite : Univ. Wisconsin, Madison, USA. Le Prof. Miron Livny, directeur du projet Condor de l'Université du Wisconsin, Madison, USA, a visité le projet PARIS en novembre 2002, dans le cadre d'une tournée des principales équipes françaises actives dans le domaine du calcul distribué haute performance.

8.3.3. Moyen-Orient, Asie, Océanie

Indian Institute of Technology, Kharagpur. Le projet accueille depuis le 10 mai 2002 pour une durée d'un an Ramamurthy Badrinath, enseignant-chercheur dans le département d'informatique de l'*Indian Institute of Technology* de Kharagpur (financement du MAE pour les 6 premiers mois et de l'INRIA pour les 6 derniers mois). Ses travaux au sein du projet PARIS s'inscrivent dans le cadre de l'activité de recherche Gobelins et portent sur les protocoles de sauvegarde et de restauration de points de reprise d'applications parallèles.

Deakin University, Australie. Le projet PARIS entretient des relations suivies avec l'équipe du Professeur Andrzej Goscinski de Deakin University (Australie) qui mène des travaux de recherche sur la gestion des ressources dans les grappes de calculateurs. Une demande de financement d'un projet conjoint sur la haute disponibilité des grappes de calculateurs a été déposée en juillet 2002. Andrzej Goscinski a visité le projet PARIS pendant 2 jours en août 2002.

Seoul National University, Corée. Le projet PARIS a accueilli les Prof. Seung-Jo Kim et le Prof. Woo-Il Lee du *School of Mechanical Engineering and Aerospace Engineering* de l'Univ. nationale de Séoul, Corée, en décembre 2001. Cette visite a été effectuée en coordination entre l'IRISA et l'Antenne Bretagne de l'ENS Cachan. Elle a été financée par l'association *Ariel* (http://www.ariel.asso.fr/) qui soutient les relations internationales de la Conférence des grandes écoles dans le domaine des Sciences de l'ingénieur.

Un projet de coopération a été soumis en octobre 2002 à l'appel d'offre du CNRS. Il a pour objectif de créer un infrastructure de grille informatique franco-coréenne.

Université libanaise : Christine Morin co-encadre la thèse d'Ahmad Faour dans le cadre d'une convention de co-direction de thèse entre l'Université de Rennes 1 et l'Université libanaise. Les travaux d'Ahmad Faour portent sur la conception et la mise en œuvre de l'architecture de serveur Web *Weblins* sur grappe qui exploite les fonctionnalités du système Gobelins tout en spécialisant les algorithmes de gestion globale des ressources. En 2002, un simulateur a été réalisé pour comparer les algorithmes de gestion des données en mémoire et sur disque proposés pour l'architecture Weblins avec ceux de la littérature.

Accueil de stagiaires indiens. Le projet PARIS a accueilli 2 stagiaires indiens du IIT Kharagpur dans le cadre du programme *INRIA International Internship*. Parikshit Machwe a travaillé avec G. Antoniu et Nitin Jain avec C. Pérez. Leur séjour a duré 2 mois, entre mai et juillet 2002, financé pour moitié par la Direction des relations internationales de l'INRIA.

Visite : Technion, Haifa, Israël. Le Prof. Assaf Schuster a effectué un séjour d'une semaine dans le projet PARIS en septembre 2002. Il a travaillé avec L. Bougé et G. Antoniu sur les protocoles de cohérences pour DSM.

9. Diffusion des résultats

9.1. Animation de la communauté scientifique

9.1.1. Responsabilité d'animation

GDR CNRS ARP. L. Bougé dirige depuis 2000 le Groupement de recherche (GDR) CNRS *Architecture, réseaux et systèmes, parallélisme* (ARP, http://www.arp.cnrs.fr/). Ce GDR est rattaché au

département STIC. Il constitue l'un des 6 *GDR d'animation* du département. Comme les autres GDR d'animation, ARP vient d'être renouvelé au 1er janvier 2002.
J.-L. Pazat anime au sein de ce GDR le groupe de travail *Grappes* dont les activités sont centrées sur l'utilisation des réseaux de stations de travail pour le calcul haute performance.

Coordination inter-GDR STIC. L. Bougé assure depuis 2001 la coordination (informelle !) des 6 GDR d'animation du département STIC du CNRS, à la suite de Malik Ghallab. Il a en particulier coordonné les discussions avec la direction du département concernant le renouvellement de ces GDR, leur financement et l'articulation entre les GDR et les RTP.

ACI GRID du Ministère de la recherche. L. Bougé et T. Priol sont membres du Conseil scientifique de l'ACI GRID, lancée en avril 2001 (http://www.recherche.gouv.fr/recherche/aci/grid.htm). Cette ACI, consacrée à la *globalisation des ressources informatiques et des données* est dirigée par Michel Cosnard, INRIA Sophia.

Conférence Euro-Par. L. Bougé est vice-président du *Steering Committee* de la conférence annuelle *Euro-Par* (environ 250 participants, http://www.europar.org/).

