INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet SOR

Systèmes Objets Répartis

Rocquencourt

THÈME 1B

Table des matières

1 Composition de l'équipe

Responsable scientifique

Mesaac Makpangou [CR, INRIA]

Assistante de projet

Brigitte Larue [INRIA]

Conseiller scientifique

Bertil Folliot [professeur à l'Université Paris 6]

Collaborateurs extérieurs

Marc Badel [ingénieur de l'armement DGA, jusqu'en octobre]

Vincent Bouthors [ingénieur de recherche Bull]

Patrick Duval [professeur assistant, Pôle Universitaire Léonard de Vinci]

Marc Shapiro [Microsoft Research Cambridge]

Pierre Sens [Mdc Université Paris 6]

Ingénieurs experts

Pierre Albertin [jusqu'en mai]

Xavier Blondel

Ian Piumarta [jusqu'en juillet]

Ngock-Koi Tô [à partir de novembre]

Doctorants

Aline Baggio [bourse INRIA, Université Paris 6, jusqu'en juin]

Carine Baillarguet [bourse MENRT, Université Paris 6]

Olivier Dedieu [bourse Bull, Université Marne-La-Vallée]

Neilze Dorta [bourse INRIA, Université Paris 6]

Christian Khoury [bourse INRIA, Université Paris 6, jusqu'en septembre]

Fabrice Le Fessant [bourse de l'École Polytechnique, Université Paris 7, co-tutelle avec le projet MOSCOVA]

Simon Patarin [bourse ENS Lyon, Université Paris 6, depuis novembre]

Guillaume Pierre [bourse INRIA, Université d'Évry-Val d'Essonne, jusqu'en juin]

Nicolas Richer [bourse MENRT, Université Paris 6]

Stagiaires

Gautier Harmel [DEA Systèmes Informatiques, Paris 6]

Simon Patarin [ENS Lyon, DEA Informatique de Lyon]

Alexandru Salicianu [DEA Informatique de Lyon]

2 Présentation et objectifs généraux

Mots clés : adaptabilité, adaptation, cache, chaîne de paires souche-schion, cohérence, configuration de systèmes, dimensionnement, entrepôt de données, gestion de cohérence, grande échelle, interopérabilité, machine virtuelle virtuelle, mémoire partagée répartie, mobilité, optimisation de systèmes, PerDiS, persistance, ramasse-miettes réparti, Relais, réplication, spécialisation, système réparti, travail coopératif, WWW.

Le projet SOR a pour thème central les mécanismes de partage de données par des agents ou processus coopérant sur l'Internet. Notre recherche est guidée à la fois par l'innovation technologique (démarche ascendante) et par les besoins des applications réelles (démarche descendante). Elle comprend quatre facettes : (1) Caractérisation de l'application, des environnements d'exécution cibles, ainsi que du schéma d'accès aux données. Ceci nous conduit à préciser les hypothèses et les contraintes à prendre en compte lors de l'élaboration des solutions. (2) Spécification des algorithmes, des modèles ou des protocoles satisfaisant aux contraintes identifiées. Lorsque cela s'y prête bien, les preuves (de correction et/ou de vivacité) des algorithmes et protocoles sont réalisées. (3) Prototypage des mécanismes, algorithmes et protocoles. Le prototypage, qui représente un investissement lourd et ingrat, est néanmoins indispensable pour se convaincre de la viabilité des solutions proposées. (4) Évaluation, qui se fait en deux temps. En amont du prototypage, les simulations permettent d'évaluer et de comparer différents compromis. En aval du prototypage, l'évaluation nous apprend, d'une part si le système satisfait aux attentes des utilisateurs, et d'autre part si le comportement obtenu est conforme aux prédictions.

3 Fondements scientifiques

3.1 La réplication de données et la cohérence

Mots clés : cache, cohérence, grande échelle, réplication.

La coopération sur l'Internet implique le partage de l'information entre des agents ou processus répartis sur des sites géographiquement séparés. Or, les ordinateurs d'un système réparti ne partagent pas de mémoire. Ils peuvent communiquer par des appels de procédure à distance, mais la latence des communications entre des machines distantes rend cette solution inadaptée pour une coopération efficace sur l'Internet. La réplication des données s'impose donc, mais celle-ci pose des problèmes fondamentaux et difficiles.

Considérons des objets, désignés par des références, et accédés par invocation de leurs méthodes. On peut rendre une référence à un objet distant, et l'appel de ses méthodes, indifférenciables du cas local grâce à un objet d'indirection, dit souche ou mandataire. Ce schéma dit d'*invocation distante* permet bien le partage transparent des objets ; c'est celui qui est utilisé dans Corba [DHH+91] et Java RMI [WRW96]. Mais l'invocation distante est coûteuse, et ne passe pas à l'échelle si l'objet partagé constitue un goulot d'étranglement. De plus, si ce schéma simplifie la communication, il ne résout pas les problèmes plus profonds de la répartition : parallélisme, défaillances, coûts, etc.

Le coût de l'accès distant et la présence de ressources matérielles abondantes conduisent naturellement à gérer des copies, ou *réplicats*, de la donnée distante. Si le système crée automatiquement les réplicats distants au fur et à mesure des accès, dans une mémoire tampon invisible, on appelle cela un *cache*. Si toute la mémoire est gérée de cette façon, on parle d'une *mémoire répartie virtuellement partagée* ou MRVP. La réplication augmente la disponibilité des données et permet de tolérer les fautes. Lorsque les données sont accédées en lecture, la réplication élimine les goulots d'étranglement. Si une donnée répliquée est modifiée, les autres réplicats deviennent incohérents, et il faut propager la mise à jour. La réplication à grande échelle rencontre une contradiction majeure : assurer la cohérence des réplicats tout en conservant des performances acceptables. Un protocole de cohérence fort [LH89] est simple mais ne passe pas à l'échelle, d'où des recherches actives sur des cohérences affaiblies [KCZ92,GC89,BZ91,GL91]. Il y a lieu de rechercher un compromis entre contraintes de cohérence, tolérance aux fautes et performances.

[DHH+91] Digital Equipment Corporation, Hewlett-Packard Company, HyperDesk Corporation, NCR Coporation, Object Design, Inc., SunSoft, Inc., « The Common Object Request Broker: Architecture and Specification», *rapport de recherche n° 91-12-1*, Object Management Group, Framingham MA (USA), décembre 1991.

[WRW96] A. Wollrath, R. Riggs, J. Waldo, « A Distributed Object Model for the Java System», *in : Conf. on Object-Oriented Technologies*, Usenix, Toronto, Ontario (Canada), juin 1996.

[LH89] K. Li, P. Hudak, « Memory Coherence in Shared Virtual Memory Systems», p. 321–359.

[KCZ92] P. Keleher, A. L. Cox, W. Zwaenepoel, « Lazy Release Consistency for Software Distributed Shared Memory », *in : Proc. 19th Int. Symposium on Comp. Architecture*, p. 13–21, Gold Coast (Australia), mai 1992.

[GC89] C. Gray, D. Cheriton, « Leases: An Efficient Fault-Tolerant Mechanism for Distributed File Cache Consistency», *in : Proceedings of the 12th ACM Symposium on Operating Systems Principles*, ACM, p. 202–210, Litchfield Park AZ USA, décembre 1989.

[BZ91] B. N. Bershad, M. J. Zekauskas, « Midway: Shared Memory Parallel Programming with Entry Consistency for Distributed Memory Multiprocessors», *rapport de recherche n° CMU-CS-91-170*, Carnegie-Mellon University, Pittsburgh, PA (USA), septembre 1991.

[GL91] R. Golding, D. D. E. Long, « Accessing Replicated Data in a Large-Scale Distributed System», *rapport de recherche n° UCSC-CRL-91-01*, Computer Research Lab., U. of California, Santa Cruz CA (USA), janvier 1991.

