INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet OASIS

Objets Actifs, Sémantique, Internet et Sécurité

Sophia Antipolis

THÈME 2A

Table des matières

OASIS *est un projet commun à l'INRIA, au CNRS et à l'université de Nice-Sophia Antipolis, en collaboration avec l'institut Eurécom.*

1 Composition de l'équipe

Responsable scientifique

Isabelle Attali [CR, Responsable formation UR Sophia Antipolis jusqu'au 01/09/99]

Responsable permanent

Bernard Serpette [CR, depuis le 01/02/99]

Assistance administrative

Sandrine Boute [pour la gestion du personnel, à temps partiel]
Maryse Renaud [depuis le 01/09/99, Assistante DYADE et contact OASIS]

Personnel Inria

Gilles Barthe [CR, depuis le 01/10/99]
Francis Montagnac [IR, à temps partiel]
Didier Parigot [CR, depuis le 01/09/99]

Personnel UNSA

Françoise Baude [Maitre de Conférence, UNSA]
Denis Caromel [Professeur, UNSA]

Personnel BULL

Franck Chalaux [Ingénieur BULL, depuis le 06/09/99, action SmartTools]

Ingénieurs experts

Pascal Degenne [depuis le 02/11/99, action SmartTools]
Alexandre Fau [depuis le 08/10/99, action SmartTools]
Stéphane Jasinski [depuis le 01/06/99, action SmartTools]

Chercheur invité

Gilles Barthe [Université du Minho, du 01/04/99 au 30/06/99]

Chercheurs doctorants

Carine Courbis [Allocataire MESR, deuxième année]
Simao De Sousa [Boursier Portugal, depuis le 01/10/99]
Romain Guider [Allocataire MESR, quatrième année]
Fabrice Huet [Allocataire MESR + Moniteur UNSA, depuis le 01/10/99]
Christophe Roudet [Allocataire MESR, soutenance de thèse le 28/05/99]
Marjorie Russo [Allocataire MESR, troisième année]
David Sagnol [Allocataire MESR, quatrième année]
Julien Vayssière [Boursier Région Entreprise, UNSA, depuis le 01/11/99]

Chercheurs post-doctorants

Shen Wei Yu [Boursier CIES, jusqu'à décembre 1999]
Henrik Nilsson [depuis le 01/03/99, financement suédois]

Stagiaires

Alexandre Fau [DEA Informatique UNSA, du 15/01/99 au 30/09/99]
Fabrice Huet [DEA Réseaux et Systèmes Distribués, UNSA, du 01/03/99 au 30/09/99]
Stéphane Jasinski [ESSI, UNSA, du 19/04/99 au 31/05/99]
Arnaud Poizat [ESSI, UNSA, du 19/04/99 au 30/09/99]
Ludovic Henrio [Polytechnique, du 06/04/99 au 22/09/99]
Florian Doyon [Alternance IUT, du 01/04/99 au 30/09/99]
Valéry Beaujan [DEA SPP, Paris VII, du 25/05/99 au 11/07/99]
Alexandre Bergel [Licence UNSA, du 15/06/99 au 15/09/99]
David Teller [première année Magister, ENS Lyon, du 01/06/99 au 11/07/99]
Sébastien Villemot [Licence-Maitrise ENS Ulm, du 15/06/99 au 15/08/99]

Collaborateurs extérieurs Eurécom

Refik Molva [Professeur associé]
Yves Roudier [Assistant]

2 Présentation et objectifs généraux

Dans le cadre des applications réparties (réseaux Internet et intranets, cartes à puce et terminaux), l'objectif du projet est de proposer des principes fondamentaux, des techniques et des outils pour la construction, l'analyse, la validation, la vérification et la maintenance de systèmes fiables.

Le projet rassemble deux domaines d'expertise clairement identifiés :

- sémantique, environnements, compilation et analyses statiques,
- programmation à objets répartie.

De plus, les collaborateurs d'Eurécom apportent des compétences reconnues internationalement en sécurité des réseaux et des systèmes distribués.