Conférence RenPar. J.-L. Pazat est président du comité de pilotage de la conférence RenPar, les *Rencontres francophone du parallélisme* (http://www.renpar.org/). RenPar en est à sa 14e édition.

Conférence IPDPS. L. Bougé est membre du *Steering Committee* de *International Parallel and Distributed Processing Symposium* (IPDPS). Avec Michel Cosnard, ils organisent la prochaine édition de la conférence à Nice en avril 2003, en appui sur l'UR INRIA de Sophia-Antipolis, l'UMR CNRS I3S et l'Université de Nice (http://www.ipdps.org/). Environ 350 personnes sont attendues.

ACI GRID GRID2. J.-L. Pazat est responsable du projet d'animation *GRID2 : Groupe de rencontres, d'information et de discussion sur la globalisation des ressources informatiques et des données* sélectionné lors de l'appel à propositions 2001 de l'ACI GRID (http://www.irisa.fr/grid2/).
C. Pérez est coordonnateur du sous-thème *Communications et middleware* du Thème 2 de l'ACI d'animation GRID2.

ACI GRID RMI. C. Pérez est responsable du projet logiciel *RMI : Objets distribués haute performance pour la grille de calcul* sélectionné lors de l'appel à propositions 2001 de l'ACI GRID (http://www.irisa.fr/Grid-RMI/).

Réseau d'excellence européen CoreGRID. T. Priol est *Deputy Director* de la proposition CoreGRID (http://www.irisa.fr/CoreGRID/) à l'appel à expression d'intérêt lancé en juin 2002 par l'Union européenne, dans la catégorie *Network of Excellence*. Cette proposition est dirigée par Michel Cosnard, de l'UR de Sophia-Antipolis.

Projet européen IST POP. C. Pérez est correspondant scientifique de l'INRIA pour le projet européen IST POP lancé au 1er décembre 2001.

RTP STIC CNRS. L. Bougé est membre du Comité de pilotage du Réseau thématique pluridisciplinaire (RTP) intitulé *Calcul à hautes performances et calcul réparti* dirigé par Yves Robert (Projet ReMaP) et B. Plateau (Projet Apache).

9.1.2. Comités de rédaction, de pilotage et de programme

C. Morin a été membre des comités de programme suivants.

- *International Workshop on Distributed Shared Memory on Clusters* (DSM2002) organisé dans le cadre de la conférence *IEEE International Symposium on Cluster Computing and the Grid* (CCGrid'2002) qui a eu lieu en mai 2002.
- *International Workshop on Caching, Coherence and Consistency* (WC3 '02) organisé dans le cadre de la conférence *ACM International Conference on Supercomputing* qui a eu lieu en juin 2002.
- *5th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP)* qui a eu lieu en octobre 2002 à Pékin (Chine).

- *International Workshop on Distributed Shared Memory on Clusters* (DSM2003) organisé dans le cadre de la conférence *IEEE International Symposium on Cluster Computing and the Grid (CCGrid'2003)* qui aura lieu en mai 2003.
- *International Conference on Distributed Computing Systems* (ICDCS 2003) qui aura lieu en mai 2003.
- *International Symposium on Object-oriented Real-time Distributed Computing* (ISORC 2003) qui aura lieu en mai 2003.

T. Priol T. Priol est le co-président du comité de programme du thème *Grid Computing and Middleware Systems* de la conférence Euro-Par'03 qui se tiendra en Août 2003 à Klagenfurt (Autriche).
T. Priol est membre du comité de lecture de la revue *Parallel Computing*.
Il a été membre du comité de lecture d'un numéro spécial *Computing : Computational Grids* du journal *Journal of Parallel Distributed Computing* en 2002.
Il a été membre des comités de programme suivants.

- *4th Eurographics Workshops on Parallel Graphics and Visualisation* qui a eu lieu en septembre 2002 à Blaubeuren (Allemagne).
- *5th International Conference on High Performance Computing and Computational Science* (VECPAR'02) qui a eu lieu en juin 2002.
- *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid* (CCGRID'02) qui a eu lieu en mai 2002 à Berlin (Allemagne).
- *Workshop on Advanced Collaborative Environment* (WACE02) qui s'est tenu en juillet 2002 à Edinbourg.
- *11th Euromicro Conference on Parallel, Distributed and Network-based Processing* qui se tiendra en à Gênes en février 2003.

J.-L. Pazat a été membre du comité de programme du workshop *Java in High Performance Computing* à HPCN 2002.

L. Bougé est membre du *Editorial Advisory Board* du journal *Scientific Programming*, édité par IOS Press.
Il a été membre des comités de programme suivants.