De plus, les défaillances et l'hétérogénéité des politiques d'administration et de protection induisent des discontinuités dans le réseau. Le programmeur d'application exige néanmoins une continuité de service. Il faut donc d'une part, anticiper de telles discontinuités de service et d'autre part offrir des mécanismes de réconciliation pour rétablir la cohérence des réplicats qui ont été momentanément isolés les uns des autres.

Nos recherches sur la réplication à grande échelle concernent : (i) les moyens offerts aux programmeurs pour exprimer les compromis qui répondent le mieux à leurs attentes (voir §6.3 et §5.5) ; (ii) les modèles de cohérence faible pour nos applications cibles (voir §5.4) ; (iii) la garantie de la continuité de service sur des machines faiblement connectées (voir §6.5).

3.2 Persistance et ramasse-miettes

Mots clés : entrepôt de données, mémoire partagée répartie, PerDiS, persistance, ramasse-miettes réparti.

Le partage des données entre programmes ne s'exécutant pas en même temps nécessite des données «persistantes», conservées par une exécution en vue d'utilisation ultérieure par une autre.

La persistance est le complément indispensable de la répartition pour permettre le partage de données, aussi bien pour les tâches quotidiennes comme le traitement de texte, que pour des applications émergentes telles le travail coopératif. L'importance de la persistance est généralement sous-estimée par la communauté des systèmes répartis. Les programmeurs ont l'habitude de la gérer manuellement (grâce aux systèmes de fichiers ou aux bases de données). Cependant, la gestion manuelle de la persistance pose le problème de détection et du ramassage de miettes (i.e. des objets persistants référencés par aucun utilisateur) dans les systèmes répartis.

Un modèle efficace qui assure la gestion automatique de la persistance est la «persistance par atteignabilité» : un objet est persistant si et seulement si il est atteignable depuis une racine. Ce modèle exige un ramasse-miettes. Pourtant, les algorithmes de ramasse-miettes répartis sont mal connus. La conception et la mise en œuvre de ces algorithmes est l'axe central de recherche du projet SOR (voir §6.2).

3.3 Extensibilité et interopérabilité

Mots clés : adaptabilité, machine virtuelle virtuelle, spécialisation.

La plupart des systèmes d'exploitation sont mal adaptés aux paradigmes de programmation actuels. Ils sont difficilement spécialisables pour répondre aux besoins d'une application donnée. Or, les programmes, données et leurs politiques de gestion doivent être spécialisables et adaptables afin de prendre en compte par exemple les évolutions technologiques, les besoins spécifiques d'un domaine applicatif, ou les caractéristiques de l'environnement d'exécution.

Un environnement d'exécution virtuel et un langage de programmation à objets réduisent la complexité des développements, facilitent la ré-utilisation, tout en améliorant la qualité des logiciels. Si dans le passé la technologie des machines virtuelles a été considérée comme trop coûteuse pour des systèmes d'exploitation, cette objection tombe avec les nouvelles générations d'architectures processeurs à haute performance et les nouvelles techniques de compila-

tion [NHCL98]. La construction d'un système d'exploitation basé sur une machine virtuelle et un langage de programmation à objets, de manière similaire à Java, est donc une approche attrayante. Cependant, les machines virtuelles existantes sont encore trop rigides et imposent un contrôle strict sur ce que l'application peut effectuer. Autrement dit, si la machine virtuelle ne contient pas explicitement toutes les opérations dont le langage de programmation a besoin, il n'y a pas d'autres solutions que de modifier la machine virtuelle. Ceci entraîne une multiplication de machines virtuelles différentes et incompatibles, la difficulté de ré-utilisation des logiciels et l'absence de coopération entre applications écrites dans des langages différents.

Nos recherches dans ce domaine visent à unifier les environnements d'exécution virtuelle, au moyen d'une Machine Virtuelle Virtuelle (MVV). A la différence d'une machine virtuelle «classique», la MVV est capable d'étendre à la volée son jeu d'instructions de manière à s'adapter dynamiquement à de nouveaux types d'applications (voir §6.6).

Cette approche de MVV présente les avantages suivants : 1) elle n'impose pas un langage d'exécution unique, mais permet d'exécuter simultanément des applications écrites dans différents langages ; 2) la représentation interne des objets manipulés par la MVV étant neutre du point de vue de l'application, il n'y a pas de limite à l'interopérabilité des différents langages qui s'exécutent ; 3) les techniques de recompilation dynamique et de ré-organisation de *bytecode* sont incluses dans la MVV. Ainsi, les performances ne devraient pas être dégradées par rapport à une machine virtuelle "classique", tout en diminuant la difficulté de programmation.

4 Domaines d'applications

4.1 Panorama

Résumé : *Tous les domaines d'application du travail coopératif sur l'Internet sont visés. Actuellement le projet se concentre sur la réalisation d'entrepôts de documents partagés par des communautés d'utilisateurs et la mise en commun d'expertises au-dessus du World-Wide Web (voir §4.2), à l'ingénierie coopérative (voir §4.3), à la gestion des données personnelles sur machines nomades (voir §4.4), et aux systèmes configurables comme les réseaux actifs (voir §4.5).*

4.2 Application au World-Wide Web

Participants :
Mesaac Makpangou, Vincent Bouthors, Patrick Duval, Guillaume Pierre, Olivier Dedieu, Simon Patarin, Christian Khoury, Neilze Dorta.

Le succès du World-Wide Web (la *Toile*) a fait naître de nombreux besoins de partage d'informations à grande échelle. En particulier, les personnels des entreprises réparties sur plusieurs sites géographiques utilisent de plus en plus la *Toile* comme véhicule pour le partage des informations produites en interne ou non. Ces différentes applications n'ont pas toutes les mêmes

[NHCL98] F. Noël, L. Hornof, C. Consel, J. L. Lawall, « Automatic, Template-Based Run-Time Specialization: Implementation and Experimental Study », *in : Proceedings of ICCL'98*, 1998. http://www.irisa.fr/compose/papers/rt_bench.ps.gz.

attentes concernant par exemple la cohérence de l'information accédée, les performances, le coût des communications pour l'entreprise, etc. Les besoins varient notamment en fonction des caractéristiques des connexions entre les sites, du nombre de sites et des contraintes de l'entreprise.

Pour faciliter ce partage, nous offrons l'abstraction d'un entrepôt réparti de documents garantissant à chaque entreprise la qualité de service conforme à ses besoins. L'entrepôt est réalisé concrètement par un système de caches réparti et flexible. Celui-ci est couplé à un système d'évaluation qui permet de configurer les caches de façon optimale, en fonction des attentes, des caractéristiques du trafic et des moyens de l'entreprise (voir §5.5 et §6.3).

En outre, nous développons un service d'annotations, permettant à un groupe de personnes de partager des jugements sur les documents disponibles. Ce type de méta-information permettra par exemple d'interdire l'accès de certains documents à certaines catégories d'utilisateurs, ou de faciliter la recherche de documents pertinents. Un exemple est l'utilisation par des enseignants, annotant les documents du Web et utilisant les annotations de leurs collègues. Ceci leur permettra par exemple de trouver les documents présentant un intérêt pédagogique pour leurs élèves (voir §5.8).

4.3 Application à l'ingénierie coopérative

Participants : Marc Shapiro, Xavier Blondel, Nicolas Richer, Alexandru Salicianu, Ngock-Koi Tô, Pierre Albertin.

L'abstraction d'un entrepôt persistant réparti est adaptée aux besoins du partage d'informations des applications de génie civil et de la construction (voir §5.2 et §6.2).