A partir de ces compétences, l'objectif du projet est de créer une synergie sur les thématiques suivantes :

- environnement fondé sur la sémantique pour le développement, l'analyse et la vérification d'applications réparties et communicantes liées à l'Internet (par exemple Java, Java Card) ;
- construction de bibliothèques facilitant la programmation et la maintenance d'applications multi-threadées, distribuées, sécurisées, en particulier pour les applications collaboratives et le commerce électronique.

D'autre part, l'avènement d'Internet a créé la nécessité d'étendre, de manière fondamentale, les conceptions existantes de mobilité et sécurité. Nous disposons de compétences, d'outils et de méthodes qu'il nous parait intéressant de mettre à profit dans un domaine d'application devenu majeur. Plus précisément, nous visons les domaines d'applications suivants : applications embarquées, carte à puce, commerce électronique, télécommunications, téléphonie mobile.

3 Fondements scientifiques

3.1 Spécifications, environnements, analyses et transformations

Mots clés : sémantique, environnement, méthodes formelles et preuves, analyse de programme, fiabilité du logiciel, sécurité.

Participants : Isabelle Attali, Denis Caromel, Carine Courbis, Franck Chalaux, Pascal Degenne, Alexandre Fau, Ludovic Henrio, Stéphane Jasinski, Francis Montagnac, Henrik Nilsson, Didier Parigot, Christophe Roudet, Marjorie Russo.

Depuis les débuts de l'informatisation des tâches, l'utilisateur (du plus novice au plus expert) a toujours eu besoin d'aide pour mettre au point ses programmes, manipuler une banque de données, construire un circuit imprimé ou encore concevoir un plan d'architecte. L'existence des termes comme CAO (Conception Assistée par Ordinateur), EAO (Enseignement Assisté par Ordinateur), PAO (Publication Assisté par Ordinateur), ou dans un domaine plus proche, CASE (Computer-Aided Software Engineering), montre bien l'état des besoins.

Les domaines d'utilisation d'outils d'aide sont nombreux et variés : les interfaces homme-machine, la programmation, les aspects distribution et configuration de systèmes, les environnements de développement de preuves, la gestion de librairies, la simulation et l'évaluation de performance, etc.

Nous sommes donc convaincus de l'utilité d'environnements de développement dans lesquels le programmeur sera guidé dans la mise au point de ses applications, par des outils interactifs, graphiques, utilisant la sémantique du langage. A ce titre, Java Card est un bon exemple par la taille réduite des applications, et permet d'envisager le traitement de problèmes insolubles dans le cas plus général d'applications contenant des millions de lignes de code.

Nous nous appuyons sur la Sémantique Naturelle (formalisme issu de la sémantique opérationnelle structurelle et de la déduction naturelle). Une spécification est un ensemble de règles

d'inférence qui décrivent comment déduire une conclusion à partir de prémisses. Les règles décrivent le comportement des constructeurs du langage manipulé. L'approche structurelle de la méthode apparait dans la forme des séquents, qui ont généralement un *sujet*, terme d'une syntaxe abstraite. Nous avons étudié plusieurs classes de langages de programmation, en nous spécialisant sur les langages à objets (Eiffel, Java) [2, 1].

Ces environnements gagnent à être associés à des outils d'analyses statiques et de transformations qui assurent une certaine aide à la programmation et à l'optimisation de programmes [10].

Enfin le contexte (Internet, commerce électronique, cartes à puce et sécurité) nous pousse à nous interesser à des outils de vérification qui seront particulièrement utiles aux programmeurs pour garantir qu'une politique de sécurité est respectée (identification, intégrité, confidentialité), par exemple, que tel programme gardera des données intègres en cas de retrait inopiné de la carte.

Notre travail sur les transformations de programmes comme outils de généricité, nous a amené à comparer différentes techniques de transformations issues de divers styles de programmation : fonctionnelle pour la technique de déforestation et la programmation polytypique, à objets pour les *visitor patterns* ou les *tree traversals* et enfin les grammaires attribuées [9, 7].