- *7th International Workshop on High-Level Parallel Programming Models and Supportive Environments* (HIPS 2002), workshop satellite de la conférence IPDPS 2002, Fort Lauderdale, Floride, avril 2002.
- *8th International Workshop on High-Level Parallel Programming Models and Supportive Environments* (HIPS 2003), workshop satellite de la conférence IPDPS 2003, Nice avril 2003.
- *11th International Conference on Parallel Architectures and Compilation Techniques* (PACT 2002), Charlottesville, Virginie, september 2002.
- *IEEE 4th International Conference on Cluster Computing* (Cluster 2002), Chicago, septembre 2002.
- *14^e Rencontres francophones du parallélisme*, Hamamet, avril 2002.

9.1.3. Comités d'évaluation et expertises

T. Priol a été invité à participer à un groupe d'experts auprès de la Commission européenne pour aider à la définition du FP6 dans le domaine des grilles de calcul (ce groupe s'est réuni en octobre 2002) et à coordonner les programmes internationaux dans ce domaine.
Il est évaluateur auprès de la commission européenne des projets IST JPD, DEBUT, BIOWULF, DAMIEN et EUROGRID.

L. Bougé a été expert pour la sélection des projets RNTL de l'appel à proposition de février 2002.
Il a été sollicité pour des rapports scientifiques sur des candidatures à l'*appellation de Maître de conférence* à l'INT et à l'ENST Bretagne.
Il a participé au jury d'attribution des *primes d'encadrement doctoral et de recherche* (PEDR) du MENRT en septembre 2002.
Il est membre du *Conseil scientifique* de l'*École doctorale Informatique, information et société* (EDISS) de Lyon.

9.2. Enseignement universitaire

C. Morin est responsable du module de *programmation des grappes de calculateurs pour le calcul haute performance* (PGC) du DEA informatique de l'Université de Rennes 1.
Elle a donné un cours sur les *systèmes distribués* dans le cadre du DEA calcul intensif de l'Université libanaise à Beyrouth au Liban en janvier 2002.
Elle a donné un cours sur les grappes de calculateurs dans le cadre de l'option de 3e année *architecte de services en réseaux* à l'Institut national des télécommunications (INT) d'Evry en novembre 2002.

T. Priol intervient dans le module de *programmation des grappes de calculateurs pour le calcul haute performance* (PGC) du DEA d'Informatique de l'Université de Rennes 1.

J.-L. Pazat et C. Pérez ont mis en place une option de 5e année à l'INSA de Rennes sur le thème *objets et composants pour la programmation répartie*.

G. Antoniu est intervenu dans le module *Méthodes de programmation parallèle* en 4e année de l'INSA de Rennes au deuxième semestre 2001-2002, dont Jean-Louis Pazat est responsable. Il intervient dans le cadre du module *Système* (SY) en IUP 2 MIAGE, IFSIC, au premier semestre 2002-2003.

L. Bougé est responsable du *Magistère Informatique et télécommunications* (le MIT Rennes !), cohabilité par l'ENS Cachan et l'Université Rennes 1, qui a ouvert à la rentrée 2002. Le coresponsable est Olivier Ridoux, professeur à l'IFSIC, du projet Lande de l'IRISA.
Il a servi comme examinateur pour la session 2002 du concours *Informatique* commun à l'ENS Ulm et l'ENS Lyon. Il participe à la procédure d'adoption de ce concours par l'ENS Cachan pour la session 2003.

9.3. Participation à des colloques, séminaires, invitations

Outre les conférences et workshops donnant lieu à publication des actes listés dans la bibliographie, les membres du projet PARIS ont présenté leurs travaux dans plusieurs séminaires ou workshops.

Journée IrisaTech *Peer-to-Peer*. L. Bougé et T. Priol ont organisé une session de la série IrisaTech en février 2002, consacrée aux approches *pair-à-pair*. Cette session a réuni une trentaine de participants. Les exposés ont été présentés par L. Bougé, Franck Cappello (LRI, Orsay) et Quentin Gallet (Société *La deuxième tête*, Rennes Atalante).

Matinales de Rennes Atalante. L. Bougé a été invité a faire un exposé sur le *calcul pair-à-pair* dans le cadre des *Matinales de Rennes Atalante* consacrée à ce sujet en mai 2002. Cette réunion rassemblait une centaine de participants.

CEA Saclay. C. Pérez et T. Priol ont participé à une journée sur le couplage de code et le parallélisme au CEA Saclay. C. Pérez a présenté un exposé intitulé *Objets et composants parallèles pour les grilles de calcul*.

Stand INRIA à SuperComputing 2002. Le projet PARIS est bien représenté sur le stand INRIA à SuperComputing 2002 en novembre 2002. C. Pérez y présente les travaux autour des objets CORBA parallèles et de PADICOTM ; Renaud Lottiaux et Pascal Gallard y présentent un prototype du système Gobelins.

Journée *Grilles informatiques* du MENRT. T. Priol a été invité à donné un exposé sur les défis et les perspectives des grilles informatiques dans le cadre de la journée de réflexion sur le thème : *Les grilles informatiques : une nouvelle révolution ?* organisée au MENRT en mars 2002.