Un bâtiment est constitué d'un grand nombre d'objets physiques en relation complexe, et soumis à des règles strictes. Des systèmes de CAO spécialisés existent en version mono-poste, mais le partage de l'information entre architectes et ingénieurs est non résolu dans la pratique.

Par ailleurs, la conception et la construction d'un grand bâtiment fait intervenir un nombre important d'acteurs : architectes, ingénieurs de structure, ingénieurs chauffagistes, ingénieurs électriciens, etc. Ceux-ci appartiennent souvent à des entreprises différentes, mais se regroupent pour un projet particulier en «entreprise virtuelle». Il faut encourager le partage de l'information pertinente pour ce projet, tout en protégeant l'accès à d'autres données. Enfin, et en particulier pour des raisons légales, les informations sont à stocker de façon très fiable sur de longues durées.

Afin de faciliter le portage des applications de CAO existantes, le projet Esprit PerDiS (voir §8.2.1) propose une mémoire partagée répartie persistante. La conjonction des techniques de persistance par atteignabilité et de mémoire partagée répartie facilite énormément la tâche du programmeur d'application. La mémoire persistante doit par ailleurs protéger la confidentialité des données et tolérer les pannes.

4.4 Application aux données personnelles et à l'informatique nomade

Participants : Marc Shapiro, Aline Baggio, Fabrice Le Fessant.

Le troisième domaine d'application est le partage des données personnelles, entre les dif-

férentes machines qu'un utilisateur donné accède au cours du temps. Ces données sont, par exemple, ses répertoires de courrier électronique ou son agenda.

Nous nous intéressons plus particulièrement au cas des machines nomades. Ce type de réplication est plus simple à gérer que le problème général de la réplication sur des machines faiblement connectées. C'est la présence de l'utilisateur et ses actions explicites qui déclenchent l'activité de cohérence. Cela a pour conséquence, d'une part que le contrôle de concurrence est particulièrement simple, et d'autre part qu'une cohérence affaiblie est bien adaptée. Nous étudions en particulier les algorithmes de réplication dits «épidémiques» [TTP+95,PST+97] qui sont bien adaptés aux besoins applicatifs et aux capacités des machines nomades. Par ailleurs, la sémantique de certains types de données est bien connue, ce qui permet la réconciliation automatique entre des réplicats ayant divergé.

4.5 Application aux systèmes configurables

Participants : Ian Piumarta, Bertil Folliot, Carine Baillarguet.

Plusieurs systèmes ont besoin d'être convenablement configurés ou spécialisés afin de prendre les contraintes qu'imposent par exemple leur environnements d'exécution. À titre d'exemple, on peut citer les réseaux actifs, les agents mobiles, les systèmes embarqués et les cartes à puce. Chacune de ces applications a ses contraintes particulières ; par exemple dans les cartes à puces ou les réseaux actifs, la taille du code est extrêmement limitée alors que le jeu d'instructions doit être spécialisable en fonction des applications supportées par la carte à puce ou les protocoles spécifiques. La Machine Virtuelle Virtuelle offre à ces applications une plate-forme d'exécution susceptible d'accueillir des agents dynamiquement spécialisables, interchangeables et inter-opérables (voir §6.6).

5 Logiciels

5.1 Panorama

Résumé : *Plusieurs logiciels destinés aux utilisateurs ou programmeurs des domaines d'applications cités au § 4.1 sont aujourd'hui distribués sur le Web par le projet SOR. Ces logiciels sont aujourd'hui des prototypes stables, robustes, utilisés aussi bien dans le projet qu'à l'extérieur.*

PerDis (voir §5.2) constitue la plate-forme de développement du projet Esprit LTR PerDis dont le projet SOR assure la coordination; cette plate-forme est accompagnée d'un système de mesure et d'analyse des caractéristiques des mémoires

[TTP+95] D. Terry, M. Theimer, K. Petersen, A. Demers, M. Spreitzer, C. H. Hauser, «Managing Update Conflict in Bayou, a Weakly Connected Replicated Storage System», *in: 15th ACM Symposium on Operating Systems Principles*, Copper Mountain Resort, Colorado, US, décembre 1995. http://www.parc.xerox.com/csl/projects/bayou/.

[PST+97] K. Petersen, M. J. Spreitzer, D. B. Terry, M. M. Theimer, A. J. Demers, « Flexible Update Propagation for Weakly Consistent Replication », *in: Sixteen ACM Symposium on Operating Systems Principles*, Saint Malo, France, octobre 1997. http://www.parc.xerox.com/csl/projects/bayou/pubs/sosp-97/.

d'objets permettant d'évaluer l'adéquation des politiques et heuristiques d'allocation et dés-allocation automatique de la mémoire (voir §5.3).

À l'intention des utilisateurs du Web, nous avons développé plusieurs logiciels dont les plus importants sont: Relais, un système de caches coopérants (§5.4) ; un outil de monitoring du trafic Web pour des larges communautés d'utilisateurs (§5.6) ; un système d'évaluation, de configuration et le dimensionnement des systèmes de caches Web répartis (§5.5) ; le proxy configurable (§5.7) et le système d'annotations et de recommandations de documents sur le Web (§5.8).

Concernant les deux autres domaines d'applications, nous mettons à disposition un système support pour le partage des données personnelles sur des machines mobiles (voir §5.9) et la Machine Virtuelle Récursive (MVR), première étape vers la Machine Virtuelle Virtuelle (voir §5.10).

5.2 PerDiS : Un entrepôt persistant réparti

Participants : Xavier Blondel, Nicolas Richer, Alexandre Salicianu, Marc Shapiro.

Dans le cadre du projet Esprit PerDiS, le projet SOR dirige le développement de la plate-forme PerDiS. Cette plate-forme offre aux utilisateurs un entrepôt persistant réparti (§6.2 et §8.2.1). L'entrepôt permet le partage de grappes de données à travers l'Internet. La première application est la CAO coopérative pour le bâtiment. Pour la réalisation de la plate-forme, nous utilisons une combinaison novatrice des techniques de ramasse-miettes, de cache, de transactions et de sécurité.

La dernière version de la plate-forme (`http://www.perdis.esprit.ec.org/download/`) fonctionne sous Windows NT. Des versions antérieures pour Sun, Linux et HP/UX sont également disponibles. Actuellement, les applications utilisant l'entrepôt PerDiS doivent être écrites en C ou en C++.

5.3 Analysis : Plate-forme de mesure et d'analyse des caractéristiques des mémoires d'objets

Participant : Nicolas Richer.

Dans le cadre du projet Esprit d'entrepôt persistant réparti PerDiS, un ensemble d'outils de mesure, de simulation et d'analyse des caractéristiques des mémoires d'objets (appelé Analysis) a été réalisé. Analysis permet d'évaluer les politiques et heuristiques d'allocation et de dés-allocation automatique de la mémoire en mesurant le comportement d'applications réelles. Cette connaissance permet ensuite de sélectionner les meilleures stratégies de gestion de la mémoire pour l'entrepôt PerDiS et éventuellement d'adapter celles-ci aux besoins spécifiques de certaines classes d'applications. L'usage de plate-forme Analysis n'est toutefois pas limité à PerDiS. Analysis peut être utilisé pour caractériser n'importe quel type de mémoire d'objet.

Cette plate-forme s'articule autour d'une représentation des graphes d'objets persistants sous la forme d'un journal générique binaire et d'une bibliothèque permettant de manipuler ce journal. Pour l'heure, la plate-forme comprend : un outil d'analyse de graphe avec de nombreuses fonctionnalités et très facilement extensible ; un outil de simulation permettant de

reproduire les allocations/dés-allocations mémoires avec différentes politiques ; et un outil permettant de recueillir dans le journal le graphe d'objets présents en mémoire durant l'exécution d'une application.