Notre travail sur la formalisation des sémantiques des langages nous a amené à proposer des extensions aux systèmes de types existants. Nous avons tout particulièrement étudié le sous-typage par constructeurs [3], qui permet de formaliser d'une manière élégante de nombreux exemples de sémantique naturelle.

3.2 Programmation objet, concurrence et répartition

Mots clés : programmation à objets répartie, analyse de programme, code mobile, collecticiels, communication de groupe, concurrence, distribution, synchronisation, sécurité.

Participants : Isabelle Attali, Françoise Baude, Alexandre Bergel, Denis Caromel, Florian Doyon, Romain Guider, Fabrice Huet, Stéphane Jasinski, Francis Montagnac, Arnaud Poizat, David Sagnol, Bernard Serpette, Julien Vayssière.

Le paradigme objet, même s'il date des années 70, reste un des aspects des langages de programmation les plus étudiés de nos jours. Ces dernières années, avec l'apparition du langage Java, on a pu observer une recrudescence de l'activité autour de la méthodologie objet. Autant le concept se veut universel, autant les variations des modèles et leurs implémentations possèdent des propriétés spécifiques souvent mal définies : sous la même terminologie objet, se retrouvent des thèmes particuliers comme l'héritage (simple ou multiple), le sous-typage, la surcharge, etc.

D'autre part, les aspects programmation concurrente (en particulier, multi-threading, accès concurrents) viennent apporter un degré supplémentaire de complexité. Le mélange de l'ensemble de ces traits peut faire apparaître des cas où la spécification du langage reste floue et pour lesquels la construction d'applications et leur mise au point restent délicates.

Le langage Java peut être également abordé comme un langage très prometteur pour la programmation distribuée sur un réseau ; l'arrivée de Java a laissé espérer que l'on pourrait distribuer des applications haute-performance sur le réseau Internet, premier pas vers le méta-

computing. Malheureusement, les composants Java standards tels que RMI (Remote Method Invocation) n'aident en fait pas à construire de manière transparente des applications séquentielles, multi-threadées, ou distribuées, en permettant l'exécution d'une même application sur une architecture multi-processeurs à mémoire partagée aussi bien que sur un réseau de stations de travail (intranet, Internet), ou encore n'importe quelle combinaison hiérarchique des deux.

La question est donc comment construire à partir des outils standards (par exemple threads, RMI, etc), des modèles et bibliothèques facilitant la programmation distribuée.

Nous avons développé des compétences en programmation concurrente, répartie et parallèle dans le cadre des langages à objets ; ces recherches sont menées aussi bien sur des aspects plus théoriques comme les sémantiques formelles, ou les analyses statiques et transformations pour la répartition automatique ou semi-automatique, que sur des aspects plus pragmatiques comme la conception de langages et de méthodes de programmation [6], ou la construction de bibliothèques pour le parallélisme [4, 5].

Nous avons également étudié les problèmes des supports d'exécution de ces bibliothèques [8, 13] (mise en œuvre portable, optimisations, etc).

4 Domaines d'applications

4.1 Panorama

Résumé : *Les domaines d'application du projet Oasis couvrent tous les aspects du logiciel embarqué (cartes à puces, commerce électronique, téléphonie mobile) ainsi que les aspects liés aux applications réparties et communicantes sur intranets et Internet (par exemple applications collaboratives).*

4.2 Maintenance et manipulation de programmes

La maintenance des systèmes logiciels est l'un des points critiques du cycle de vie d'un logiciel. La durée de vie de ces logiciels a augmenté ainsi que leur complexité due à des mises à jour successives, soit pour corriger des bugs ou soit pour ajouter de nouvelles fonctionnalités. Les problèmes de l'an 2000 et du passage à l'Euro sont deux exemples concrets de maintenance de logiciel. La taille et la complexité de ces systèmes logiciels rendent leur maintenance coûteuse et hasardeuse si elle n'est pas automatisée. Les systèmes de manipulation et de transformation de programmes, cantonnés jusqu'ici dans des applications de taille réduite, semblent désormais adaptés pour répondre à ces nouveaux besoins.