Conférence invitée HIPS 2002. T. Priol a été *keynote speaker* du *7th International Workshop on High-Level Parallel Programming Models and Supportive Environments* (HIPS 2002) qui s'est tenu en avril 2002 à Fort Landerdale, FLoride.

Séminaire à Dagstuhl. T. Priol a été invité à présenter ses travaux dans le domaine de la programmation des grilles de calcul lors d'un séminaire Dagstuhl intitulé *Performance Analysis and Distributed Computing* en août 2002.

CINES. T. Priol a été invité à présenter ses travaux sur les grilles de calcul dans le cadre de la journée visualisation et grands calculs organisée par le CINES en octobre 2002.

Séminaire X-Aristote. T. Priol a organisé avec Philippe d'Anfray une journée de séminaire consacrée *Grid Computing et simulation numérique* en avril 2002. Il y a présenté l'ACI GRID.

Congrès annuel de l'APMEP. L. Bougé et Y. Jégou ont animé un atelier sur *Science et Technologie de l'Information et de la Communication : grappes de calculateurs et grilles de calculs* lors du congrès annuel de l'*Association des professeurs de mathématiques de l'enseignement public* à Rennes en octobre 2002.

Conférence des grandes écoles de Bretagne. L. Bougé et C. Morin ont présenté les activités de recherche du projet PARIS lors de la réunion annuelle organisée par l'IFSIC à Rennes en avril.

Assemblée générale de Specif. L. Bougé a été invité à une table ronde sur *la place de l'informatique dans les STIC* lors de l'assemblée générale annuelle de Specif à Grenoble en janvier 2002.

Club des utilisateurs du calcul parallèle. C. Pérez est intervenu dans les journées du 17 et 18 juillet 2002 du *Club des utilisateurs du calcul parallèle* à l'UR Sophia Antipolis. Il y a présenté l'ACI GRID RMI.

Autres présentations. G. Antoniu a fait une présentation intitulée *Partage de mémoire à très grande échelle sur des réseaux pair-à-pair* à la journée organisée par l'ACI GRID2 lors de la conférence RenPar 2002 (Hammamet, avril 2002). Il fera une présentation intitulée *Partage de mémoire sur une infrastructure pair-à-pair* à l'École Grid 2002 (Aussois, décembre 2002).

C. Morin a présenté l'activité de recherche *Gobelins* le 12 novembre 2002, dans le cadre du séminaire du département informatique et télécommunication de l'ENS Cachan, antenne de Bretagne.

9.4. Responsabilités

J.-P. Banâtre est en charge des relations européennes à la Direction des relations internationales (DRI) de l'INRIA.

L. Bougé est le directeur du département Informatique et télécommunications (DIT) à l'*Antenne Bretagne* de l'ENS Cachan sur le Campus de Ker Lann, à Bruz.

C. Morin est membre élue de la Commission d'évaluation de l'INRIA depuis juillet 2002.

Elle est présidente de la Commission des utilisateurs des moyens informatiques (CUMI) de l'IRISA depuis octobre 2002.

T. Priol est le vice-président de la Commission d'évaluation de l'INRIA.

9.5. Divers

L. Bougé a participé au jury de recrutement CR2 de l'UR INRIA de Rennes au printemps 2002.

Il représente le projet PARIS au Comité des projets de l'IRISA.

Il est membre suppléant Commission de spécialistes de la 27^{e} section (informatique) de l'ENS Cachan.

C. Morin est membre élue de la commission de spécialistes de la 27e section (informatique) de l'Université de Rennes 1 (jusqu'en septembre 2002 en raison de son changement de statut).
Elle est membre de la commission locale des postes d'accueil (CLPA) de l'IRISA.

J.-L. Pazat est membre nommé de la Commission de spécialistes de la 27e section (informatique) de l'université d'Orsay et membre nommé de la commission de spécialistes de la 27e section (informatique) de l'université de Bretagne Sud.
Jean-Louis Pazat est le responsable pédagogique de la cinquième année informatique de l'INSA de Rennes.
Jean-Louis Pazat est membre élu du Conseil d'administration de l'INSA de Rennes et membre du Conseil d'administration du CRI de l'INSA de Rennes

T. Priol est membre suppléant nommé de la commission de spécialistes de la 27e section (Informatique) de l'université de Rennes 1 depuis septembre 2001.

G. Antoniu est responsable de l'organisation d'un séminaire *Gestion de données à grande échelle*, organisé au sein du projet PARIS.

10. Bibliographie

Bibliographie de référence

[1] F. ANDRÉ, M. LE FUR, Y. MAHÉO, J.-L. PAZAT. *The Pandore Data Parallel Compiler and its Portable Runtime.* in « High-Performance Computing and Networking (HPCN Europe 1995) », série Lecture Notes in Computer Science, volume 919, Springer Verlag, pages 176-183, Milan, Italy, mai, 1995.