La dernière version de la plate-forme est toujours disponible depuis la page: `http://www-sor.inria.fr/projects/perdis/analysis/`. Elle est régulièrement testé sous Linux, Solaris et Digital Unix, mais elle devrait fonctionner sans problème majeur sur la plupart des systèmes Unix. Une documentation complète des différents outils et du journal générique est fourni en format Postscript et Adobe pdf.

5.4 Relais : Système de caches Web coopérants

Participants : Mesaac Makpangou, Christian Khoury.

Relais met en œuvre la coopération entre des caches Web, par exemple à l'intérieur d'une entreprise. Grâce à Relais, tout utilisateur de l'entreprise accède aux documents mis en cache n'importe où dans l'entreprise.

Relais permet notamment d'améliorer les performances et la disponibilité des documents Web, quelles que soient la localisation et la charge des serveurs d'origine. Relais offre également des garanties de cohérence à ses utilisateurs. En particulier, il garantit à chaque utilisateur une progression monotone et rapide dans les versions des documents qu'il accède, ainsi que la convergence des caches coopérants [12].

Un prototype de Relais est aujourd'hui disponible (`http://www-sor.inria.fr/projects/relais/relais.html`) sur différentes plates-formes, notamment : Linux, Digital Unix, AIX, Solaris et SunOS. Ce prototype est utilisé par le projet SOR pour ses accès Web.

5.5 Saperlipopette! : Environnement de simulation de caches Web distribués

Participants : Guillaume Pierre, Mesaac Makpangou.

Saperlipopette! est un outil d'évaluation des performances de configurations de caches Web répartis. Il permet à un administrateur système de capturer les caractéristiques du trafic Web de ses utilisateurs, de même que la qualité de service de sa connexion réseau. Il est alors possible d'injecter le trafic capturé dans différentes configurations du système, et d'analyser les performances résultantes. Grâce aux informations fournies par Saperlipopette!, les administrateurs peuvent décider objectivement d'une configuration à mettre en place, en connaissance de cause par rapport aux performances attendues comme aux coûts induits.

L'environnement Saperlipopette! est diffusé sous licence GPL (`http://www-sor.inria.fr/projects/relais/saperli/`). Cette diffusion libre a permis à l'ENST Bretagne de l'utiliser et de l'étendre lors d'un stage de DEA afin d'y intégrer un meilleur modèle de simulation des réseaux sous-jacents. La Compagnie des Signaux l'utilise également pour la simulation prospective de l'Internet français à un horizon de 3 ans.

5.6 Pandora : un système de collecte de traces du trafic Web de communautés d'utilisateurs réparties

Participants : Simon Patarin, Mesaac Makpangou.

Pandora est un outil de surveillance réseau qui peut être utilisé pour capturer le trafic Web d'une communauté d'utilisateurs répartie. Ces informations sont obtenues en reconstituant le trafic HTTP à partir des paquets bruts circulant sur le réseau [15, 14]. Malgré les difficultés inhérentes à cette étape de reconstruction, la transparence et la facilité de déploiement qu'offrent cette solution la rendent incomparable.

Pandora dispose des fonctionnalités suivantes : traitement à la volée, compatibilité avec le protocole HTTP/1.1, masquage des adresses IP et des URL accédées – tout en conservant leur structure – pour des raisons de confidentialité. Dans les cas où des caches Web sont présents sur le système, Pandora est capable de reconstruire le trafic qui aurait été vu en leur absence.

Pandora offre plusieurs perspectives. Il fournit la matière première pour caractériser l'usage du Web. Il permet d'envisager la conception de logiciels Web adaptatifs. Il peut aussi servir à évaluer des piles de protocoles en mesurant leurs coûts et leur performances sur des prototypes réels, au lieu de simulations.

Le prototype, toujours en développement, sera bientôt disponible pour les plates-formes Digital Unix, Linux et Solaris.

5.7 Pluxy : un proxy Web modulable

Participants : Olivier Dedieu, Vincent Bouthors.

Pluxy est un proxy Web modulaire capable d'héberger un nombre variable et dynamiquement extensible de services. Il fournit l'infrastructure nécessaire au téléchargement, à l'exécution et à la collaboration de ces services. Pluxy est accompagné d'un jeu de services de base qui permettent à plusieurs services présents de participer au traitement d'une même requête HTTP, de disposer d'une interface d'interaction avec l'utilisateur et d'interagir avec des services distants.

Afin de réduire les conflits entre services, Pluxy découpe le traitement d'une requête et d'une réponse en huit étapes élémentaires. Chaque service peut participer à une ou plusieurs de ces étapes.

La gestion des services est assurée par une plate-forme de composants logiciels. Chaque service est représenté par un composant. Ce composant peut être télé-chargé, installé puis chargé et retiré dynamiquement dans Pluxy. Un composant est matérialisé par son code (un ensemble de classes Java) et les données qui l'accompagnent (fichier de description, fichier de configuration, icônes, ...). Les composants peuvent collaborer. Pour cela, ils expriment des dépendances de chargement ou de comportement. Chaque composant reçoit lors de son installation une zone d'exploitation (i.e. un répertoire dans le système de fichier) qui lui permet de lire et d'écrire des données de façon persistante. Cette zone est sous le contrôle d'un gestionnaire de sécurité.

Plusieurs services ont été développés pour valider les concepts de Pluxy: Pharos (voir §5.8), la redirection de requêtes sur des miroirs, un traducteur de pages (en redirigeant sur *babelfish*), l'élimination de la publicité et des *cookies*.

Pluxy a été développé dans le cadre de l'action de développement Dyade/Webtools.

5.8 Pharos : Système de recommandation pour le Web

Participants : Vincent Bouthors, Olivier Dedieu.

Internet, avec le Web, est devenu une source d'informations d'une très grande richesse qu'il devient crucial pour les professionnels de savoir exploiter efficacement. Pharos permet à des groupes de personnes d'évaluer et de se recommander des pages Web pertinentes selon leur domaine d'intérêt [13, 10]. Les bases d'évaluations sont exploitées de façon collaborative. Pour affiner l'utilisation de ce service, des algorithmes de synthèse produisent pour chaque membre du groupe des recommandations personnalisées de pages intéressantes.

Pharos se compose d'une partie client et d'une partie serveur. Un serveur peut héberger un ou plusieurs groupes. Chaque groupe se compose de deux parties : le *backend* hébergé dans le serveur et le *frontend* hébergé dans le client. La communication se fait au dessus de RMI (*Remote Method Invocation*). La gestion des *frontends* et des *backends* repose sur un mécanisme de composants logiciels similaire à celui développé dans Pluxy (voir §5.7).

Un premier prototype comprenant les parties client et serveur est aujourd'hui disponible. Ceci permet de déployer Pharos chez plusieurs clients avec une base d'évaluations gérée par un serveur centralisé. Afin d'augmenter la disponibilité des bases d'évaluations, ces dernières peuvent être répliquées. Un gestionnaire de cohérence réparti assure la propagation des mises à jours entre les différents réplicats ainsi que la gestion des conflits.

Le développement de Pharos se fait dans le cadre de l'action Dyade/Webtools, en collaboration avec le CNET Lannion qui expérimente ce nouveau service auprès d'enseignants de la région Bretagne.

5.9 Cadmium : un système support pour le partage de données personnelles sur des machines faiblement connectées

Participant : Aline Baggio.

Les recherches menées sur les chaînes de PSS déconnectables et le support des applications mobiles, ont conduit à la création du prototype Cadmium.

Le but de Cadmium est de fournir aux utilisateurs mobiles un environnement de travail quasi-normal, c'est-à-dire similaire à leur environnement sur station fixe. Ceci passe par le support de la continuité de service, et par une gestion appropriée des dégradations (déplacement, perturbation sur le médium de transmission, déconnexion, etc.).