4.3 Logiciels sécurisés pour le Commerce Electronique

Plusieurs domaines d'applications du réseau Internet et des cartes à puces, dont le commerce électronique, demandent la protection de données précieuses (numéros de compte en banque, codes confidentiels, etc) qui, en un moment déterminé, seront disponibles sur des plates-formes reliées physiquement à un vaste réseau. Ici le besoin de sécurité maximale est réel et partagé :

– les sociétés de services (banques, administration etc.) doivent avoir la garantie qu'une application vérifie des propriétés de sécurité, définies dans une politique de sécurité glo-

bale (par exemple identification, intégrité, confidentialité, ou non-répudiation) et pourra ainsi résister à des attaques systématiques ;
- leurs clients doivent être assurés que les informations qu'ils fournissent lors d'une demande de services ne seront pas utilisées à mauvais escient ou détournées vers un tiers.

4.4 Programmation répartie, collaborative et sécurisée pour Internet

Nos champs d'application sont les systèmes collaboratifs (par exemple des systèmes d'entreprise intranet et Internet), mais également en choisissant de mettre l'accent sur les aspects liés à la sécurité, les systèmes transactionnels commerciaux, bancaires, etc.

Un des domaines d'application particulièrement représentatif est la construction et l'évolution de collecticiels (plusieurs utilisateurs distants travaillent de manière coordonnée à une même tâche) dans lesquels se posent des problèmes d'élection, de synchronisation, de répartition des calculs, etc.

5 Logiciels

5.1 Outils Interactifs Génériques SmartTools

Participants : Isabelle Attali [correspondant], Denis Caromel, Franck Chalaux, Pascal Degenne, Alexandre Fau, Stéphane Jasinski, Francis Montagnac, Didier Parigot.

Le système Centaur a été conçu pour générer des environnements interactifs dédiés à un langage de programmation à partir de spécifications formelles syntaxiques et sémantiques de ce langage. Ainsi ont pu être développés, de manière largement automatisée, des éditeurs structurés, des outils d'aide à la mise au point, des vérificateurs de types, des interprètes et des traducteurs. Une première étape a été menée avec la construction d'Aïoli (noyau développé par Laurent Théry) sur lequel nous nous appuyons pour SmartTools.

Nous souhaitons développer des outils qui pérennisent les investissements existants, c'est à dire les systèmes construits sur l'environnement Centaur actuel, tout en améliorant les fonctionnalités et les performances de l'outil actuel par un certain nombre d'innovations techniques.

Trois objectifs globaux sont d'ores et déjà identifiés :
- supporter les plates-formes Windows,
- utiliser autant que possible la technologie Java,
- supporter une intégration avec le Web et ses outils.

En termes de fonctionnalités, un objectif important consiste à rendre possible l'intégration aux IDE existants.

Après 6 mois d'existence, les résultats intermédiaires de l'action Dyade SmartTools sont les suivants :
- édition libre permettant d'écrire du texte à la volée ;
- édition structurée, dirigée par la syntaxe abstraite ;
- éditeur multi-langages ;
- gestion générique et instrumentation de l'éditeur par les actions élémentaires d'édition (menus, bouton et raccourcis clavier) ;

- gestion de la fenêtre des messages d'erreur ;
- encapsulation de l'éditeur en terme de composant ;

Les instances SmartTools en cours de réalisation sont dédiées aux langages de spécification et programmation suivants :

- les langages de spécification comme Metal (description de la syntaxe abstraite) et Ppml (description de l'affichage d'une structure abstraite) ;
- Java et Java Card comme exemples de taille réelle ;
- un ensemble de langages jouets ;

Pour plus d'information, voir `http://www.inria.fr/oasis/SmartTools`.