[2] G. ANTONIU, L. BOUGÉ. *DSM-PM2 : A portable implementation platform for multithreaded DSM consistency protocols.* in « Proc. 6th International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS '01) », série Lect. Notes in Comp. Science, volume 2026, Held in conjunction with IPDPS 2001. IEEE TCPP, Springer-Verlag, pages 55-70, San Francisco, avril, 2001, http://www.inria.fr/rrrt/rr-4108.html, Available as INRIA Research Report RR-4108.

[3] A. DENIS, C. PÉREZ, T. PRIOL. *PadicoTM : An Open Integration Framework for Communication Middleware and Runtimes.* in « IEEE Intl. Symposium on Cluster Computing and the Grid (CCGrid2002) », IEEE Computer Society, pages 144-151, Berlin, Germany, mai, 2002, http://www.inria.fr/rrrt/rr-4554.html, Available as INRIA Reserach Report RR-4554.

[4] A.-M. KERMARREC, C. MORIN, M. BANÂTRE. *Design, Implementation and Evaluation of ICARE.* in « Software Practice and Experience », numéro 9, 1998, pages 981-1010.

[5] T. KIELMANN, P. HATCHER, L. BOUGÉ, HENRI E. BAL. *Enabling Java for High-Performance Computing : Exploiting Distributed Shared Memory and Remote Method Invocation.* in « Communications of the ACM », numéro 10, volume 44, octobre, 2001, pages 110-117, http://www.irisa.fr/paris/biblio/Papers/Bouge/KieHatBouBal01CACM.ps.gz, Special issue on Java for High Performance Computing.

[6] Z. LAHJOMRI, T. PRIOL. *KOAN : A Shared Virtual Memory for iPSC/2 Hypercube.* in « Proc. of the 2nd Joint Int'l Conf. on Vector and Parallel Processing (CONPAR'92) », série Lecture Notes in Computer Science, volume 634, Springer Verlag, pages 441-452, septembre, 1992, http://www.inria.fr/rrrt/rr-1634.html.

[7] T. PRIOL. *Efficient support of MPI-based parallel codes within a CORBA-based software infrastructure.* in « Response to the Aggregated Computing RFI from the OMG, Document orbos/99-07-10 », juillet, 1999.

Articles et chapitres de livre

[8] G. ANTONIU, L. BOUGÉ, P. HATCHER, M. MACBETH, K. MCGUIGAN, R. NAMYST. *The Hyperion system : Compiling multithreaded Java bytecode for distributed execution.* in « Parallel Computing », volume 27, octobre, 2001, pages 1279-1297, http://www.irisa.fr/paris/biblio/Papers/Antoniu/AntBouHatBetGuiNam01ParCo.ps.gz.

[9] O. AUMAGE, L. BOUGÉ, A. DENIS, L. EYRAUD, J.-F. MÉHAUT, G. MERCIER, R. NAMYST, L. PRYLLI. *High Performance Computing on Heterogeneous Clusters with the Madeleine II Communication Library.* in « Cluster Computing », volume 5, 2002, pages 43-54, http://www.irisa.fr/paris/Biblio/Papers/Bouge/AumBouDenEyrMehMerNamPry01CC.ps.gz.

[10] O. AUMAGE, L. BOUGÉ, A. DENIS, L. EYRAUD, R. NAMYST, C. PÉREZ. *Calcul réparti à grande échelle. Métacomputing.* Hermès/Lavoisier, mai, 2002, chapitre Communications efficaces au sein d'une interconnexion hétérogène de grappes, pages 103-128, http://www.irisa.fr/paris/biblio/Papers/Bouge/AumBouDenEyrNamPer01Calculateurs.ps.gz.

[11] P. D'ANFRAY, C. PÉREZ. *Composants CORBA parallèles.* in « MATAPLI », volume 68, mai, 2002, http://www.irisa.fr/paris/Biblio/Papers/Perez/AnfPer02Matapli.ps.

[12] A. DENIS. *CORBA haute performance.* in « Technique et Science Informatiques (TSI) », volume 21, 2002, pages 659-683, http://www.irisa.fr/paris/Biblio/Papers/Denis/Den02TSI.ps.

[13] A. DENIS, C. PÉREZ, T. PRIOL. *Achieving Portable and Efficient Parallel CORBA Objects.* in « Concurrency and Computation : Practice and Experience », 2002, à paraître.

[14] A. DENIS, C. PÉREZ, T. PRIOL. *PadicoTM : An Open Integration Framework for Communication Middleware and Runtimes.* in « Future Generation Computer Systems », 2003, à paraître.

[15] A. DENIS, C. PÉREZ, T. PRIOL, A. RIBES. *Process Coordination and Ubiquitous Computing.* CRC Press, juin, 2002, chapitre Programming the Grid with Distributed Objects, pages 133-148.

[16] A.-M. KERMARREC, C. MORIN. *HA-PSLS : a Highly Available Parallel Single Level Store System.* in « Concurrency and Computation : Practice and Experience », 2002, à paraître.

Communications à des congrès, colloques, etc.