Afin d'assurer la continuité de l'accès aux données, Cadmium offre un environnement de désignation et de réplication adapté. Dans Cadmium le système et les applications collaborent afin de supporter la mobilité. Le système fournit les mécanismes de base : support de la migration des objets et des références flexibles. Les références flexibles sont une extension des Chaînes de PSS avec support de la réplication ; elles restent valides malgré les déconnexions. Le système fournit également un ensemble de politiques de base pour le contrôle d'accès, la réplication, la cohérence, etc. Finalement, il comprend les moyens de connaître l'évolution de l'environnement (monitoring) et d'en notifier les applications intéressées.

Une application utilisant Cadmium peut donc référencer les objets dont elle a besoin indépendamment des problèmes de mobilité. Elle peut toutefois personnaliser les mécanismes fournis par le système en ajoutant de nouvelles politiques, et en tenant compte des modifications de l'environnement pour adapter dynamiquement sa qualité de service.

Le prototypage a été réalisé en JoCaml (Objective Caml augmenté de fonctionnalités du Join Calculus). Ce prototype utilise une machine virtuelle modifiée, développée par Fabrice Le Fessant.

5.10 MVR : Machine Virtuelle Recursive

Participants : Carine Baillarguet, Bertil Folliot, Ian Piumarta.

La Machine Virtuelle Recursive (MVR) (`http://www-sor.inria.fr/projects/vvm/`) permet aux programmeurs d'étendre dynamiquement le jeu d'instructions de la machine virtuelle, ainsi que son jeu de primitives, par transformation de fonctions applicatives en instructions ou primitives optimisées. Elle démontre également l'intérêt d'une implémentation «ouverte», permettant aux programmeurs de définir eux-mêmes des facilités systèmes, tels que les processus légers, la synchronisation, les «continuations», de nouveaux éléments syntaxiques, et les extensions orientées-objet. Elle valorise aussi les mécanismes d'optimisation dynamique développés antérieurement. Ces mêmes mécanismes ont étés utilisés pour améliorer les performances de la machine virtuelle d'Objective Caml `http://www-sor.inria.fr/~piumarta/pldi98/ocaml-1.05/`.

6 Résultats nouveaux

6.1 Panorama

Résumé : *En 1999, en plus de la mise au point des logiciels ainsi que leur portage sur plusieurs plates-formes, nous avons obtenu des résultats nouveaux qui permettent de progresser dans la résolution et/ou dans la compréhension de certains des problèmes que nous étudions. Ces résultats concernent : les entrepôts persistants répartis (§6.2), l'architecture et le dimensionnement de systèmes de caches Web répartis (§6.3), la localisation des réplicats (§6.4), le partage des données sur des machines faiblement connectées (voir §6.5), et les systèmes adaptatifs (§6.6).*

6.2 Mémoire répartie persistante et partagée

Participants : Xavier Blondel, Nicolas Richer, Alexandru Salcianu, Marc Shapiro.

Mots clés : mémoire répartie, mémoire virtuelle partagée, PerDiS, ramasse-miettes.

Cette année a vu la concrétisation, par la mise en œuvre dans PerDiS, de nombreux efforts de recherche, en cours l'an dernier, et ayant pour thème général la gestion de la mémoire. Les deux aspects étudiés ont été les problèmes d'allocation d'une part, et de ramasse-miettes réparti grande échelle et tolérant aux fautes d'autre part.

Les problèmes d'allocation et, plus généralement, de gestion des accès concurrents aux méta-données ont été résolus par l'utilisation d'un mécanisme dit de transactions-systèmes, qui est décrit dans [9].

Des travaux théoriques ont été menés sur le ramasse-miettes à grande échelle et tolérant aux fautes. Notamment, une nouvelle interprétation du ramasse-miettes de Goldberg, dit de *comptage de références par générations* [Gol89], a été identifiée comme une solution possible au problème de la tolérance aux fautes. Ses limitations ont fait l'objet d'un travail purement théorique initialement, et son intégration à la plate-forme PerDiS va nous permettre de vérifier nos théories. La mise en œuvre ainsi que son évaluation constituent des *deliverables* du projet PerDiS.

En plus des avancées dans le domaine des algorithmes de ramasse-miettes, l'année 1999 a vu aussi la concrétisation des efforts entrepris depuis le début de l'année 1998 afin d'analyser les caractéristiques des mémoires d'objets. La première version de la plate-forme de mesure, `Analysis` (voir §5.3), a été utilisée pour caractériser le comportement d'une première application : *Persistent Xfig*, l'outil de dessin vectoriel Xfig porté sur la mémoire persistante PerDiS. Les résultats de cette étude ont fait l'objet d'un *deliverable* du projet PerDiS [16].

Cette première utilisation de la plate-forme `Analysis` a mis en évidence un certain nombre de problèmes dans la mise en œuvre qui ont été progressivement corrigés durant l'année. Ces différentes améliorations ont permis de réduire le temps nécessaire pour une même analyse d'un facteur proche de 100. Dans le même temps, il a été mis en œuvre un outil permettant de reproduire les allocations/dés-allocations mémoire en utilisant différentes politiques. Lorsqu'il est combiné avec l'outil de collecte des caractéristiques des sites Web qui a également été développé durant l'année, cet outil permettra de fournir des résultats sur les mémoires d'objets de très grande taille. Les résultats font l'objet d'un *deliverable* du projet PerDiS ; ils seront ensuite étendus et soumis à publication au début de l'année prochaine.

6.3 Architecture et dimensionnement d'infrastructures de caches Web pour Intranets décentralisés

Participants : Guillaume Pierre, Mesaac Makpangou.

Mots clés : Systèmes de caches, configuration de systèmes, optimisation de systèmes, Relais.

Les systèmes de caches Web mis en place par les administrateurs sont rarement optimaux. En effet, de trop nombreux paramètres influent sur la performance du système pour pouvoir être pris en compte lors d'une installation au jugé. Pour concevoir rationnellement un système de caches, il faut définir une notion de qualité de service et considérer les spécificités du trafic à gérer et des réseaux qui le supporteront. Les moyens à mettre en oeuvre sont également très variés: il faut décider d'une architecture du système; il faut choisir des politiques de remplacement, de cohérence et de coopération; enfin, il faut dimensionner les caches, les durées de vie des documents, etc.

[Gol89] B. Goldberg, «Generational Reference Counting: A Reduced-Communication Distributed Storage Reclamation Scheme», *in : Programming Languages Design and Implementation, SIGPLAN Notices*, 24(7), SIGPLAN, ACM Press, p. 313–321, Portland OR (USA), juin 1989.

L'approche proposée repose sur la simulation basée sur des traces. La première étape consiste à définir l'objectif de l'optimisation grâce à une fonction de coût. Cette fonction permet de quantifier la qualité de service qu'offre tel ou tel système de caches. Pour évaluer la fonction de coût d'un système, on commence par collecter des informations sur les schémas d'accès des utilisateurs; ensuite un simulateur permet de rejouer *in vitro* les traces collectées, en interposant le système de caches à évaluer entre les clients et les serveurs.

Une étude statistique sur les traces permet de dégager un petit nombre d'architectures de systèmes susceptibles d'offrir de bonnes qualités de service. Une fois ces architectures définies, le problème se ramène à la minimisation d'une fonction à plusieurs variables; des algorithmes d'optimisation permettent de déterminer les valeurs idéales des variables, et donc de définir le système optimal.

Ce travail a fait l'objet de la soutenance de thèse de Guillaume Pierre [6].

6.4 Localisation des miroirs par coopération entre organismes

Participants : Neilze Dorta, Ian Piumarta, Pierre Sens.