5.2 TrfL : un langage pour les transformations de programmes

Participants : Isabelle Attali, Didier Parigot [correspondant], Christophe Roudet.

Nous avons conçu et implanté TrfL, un langage de transformation de programmes qui contribue à résoudre certains problèmes liés à la maintenance, en particulier la restructuration de code, le portage vers de nouvelles architectures et la documentation de code (voir `http://www.inria.fr/croap/personnel/Christophe.Roudet/TrfL`). TrfL est un langage déclaratif basé sur des règles de réécriture. Nous spécifions formellement ce langage en donnant sa syntaxe, sa sémantique statique et sa sémantique dynamique. Nous présentons les deux prototypes réalisés et l'environnement comprenant différents outils facilitant la mise en œuvre des transformations. Nous avons évalué les qualités et faiblesses ainsi que les performances de ce nouveau langage [14].

Le prototype Java de TrfL (construit avec Aïoli en Java) sera intégré aux outils SmartTools.

5.3 Sémantiques et Environnements pour Java/Java Card

Participants : Isabelle Attali [correspondante], Denis Caromel, Carine Courbis, Ludovic Henrio, Henrik Nilsson, Marjorie Russo.

Nous avons spécifié la quasi-totalité de la sémantique de Java (exception faite des interfaces et du chargement dynamique des classes). Nous utilisons le système Centaur qui permet de dériver un interprète interactif fondé sur la sémantique naturelle. Les spécifications sont particulièrement lisibles, et ont l'originalité de faire cohabiter sémantique opérationnelle structurelle (*small-step*) et sémantique naturelle (*big-step*). Ce travail de spécification s'est assorti d'un travail innovant sur la mise en œuvre de l'interprète dérivé, puisque la simulation du programme Java est animée et une visualisation graphique permet de comprendre les synchronisations et les partages d'objets.

L'ensemble de ces travaux a donné lieu à un dépot à l'Agence de Protection des Logiciels sous le nom « JavaSem » (voir `http://www.inria.fr/oasis/java`).

Nous avons utilisé cette sémantique opérationnelle de Java pour décrire un modèle simulant tous les acteurs des applications Java Card (noyau pour la programmation des cartes à puce), puis programmé ce modèle en Java, ce qui permet d'obtenir un simulateur d'applets Java Card. Nous avons proposé et spécifié des extensions de la sémantique qui prennent en compte

la spécificité du modèle Java Card (pas d'activités concurrentes), et construit une interface spécifique à JavaCard (liée aux transactions entre lecteur de carte et carte).

Le simulateur permet à un développeur de construire, mettre au point et tester son application avant de charger cette application sur une carte à puce. Le test peut être soit figé dans un programme (ce qui impose une découverte manuelle des commandes acceptées par une application), soit construit de manière interactive, grâce à une analyse du source d'une application Java Card afin d'en déduire les commandes (et leur format) comprises par cette application.

Pour plus d'information, consulter la page `http://www.inria.fr/oasis/javacard`.

5.4 Bibliothèque ProActive

Mots clés : programmation objet, parallélisme et répartition, représentation Meta-Objet, metacomputing.

Participants : Françoise Baude, Denis Caromel [correspondant], Florian Doyon, Fabrice Huet, Stéphane Jasinski, Arnaud Poizat, Julien Vayssière.

Notre objectif est de permettre l'exécution d'une même application sur une architecture multi-processeurs à mémoire partagée, sur un réseau de stations de travail, sur Internet ou encore n'importe quelle combinaison hiérarchique.

Pour attaquer ce problème, nous avons développé une bibliothèque 100% Java, qui fournit des threads transparents, des objets distants, des appels asynchrones avec futurs transparents, et des mécanismes de synchronisation de haut niveau. Cette bibliothèque permet très facilement de répartir et rendre collaborative toute application écrite en java.