[17] L. BOUGÉ, V. DANJEAN, R. NAMYST. *Improving reactivity to I/O events in multithreaded environments using a uniform, scheduler-centric API.* in « Euro-Par 2002 : Parallel Processing », série Lect. Notes in Computer Science, volume 2400, Springer-Verlag, pages 605-614, Paderborn, Germany, août, 2002, http://www.inria.fr/rrrt/rr-4471.html, Also available as INRIA Research Report RR-4471.

[18] A. DENIS. *PadicoTM : un environnement ouvert pour l'intégration d'exécutifs communicants.* in « 14es Rencontres Francophones du Parallélisme (RenPar'14) », pages 99-106, Hammamet, Tunisie, avril, 2002, http://www.irisa.fr/paris/Biblio/Papers/Denis/Den02RenPar.ps.

[19] A. DENIS, C. PÉREZ, T. PRIOL. *Towards High Performance CORBA and MPI Middlewares for Grid Computing.* in « Proc of the 2nd Intl. Workshop on Grid Computing (GRID 2001) », série Lecture Notes in Computer Science, numéro 2242, Springer-Verlag, éditeurs C. A. LEE., pages 14-25, Denver, Colorado, USA, novembre, 2001, http://www.irisa.fr/paris/Biblio/Papers/Denis/DenPerPri01GC.ps, Held in conjunction with SuperComputing 2001 (SC2001).

[20] A. DENIS, C. PÉREZ, T. PRIOL. *PadicoTM : An Open Integration Framework for Communication Middleware and Runtimes.* in « IEEE Intl. Symposium on Cluster Computing and the Grid (CCGrid2002) », IEEE Computer Society, pages 144-151, Berlin, Germany, mai, 2002, http://www.irisa.fr/paris/Biblio/Papers/Denis/DenPerPri02CCGRID.ps.

[21] P. GALLARD, C. MORIN, R. LOTTIAUX. *Dynamic Resource Management in a Cluster for High-Availability.* in « Euro-Par 2002 : Parallel Processing », série Lecture Notes in Computer Science, volume 2400, Springer-Verlag, pages 588-592, Paderborn, Germany, août, 2002, http://www.irisa.fr/paris/Biblio/Papers/Gallard/GalMorLot02europar.ps.

[22] P. GALLARD, C. MORIN, R. LOTTIAUX. *Gestion dynamique de services dans une grappe de calculateurs.* in « Journées des jeunes chercheurs en systèmes d'exploitation (ASF) », pages 485-592, Hammamet, Tunisie, avril, 2002, http://www.irisa.fr/paris/Biblio/Papers/Gallard/GalMorLot02renpar.ps.

[23] Y. JÉGOU. *Performance Analysis of Code Coupling on a Long Distance High Bandwidth Network.* in « EuroPar 2002, Parallel Processing », série Lecture Notes in Computer Science, volume 2400, pages 753-756, août, 2002.

[24] R. LOTTIAUX. *Les conteneurs : une approche générique pour la conception d'un système à image unique.* in « Journées des jeunes chercheurs en systèmes d'exploitation (ASF) », pages 457-464, Hammamet, Tunisie, avril, 2002, http://www.irisa.fr/paris/Biblio/Papers/Lottiaux/Lot02RenPar.ps.

[25] C. MORIN, P. GALLARD, R. LOTTIAUX, G. VALLÉE. *Towards an Efficient Single System Image Cluster Operating System.* in « Proc. of Intl. Conference on Architecture and Algorithms for Parallel Processing (ICA3PP-2002) », IEEE Computer Society, pages 370-377, Beijing, P. R. China, octobre, 2002, http://www.irisa.fr/paris/Biblio/Papers/Morin/MorGalLotVal02ica3pp.ps.

[26] C. PÉREZ, T. PRIOL, A. RIBES. *A Parallel CORBA Component Model for Numerical Code Coupling.* in « Proc. of the 3nd International Workshop on Grid Computing », série Lecture Notes in Computer Science, numéro 2242, Springer-Verlag, éditeurs C. A. LEE., Baltimore, Maryland, USA, novembre, 2002, http://www.irisa.fr/paris/Biblio/Papers/Perez/PerPriRib02GC.ps, à paraître.

[27] A. RIBES. *Vers l'utilisation de composants parallèles pour le couplage de codes de calcul scientifique.* in « 14èmes Rencontres Francophones du Parallélisme (RenPar'14) », pages 107-114, Hammamet, Tunisie, avril, 2002, http://www.irisa.fr/paris/Biblio/Papers/Ribes/Rib02RP.ps.

[28] G. UTARD, C. MORIN. *Clustix, a Cluster Operating System Based on Shared Memory Paradigm.* in « Proc. of Workshop on Caching, Coherence and Consistency (WC3 2002) », New York, USA, juin, 2002, http://www.irisa.fr/paris/Biblio/Papers/Utard/UtaMor02wc3.ps.