Mots clés : Localisation, réplication, large échelle..

La localisation des miroirs sur Internet est un problème difficile : (i) les miroirs d'un même serveur ont des noms différents ; (ii) il n'existe pas aujourd'hui un service sur l'Internet possédant une connaissance de l'ensemble de miroirs et capable de rediriger les requêtes des utilisateurs vers les miroirs les plus appropriés, c'est-à-dire ceux satisfaisant le mieux les attentes des utilisateurs concernant la fraîcheur des données et les performances.

Nous proposons, $ARÃ$, un système de localisation des miroirs par coopération entre *groupes d'organismes*. Nous entendons par organisme un groupe de sites coopérants, liés par un intérêt commun et géographiquement proches. Le groupe d'organismes coopérants sont eux aussi liés par des intérêts communs.

Notre solution se divise en deux parties : le support de localisation transparente des miroirs et le protocole de coopération inter-groupe pour la découverte des miroirs. La première partie sert à rendre transparente, vue de l'utilisateur, la localisation des miroirs; la deuxième partie permet de nourrir les bases de données de localisation de miroirs appartenant à chaque organisme.

Du point de vue de ses utilisateurs, $ARÃ$ [11] est un *proxy* de localisation. $ARÃ$ assure la redirection automatique des requêtes de ses utilisateurs vers les miroirs les plus "appropriés". $ARÃ$ est mis en œuvre par un ensemble de serveurs de localisation qui coopèrent pour la découverte des miroirs disponibles sur Internet.

La mise en œuvre du système $ARÃ$ est en cours ainsi que son évaluation. Ce travail devrait aboutir en 2000 à la soutenance de thèse de Neilze Dorta.

6.5 Cadmium : un système support pour le partage de données personnelles sur des machines faiblement connectées

Participant : Aline Baggio.

Mots clés : mobilité, système réparti, réplication, adaptation..

L'informatique mobile présente nombre de limitations dues aux déconnexions et reconnexions dans des environnements réseaux différents, des communications sans fil peu rapides, voire non fiables, et des ressources matérielles réduites. Un support système pour l'informatique mobile se doit de masquer ces limitations à l'utilisateur. Le système doit s'adapter aux changements de l'environnement afin de tirer profit de toute ressource disponible, et garantir ainsi la disponibilité des données en maintenant des copies locales.

Nous décrivons ici le système Cadmium et son support pour l'adaptation et la réplication. Cadmium permet au système et aux applications de tenir compte des changements du contexte d'exécution, et ce grâce à une surveillance de l'environnement et à une notification des événements importants. Un ensemble de mécanismes flexibles permet ensuite au système et aux applications de s'adapter dynamiquement à ces changements.

Cadmium fournit des mécanismes basés sur des objets répartis et des références, les Cd SSP Chains, grâce auxquelles une référence garde sa signification en dépit des déconnexions. Sous contrôle applicatif, un mécanisme de liaison flexible rend possible la redirection des références vers le "meilleur" objet cible. Par exemple, une référence vers un objet ayant migré est redirigée vers le nouvel emplacement; une référence vers un objet répliqué est redirigée vers la copie la moins chargée.

La réplication de données partagées pose des problèmes d'accès, de cohérence et de conflits de mise à jour. En l'absence d'une solution optimale universelle, Cadmium permet aux applications de contrôler la gestion des objets distribués, et ce grâce à des stratégies chargeables à la demande. Chaque objet ou groupe d'objets peut utiliser ses propres stratégies. Ceci permet à une application de s'adapter aux changements de son environnement, et de choisir dynamiquement la forme d'adaptation la plus pertinente.

Ce travail a fait l'objet de la soutenance de thèse d'Aline Baggio [5].

6.6 Machine virtuelle virtuelle

Participants : Carine Baillarguet, Bertil Folliot, Ian Piumarta.

Mots clés : adaptabilité, interopérabilité, machine virtuelle virtuelle, spécialisation..

La Machine Virtuelle Virtuelle (MVV) vise à fournir un environnement d'exécution modulable et spécialisable. L'environnement de base est indépendant des langages de programmation, et supporte l'exécution d'une variété de langages «bytecodés» sous la forme de «MVlets» chargeables dynamiquement. Chaque MVlet définit le jeu d'instructions de la machine virtuelle, les fonctions de bases, le modèle de représentation de la mémoire et des objets, etc. pour un langage de haut niveau particulier. Elle fonctionne en transformant dans sa représentation interne de bas niveau tous les éléments spécifiques à un langage. Cette transformation permet dans le même temps d'optimiser le code et de vérifier sa sûreté. Un résultat significatif de cette transformation est que les programmes (ou composants logiciels) écrits dans des langages différents peuvent inter-opérer. La compilation des définitions d'une «MVlet» permet de réaliser une machine virtuelle modifiable à la demande pour les besoins des applications embarquées ou réparties.

En 1999, nous avons réalisé un premier prototype «réaliste» de MVlet sur une application représentative susceptible de profiter d'un environnement du type de la MVV. Ce prototype a permis la consolidation de la Machine Virtuelle Récursive (MVR), en affinant certains de ses mécanismes, et démontre clairement la simplicité et la faisabilité de la description de machines virtuelles (même si cette description est «restreinte» au jeu d'instructions et au mécanisme de traitement des paquets d'un réseau actif).

Dans le cadre d'un contrat CTI CNET en cours de signature, nous avons commencé la conception [7] de la Machine Virtuelle Adaptative. Elle représente la deuxième étape (après la MVR) vers la MVV, et permet l'adaptation d'un modèle de machine virtuelle générique à celle décrite par une MVlet. L'adaptation ne se fait plus au niveau du jeu d'instructions et de primitives, construit à partir de celui existant (et s'exécutant) dans le cas de la MVR, mais au niveau de la machine virtuelle elle même qui est instanciée et modifiée pour correspondre à la machine virtuelle décrite.

L'aspect système d'exploitation commence également à être développé, avec la conception du noyau et du mécanisme d'adaptation du système d'exploitation. Ces éléments sont fortement couplés avec les techniques langages développées jusqu'ici, et permettent de bénéficier au niveau du cœur du système des propriétés de l'environnement d'exécution, que ce soit au niveau des performances ou de l'adaptabilité [8].

7 Contrats industriels (nationaux, européens et internationaux)

7.1 Panorama

Résumé : *Nous avons un contrat de recherche avec le CNET portant sur l'architecture et le dimensionnement des caches Web coopérants (§7.2). Nous participons aussi à l'action WebTools de Dyade (§7.3) et avons reçu (jusqu'à juin) un financement du consortium World-Wide Web (W3C) (§5.5) pour une thèse sur l'amélioration de la qualité de service sur le Web. Enfin, une proposition labellisée RNRT en collaboration avec le CNET, l'université Paris 6 et le projet Compose de l'IRISA est en cours de démarrage (voir §7.4).*

7.2 Contrat CNET «Architecture et dimensionnement des caches Web coopérants»

Relais est une infrastructure configurable d'accès au Web, destinée aux organismes décentralisés. Il vise à améliorer la cohérence des observations ainsi que les performances du Web. Il est constitué de caches et de miroirs. L'architecture et le dimensionnement de Relais, pour une entreprise donnée, prennent en compte les caractéristiques du trafic et du réseau, la qualité du service attendue par les utilisateurs, ainsi que le coût.

Le prototype actuel a toutefois des limitations. De plus, le dimensionnement est pour le moment laissé à la charge de l'administrateur. Le premier objectif de ce contrat est donc la consolidation du système Relais, c'est-à-dire : (1) une ré-ingénierie plus modulaire et plus robuste de Relais ; (2) le support des miroirs, du pré-chargement et de la technologie *push* ; (3) le développement d'un outil d'aide à la configuration de Relais permettant de déterminer la

bonne architecture de coopération entre les composants ainsi que le bon dimensionnement de chaque composant. Le deuxième objectif est l'évaluation de l'adéquation de l'offre Relais pour des cas réels, en l'occurrence pour l'INRIA et le CNET.

En 1999, l'accent a été mis sur l'évaluation et la comparaison de Relais avec d'autres protocoles de caches coopérants. Nous avons aussi travaillé sur des extensions de Relais pour la localisation des miroirs.

7.3 Action WebTools de Dyade

Le but de l'action WebTools de Dyade est de développer des outils pour améliorer la qualité du travail coopératif au-dessus du Web. Cette année, nous avons poursuivi le développement du système de caches coopérants. Nous avons aussi poursuivi la spécification d'un système d'aide de recherche d'information, exploitant l'expertise des différents utilisateurs du système.

7.4 Contrat RNRT Phénix : Noyau d'infrastructure répartie adaptable

Les technologies logicielles de la répartition sont parvenues récemment à un bon degré de maturité, notamment avec l'apparition de plates-formes d'exécution réparties conformes aux spécifications CORBA et l'émergence de la technologie Java. Malgré tout, la technologie Java/CORBA possède encore de nombreuses insuffisances, notamment pour la mise en oeuvre d'applications réparties ayant de fortes contraintes de qualité de service (contraintes de taille, contraintes temps réel, contraintes de sûreté de fonctionnement) comme celles survenant dans les réseaux d'information.

Le projet Phénix se propose de jeter les bases d'une nouvelle infrastructure logicielle répartie, qui permette de lever ces insuffisances. Le projet a pour objectif de développer un **noyau d'infrastructure répartie adaptable**, c'est-à-dire qui offre la possibilité de **se configurer** et de **s'adapter** automatiquement ou semi-automatiquement aux besoins et aux contraintes de qualité de service des applications. Le projet s'appuie sur une **approche réflexive**, consistant à construire une infrastructure ouverte à l'introspection et à la modification dynamique par les applications qui l'exploitent. Il entend combiner dans son approche plusieurs technologies logicielles émergentes : **exo/nano-noyaux de systèmes d'exploitation, micro-ORB, techniques d'évaluation partielle pour langages dédiés, machine virtuelle adaptable**.

8 Actions régionales, nationales et internationales

8.1 Actions nationales

Le projet SOR collabore, au niveau national, avec les équipes de recherche en systèmes répartis dans le cadre du GDR ARP (Architecture, Réseaux et Systèmes, Parallélisme), animé par Bertil Folliot et Michel Riveill.

8.2 Actions financées par la Commission Européenne

8.2.1 Le projet PerDiS

Une part importante de notre activité en 1999 concerne le contrat PerDiS.

PerDiS est un projet ESPRIT LTR qui devait initialement se dérouler de décembre 1996 à décembre 1999. Cette durée a été étendue jusqu'à mars 2000 pour permettre de finaliser le portage des applications réelles au-dessus de PerDiS. Ses partenaires sont académiques (QMW, Angleterre ; INRIA-SIRAC et INRIA-SOR ; INESC, Portugal) et industriels (IEZ, Allemagne ; CSTB, France). En outre, ses parrains industriels sont Sun (Chorus Systèmes), DEC External Research, IONA et Bull.

Le projet propose l'abstraction *d'Entrepôt Persistant Réparti* (EPR). Une application travaille dans l'EPR comme en mémoire normale. Toute donnée allouée dans l'EPR, et accessible, devient automatiquement persistante et peut être partagée par tous les processus du réseau. Le système gère automatiquement et efficacement la mise en cache sur le site accédant à une donnée, le ramasse-miettes, le stockage sur disque, la tolérance aux pannes, le contrôle de concurrence. Cette nouvelle abstraction facilite l'écriture d'applications réparties, et le portage au réparti d'applications centralisées. Nous testons cette capacité grâce à des applications réelles et d'envergure, dont un outil de CAO coopérative pour l'architecture et le bâtiment.

Le projet SOR joue un rôle très important dans PerDiS. L'INRIA est le coordinateur et contractant principal du projet. Nous sommes responsables de l'intégration de code et le constructeur final de la plate-forme PerDiS. Enfin, nous devons assurer la diffusion des résultats obtenus dans PerDiS vers les chercheurs et les industriels.

Les recherches dans le cadre de PerDiS sont décrites en §4.3, §5.2 et §6.2.

8.2.2 Réseaux et groupes de travail internationaux

Nous participons au réseau d'excellence Cabernet, ainsi qu'au groupe de travail Broadcast-WG.

Broadcast-WG est un «Working Group» européen de recherche sur les systèmes répartis de grande échelle comprenant : Newcastle, Grande Bretagne ; Université de Lisbonne, Portugal ; Université de Bologne, Italie ; École Polytechnique de Lausanne, Suisse ; et INRIA-Sirac, INRIA-Solidor et INRIA-Sor. La plupart des travaux de l'équipe sont inclus dans le champ d'intérêt de Broadcast-WG.

8.3 Accueils de chercheurs étrangers

Nous avons invité deux orateurs pour le colloquim : John Crowcroft en mars de Imperial College (Grande-Bretagne) et M. Satyanarayanan du CMU (USA). Satyanarayanan a aussi participé au jury de thèse d'Aline Baggio.

Nous avons aussi organisés plusieurs séminaires dont les principaux intervenants étrangers en 1999 ont été : Francisco Torés Rojas, College of Computing, Georgia Institute of Technology (USA) ; Peter Druschel, Rice University (USA) ; et Vinny Cahill, Trinity College (Irlande) qui a aussi participé au jury de thèse d'Aline Baggio.

9 Diffusion de résultats

9.1 Animation de la communauté scientifique

- Marc Shapiro est Vice-président (jusqu'en juin) du «Special Interest Group on Operating Systems» (SIGOPS) de l'ACM et président de la section française de l'ACM SIGOPS.
- Bertil Folliot est Vice-président de la section française de l'ACM SIGOPS et co-responsable avec Michel Riveill du thème Systèmes et Applications Réparties du GDR ARP (Architecture, Systèmes et Réseaux, Parallélisme).
- Marc Shapiro était membre du comité de programme de DISC'99.
- Mesaac Makpangou était membre du comité de programme de ERSADS'99.
- Xavier Bmondel et Vincent Bouthors étaient les co-organisateurs des premières journées Thèmes Emergents de l'ASF-SIGOPS (Antènne française de SIGOPS) consacrées aux services avancés sur le Web.

9.2 Enseignement universitaire

DEA SI (Systèmes Informatiques, Paris 6). Systèmes Répartis Avancés, janvier–mars : Responsables : Bertil Folliot, Marc Shapiro.

ENST (École Nationale Supérieure des Télécommunications), troisième année. Cours sur le partage de données : Marc Shapiro.

ESIEE , (5ème année). Cours sur les algorithmes répartis : Mesaac Makpangou.

FIIFO (Formation d'Ingénieur en Informatique de la Faculté d'Orsay), formation continue. Le World-Wide Web : Guillaume Pierre.

9.3 Autres enseignements

ISTM (Institut Supérieur de Technologie et Management). Cours systèmes et algorithmes répartis : Mesaac Makpangou.

CNAM (Conservatoire National des Arts et Métiers – Versailles). Introduction aux systèmes répartis : Guillaume Pierre, Olivier Dedieu, Mesaac Makpangou.

9.4 Participation à des colloques, séminaires, invitations

9.4.1 Participation à des colloques

CFSE99, 1ere Conférence Française en Systèmes d'Exploitation, Rennes. Présentation d'une communication : Xavier Blondel [9]. Autres participants : Carine Baillarguet, Neilze Dorta, Marc Shapiro.