[29] G. VALLÉE, C. MORIN, J.-Y. BERTHOU, I. D. MALEN, R. LOTTIAUX. *Process Migration based*

on Gobelins Distributed Shared Memory. in « Proc. 2002 Intl. Workshop on Distributed Shared Memory on Clusters (DSM 2002) », IEEE Computer Society, pages 325-330, Berlin, Allemagne, mai, 2002, http://www.irisa.fr/paris/Biblio/Papers/Vallee/ValMorBerDutLot02dsm.ps, Organized at IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2002).

[30] G. VALLÉE. *Mécanismes de gestion globale de la ressource processeur au sein du système Gobelins.* in « Journées des jeunes chercheurs en systèmes d'exploitation (ASF) », pages 393-400, Hammamet, Tunisie, avril, 2002, http://www.irisa.fr/paris/Biblio/Papers/Vallee/Val02asf.ps.

Rapports de recherche et publications internes

[31] G. ANTONIU, L. BOUGÉ, T. PRIOL. *Partage de mémoire à très grande échelle sur des réseaux pair-à-pair.* Rapport de recherche, numéro RR-4410, INRIA, IRISA, Rennes, France, mars, 2002, http://www.inria.fr/rrrt/rr-4410.html, À paraître dans les actes de l'école GRID 2002, Aussois, décembre 2002.

[32] R. BADRINATH, C. MORIN. *Common Mechanisms for supporting fault tolerance in DSM and message passing systems.* Rapport de recherche, numéro RR-4613, INRIA, IRISA, Rennes, France, novembre, 2002, http://www.inria.fr/rrrt/rr-4613.html.

[33] L. BOUGÉ, V. DANJEAN, R. NAMYST. *Improving reactivity to I/O events in multithreaded environments using a uniform, scheduler-centric API.* Rapport de recherche, numéro RR-4471, INRIA, IRISA, Rennes, France, juin, 2002, http://www.inria.fr/rrrt/rr-4471.html.

[34] A. DENIS. *CORBA haute peformance.* Rapport de recherche, numéro RR-4553, INRIA, IRISA, Rennes, France, septembre, 2002, http://www.inria.fr/rrrt/rr-4553.html.

[35] A. DENIS, C. PÉREZ, T. PRIOL. *Towards High Performance CORBA and MPI Middlewares for Grid Computing.* Rapport de recherche, numéro RR-4555, INRIA, IRISA, Rennes, France, septembre, 2002, http://www.inria.fr/rrrt/rr-4555.html.

[36] A. DENIS, C. PÉREZ, T. PRIOL. *PadicoTM : An Open Integration Framework for Communication Middleware and Runtimes.* Rapport de recherche, numéro RR-4554, INRIA, IRISA, Rennes, France, septembre, 2002, http://www.inria.fr/rrrt/rr-4554.html.

[37] P. GALLARD, C. MORIN, R. LOTTIAUX. *Dynamic Resource Management in a Cluster for Scalability and High-Availability.* Rapport de recherche, numéro RR-4347, INRIA, IRISA, Rennes, France, janvier, 2002, Extended version of "Journées des jeunes chercheurs en systèmes d'exploitation (ASF)".

[38] N. JAIN, É. MÉMIN, C. PÉREZ. *Parallelization of Dense Fluid Motion Estimation Application using OpenMP.* Rapport de recherche, numéro RR-4556, INRIA, IRISA, Rennes, France, septembre, 2002, http://www.inria.fr/rrrt/rr-4556.html.

[39] N. T. KARONIS, B. DE SUPINSKI, I. FOSTER, W. GROPP, E. LUSK, S. LACOUR. *A Multilevel Approach to Topology-Aware Collective Operations in Computational Grids.* rapport technique, numéro ANL/MCS-P948-0402, Mathematics and Computer Science Division, Argonne National Laboratory, avril, 2002, ftp://info.mcs.anl.gov/pub/tech_reports/reports/P948.ps.Z.

[40] C. PÉREZ, T. PRIOL, A. RIBES. *A Parallel CORBA Component Model.* Rapport de recherche, numéro RR-4552, INRIA, IRISA, Rennes, France, septembre, 2002, http://www.inria.fr/rrrt/rr-4552.html.

[41] G. V. RENAUD LOTTIAUX. *Containers : an Architecture for an Efficient Cluster Operating System.* Rapport de recherche, numéro RR-4384, INRIA, IRISA, Rennes, France, février, 2002, http://www.inria.fr/rrrt/rr-4384.html.

[42] G. VALLÉE, C. MORIN, J.-Y. BERTHOU, I. DUTKA-MALEN, R. LOTTIAUX. *Efficient Process Migration based on Gobelins Distributed Shared Memory.* Rapport de recherche, numéro RR-4518, INRIA, IRISA, Rennes, France, août, 2002, http://www.inria.fr/rrrt/rr-4518.html.

Divers

[43] L. BOUGÉ. *Si tous les ordinateurs du monde... Calcul scientifique vraiment très haute performance sur la grille mondiale.* série Bulletin de l'association des anciens élèves et des élèves de de l'ENS Cachan, numéro 216, février, 2002, http://www.irisa.fr/paris/biblio/Papers/Bouge/Bou02Bulletin.ps.gz.