JCS'99 , Journées Jeunes Chercheurs en Systèmes, Rennes (France). Participantes avec deux présentations : Carine Baillarguet [8] et Neilze Dorta [11].

OSDI'99, 3rd Symposium on Operating System Design and Implementation, New Orleans, Louisiana, USA. Participant : Xavier Blondel avec une présentation de PerDiS dans la session WIP (Work In Progress).

ICDCS'99 International Conference on Distributed Computing Systems, Austin, Texas (USA). Participant avec présentation d'un papier [12] : Mesaac Makpangou.

USITS'99 2nd Usenix Symposium on Internet Technologies and Systems, Boulder, Colorado (USA). Participant : Simon Patarin.

SOSP'99, Symposium on Operating Systems and Principles, New Orleans (USA). Participants : Carine Baillarguet et Simon Patarin.

9.4.2 Réunions de contrat

Revue de projet PerDiS, Plus d'une dizaine de reunions physiques ont eu lieu chez les différents partenaires du projet. De nombreuses réunions téléphoniques ont été organisées pour le suivi et la coordination du projet.

Les principaux participants à ces diverses réunions sont : Marc Shapiro (cordinateur du projet) et Xavier Blondel (responsable de l'intégration de différents logiciels développés par les partenaires).

Les autres participants (lorsque l'ordre du jour exige leur présence) sont : Nicolas Richer, Alexandru Salicianu, Pierre Albertin et Ngock-Koi Tô.

Relais Réunion CTI CNET à Caen, mai : Mesaac Makpangou, Guillaume Pierre, Christian Khoury, Neilze Dorta.

Phenix Plusieurs réunions pour la proposition RNRT Phenix : Mesaac Makpangou, Bertil Folliot, Carine Baillarguet et Ian Piumarta.

9.4.3 Séminaires

Inria Sophia-Antipolis, Séminaire Inria organisé par Mistral. Présentation de Saperlipopette : Guillaume Pierre et Mesaac Makpangou.

Groupe de travail PRC I3, mars, Université Saint-Quentin en Yvelines. «La mémoire persistante répartie PerDiS et ses applications», Marc Shapiro.

Groupe de travail PRC I3, mars, Université Saint-Quentin en Yvelines. «Relais : Un système de caches web distribués pour des organismes décentralisés», Mesaac Makpangou.

Journées ASF-SIGOPS, Services avancés sur le Web, 7 mai à Paris. Participants avec présentation d'un exposé: Guillaume Pierre et Olivier Dedieu. Co-organisateurs : Vincent Bouthors et Xavier Blondel ; autres participants : Marc Shapiro, Mesaac Makpangou, Nicolas Richer.

University of Brandeis, juin, Boston (USA). «Architecture and Dimensioning Web Caching Infrastructures for Decentralized Intranets», Mesaac Makpangou.

10 Bibliographie

Ouvrages et articles de référence de l'équipe

[1] P. Ferreira, M. Shapiro, «Larchant: Persistence by Reachability in Distributed Shared Memory through Garbage Collection», *in : Proc. 16th Int. Conf. on Dist. Comp. Syst. (ICDCS)*, Hong Kong, mai 1996, `http://www-sor.inria.fr/publi/LPRDSMGC_icdcs96.html`.

[2] M. Makpangou, Y. Gourhant, J.-P. Le Narzul, M. Shapiro, « Fragmented Objects for Distributed Abstractions », *in : Readings in Distributed Computing Systems*, T. L. Casavant et M. Singhal (éditeurs), IEEE Computer Society Press, juillet 1994, p. 170–186.

[3] D. Plainfossé, M. Shapiro, « A Survey of Distributed Garbage Collection Techniques », *in : Second Closed BROADCAST Workshop*, Broadcast Basic Research Action, p. 211–249, Bruxelles (Belgique), novembre 1994.

[4] M. Shapiro, « Structure and Encapsulation in Distributed Systems: the Proxy Principle », *in : Proc. 6th Intl. Conf. on Distributed Computing Systems*, IEEE, p. 198–204, Cambridge, Mass. (USA), mai 1986.

Thèses et habilitations à diriger des recherches

[5] A. Baggio, *Adaptable and Mobile-Aware Distributed Objects*, thèse de doctorat, Université Pierre et Marie Curie – Paris VI, Paris, France, juin 1999, `http://www-sor.inria.fr/publi/baggio_thesis99.html`.

[6] G. Pierre, *Architecture et dimensionnement d'infrastructures de caches Web pour Intranets décentralisés*, thèse de doctorat, Université d'Evry-val d'Essonne, Evry (France), juin 1999, `http://www-sor.inria.fr/publi/pierre_thesis99.html`.

Communications à des congrès, colloques, etc.

[7] C. Baillarguet, I. Piumarta, « An highly-configurable, modular system for mobility, interoperability, specialization, and reuse », *in : 2nd ECOOP Workshop on Object-Orientation and Operating Systems*, Lisboa, Portugal, juin 1999.

[8] C. Baillarguet, «MVV: langage et système, plus qu'un mariage de raison», *in : Proceedings of the Journée des Jeunes Chercheurs en Systèmes (JCS99)*, Rennes, France, juin 1999.

[9] X. Blondel, « Transactions système pour la manipulation des méta-données dans l'entrepôt persistant réparti PerDiS », *in : Proceedings of the First ACM/SIGOPS French Conference on Operating Systems (CFSE'1)*, juin 1999. `http://www-sor.inria.fr/publi/TSMMDEPRP_cfse99.html`.

[10] V. Bouthors, O. Dedieu, « Pharos, a Collaborative Infrastructure for Web Knowledge Sharing», *in : Research and Advanced Technology for Digital Libraries. Third European Conference, ECDL '99, Paris, France, September 22–24, 1999: Proceedings*, S. Abiteboul, A.-M. Vercoustre (éditeurs), *Lecture Notes in Computer Science*, Springer-Verlag Inc., p. 215–233, 1999.

[11] N. Dorta, « ARÃ : Un système de gestion des miroirs sur Internet », *in : Proceedings of the Journée des Jeunes Chercheurs en Systèmes (JCS99)*, Rennes, France, juin 1999.

[12] M. Makpangou, G. Pierre, C. Khoury, N. Dorta, «Reliable Directory Service for Weakly Consistent Replicated Caches », *in : Proceedings of the 19th IEEE International Conference on Distributed Computing Systems (ICDCS'99)*, juin 1999. `http://www-sor.inria.fr/publi/RDSWCDC_icdcs99.html`.

Rapports de recherche et publications internes

[13] V. Bouthors, O. Dedieu, « Pharos, a Collaborative Infrastructure for Web Knowledge Sharing », *Rapport de Recherche n° RR-3679*, Institut National de Recherche en Informatique et Automatique, mai 1999, `http://www-sor.inria.fr/publi/PCIWKS_rr3679.html`.

[14] S. Patarin, M. Makpangou, «Pandora : A Flexible Network Monitoring Platform», *Research Report n° RR-3834*, Institut National de Recherche en Informatique et Automatique, 1999, `ftp://ftp.inria.fr/INRIA/publication/publi-ps-gz/RR/RR-3834.ps.gz`.

[15] S. Patarin, «Pandora : un système de collecte de traces du trafic Web de communautés d'utilisateurs réparties», *Rapport de Recherche n° RR-3743*, Institut National de Recherche en Informatique et Automatique, juillet 1999, `http://www.inria.fr/RRRT/RR-3743.html`.

[16] N. Richer, « Meazuring and analyzing Persistent Xfig Memory Behaviour », *rapport de recherche n° tc3.2a*, PerDiS Consortium, March 1999, http://www.perdis.esprit.ec.org/deliverables/docs/T.C.3.2/tc32a.ps.