[44] Y. JÉGOU, C. MORIN. *Collaboration Alcatel-INRIA Software Bus, Contrat n^o 1 01C 0101 00 31329 01 2, livrable n^{o}1.* mars, 2002.

[45] Y. JÉGOU, C. MORIN. *Collaboration Alcatel-INRIA Software Bus, Contrat n^o 1 01C 0101 00 31329 01 2, livrable n^{o}2.* novembre, 2002.

[46] Y. JÉGOU, C. PÉREZ, T. PRIOL. *Projet RNRT VTHD++, livrable n^{o}1.* novembre, 2002.

[47] Y. JÉGOU, C. PÉREZ, T. PRIOL. *Projet RNRT VTHD, Couplage de codes pour la simulation numérique distribuée, livrable final.* janvier, 2002.

[48] S. LACOUR. *Mémoire virtuellement partagée pour grappes de grappes : conception, mise en oeuvre et évaluation d'un protocole de cohérence.* Rapport de stage de DEA, DEA d'informatique de l'IFSIC, Université de Rennes 1, France, juin, 2002, http://www.irisa.fr/paris/biblio/Papers/Lacour/rapport_DEA.pdf.

Bibliographie générale

[49] G. ANTONIU. *DSM-PM2 : une plate-forme portable pour l'implémentation de protocoles de cohérence multithreads pour systèmes à mémoire virtuellement partagée.* Thèse de doctorat, ENS Lyon, France, LIP, novembre, 2001, In French.

[50] P. ARBENZ, W. GANDER, M. OETTLI. *Remote computation system.* in « Parallel Computing », numéro 10, volume 23, octobre, 1997, pages 1421-1428.

[51] H. CASANOVA, J. DONGARRA. *NetSolve : A Network-Enabled Server for Solving Computational Science Problems.* in « The International Journal of Supercomputer Applications and High Performance Computing », numéro 3, volume 11, 1997, pages 212-223.

[52] I. FOSTER, C. KESSELMAN. *Globus : A Metacomputing Infrastructure Toolkit.* in « The International Journal of Supercomputer Applications and High Performance Computing », numéro 2, volume 11, 1997, pages 115-

128.

[53] éditeurs I. FOSTER, C. KESSELMAN., *The Grid : Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers, 1998.

[54] K. GHARACHORLOO, D. LENOSKI, J. LAUDON, P. GIBBONS, A. GUPTA, J. HENESSY. *Memory Consistency and event ordering in scalable shared memory multiprocessors.* in « 17th Annual Intl. Symposium on Computer Architectures », ACM, pages 15-26, mai, 1990.

[55] A. S. GRIMSHAW, W. A. WULF. *The Legion vision of a worldwide virtual computer.* in « Communications of the ACM », numéro 1, volume 40, janvier, 1997, pages 39-45.

[56] P. KELEHER, A. COX, W. ZWAENEPOEL. *Lazy Release Consistency for Software Distributed Shared Memory.* in « 19th Intl. Symposium on Computer Architecture », pages 13-21, mai, 1992.

[57] P. KELEHER, D. DWARKADAS, A. COX, W. ZWAENEPOEL. *TreadMarks : Distributed Shared Memory on standard workstations and operating systems.* in « Proc. 1994 Winter Usenix Conference », pages 115-131, janvier, 1994.

[58] Z. LAHJOMRI, T. PRIOL. *KOAN : A Shared Virtual Memory for iPSC/2 Hypercube.* in « Proc. of the 2nd Joint Int'l Conf. on Vector and Parallel Processing (CONPAR'92) », pages 441-452, septembre, 1992.

[59] K. LI. *Shared Virtual Memory on Loosely Coupled Multiprocessors.* PhD Thesis, Yale University, septembre, 1986.

[60] G. MULLER, B. MOURA, F. BELLARD, C. CONSEL. *Harissa : A Flexible and Efficient Java Environment Mixing Bytecode and Compiled Code.* in « Proc. 3rd Conference on Object-Oriented Technologies and Systems », Usenix Association, pages 1-20, Berkeley, juin, 1997.

[61] *OpenMP Fortran Application Program Interface.* novembre, 2000, Version 2.0.

[62] J. R. RICE, R. F. BOISVERT. *From Scientific Software Libraries to Problem-Solving Environments.* in « IEEE Computational Science and Engineering », numéro 3, volume 3, 1996, pages 44-53.

[63] A. TAKEFUSA, U. NAGASHIMA, S. MATSUOKA, H. OGAWA, H. NAKADA, S. SEKIGUCHI, H. TAKAGI, M. SATO. *Multi-client LAN/WAN Performance Analysis of Ninf : A High Performance Global Computing System.* in « Proc. SC97 : High Performance Networking and Computing », ACM Press and IEEE Computer Society Press, San Jose, California, novembre, 1997.