Afin de démontrer la puissance de cette bibliothèque, nous avons développé des applications collaboratives parallèles et distribuées qui permettent à plusieurs utilisateurs de travailler ensemble sur une scène 3D avec des mécanismes d'élection et de synchronisation : l'image de la scène est calculée par un ensemble dynamique de moteurs de rendu utilisant un algorithme de lancé de rayon. Nous avons également développé DIVA (Distributed and Interactive Virtual world in Java), une application répartie collaborative pour un monde virtuel interactif entièrement écrit en Java en utilisant Java3D, RMI, and ProActive.

Enfin, une étude récente financée par un grand constructeur informatique, a démontré que l'utilisation de ces bibliothèques permet d'économiser 30 % de code. ProActive est ainsi particulièrement adaptée aux applications réparties sur l'Internet grâce à la réutilisation de code initialement non réparti, à une synchronisation automatique et à la possibilité de faire migrer des activités d'une machine à l'autre.

Pour plus d'information, consulter la page `http://www.inria.fr/oasis/proactive`.

5.5 La bibliothèque C++//

Mots clés : programmation objet, parallélisme et répartition, représentation Meta-Objet, metacomputing.

Participants : Françoise Baude, Alexandre Bergel, Denis Caromel [correspondant], David Sagnol.

C++// est une bibliothèque pour C++ permettant de répartir une application en réutilisant le maximum du code séquentiel. La version au dessus de Nexus (bibliothèque de plus bas niveau permettant à des threads répartis d'interagir, développé au Argonne National Laboratory de Chicago et à USC) est tout à fait opérationnelle.

Une version de C++// au dessus de Globus est à présent disponible. Par rapport à Nexus, Globus permet que les différents threads s'exécutent sur des ordinateurs répartis sur différents continents, Globus se chargeant du contrôle d'accès et de l'authentification de l'utilisateur sur ces machines. Il a donc été possible, en particulier lors de démonstrations effectuées au stand INRIA à SuperComputing'99, de faire s'exécuter une application C++// en se servant en même temps de machines situées à l'INRIA Sophia-Antipolis, à l'Université de Delft, à l'Université de Southern California, au Argonne National Laboratory de Chicago.

Pour plus d'information, consulter la page `http://www.inria.fr/oasis/c++ll`.

6 Résultats nouveaux

6.1 Environnements de développement et vérification pour Java et Java Card

Participants : Isabelle Attali, Denis Caromel, Carine Courbis, Henrik Nilsson, Marjorie Russo, Bernard Serpette, Shen-Wei Yu.

Nous souhaitons, à partir d'une description formelle des sémantiques des langages Java et Java Card, dériver un ensemble d'outils pour la simulation, la mise au point, et la vérification d'applications.

Les recherches s'orientent vers des directions communes (Java et Java Card) pour ce qui concerne la simulation, et les aides à la mise au point :

- spécification complète de la sémantique dynamique du langage puis simulation de l'exécution du programme par interprétation de la sémantique ;
- fonctions de mise au point et de visualisation afin de faciliter le développement des applications.

Certains points sont spécifiques à Java ; nous avons notamment décrit une sémantique formelle des exceptions Java en Typol et développé une visualisation de ces exceptions sous Centaur. D'autre part, nous avons étudié différentes possibilités d'abstraction sur le graphe d'objets obtenu lors de l'interprétation de programmes Java et intégré une de ces possibilités (celle consistant à représenter le sous-graphe d'un thread donné) à l'environnement d'interprétation Java sous Centaur.

D'autre part, les particularités de Java Card nous entraînent sur un terrain spécifique, notamment par l'existence d'un environnement d'exécution dédié sur carte à puce, d'un lecteur de carte qui sert d'interface avec l'utilisateur, et de communications via des APDUs (Application Protocol Data Unit). La sémantique opérationnelle du langage Java Card doit donc être associée aux spécifications de ces différents éléments, et permet de dériver un environnement spécifique offrant les fonctionnalités suivantes :

- connaissance du langage Java Card (contrairement aux outils commerciaux qui s'appuient

sur des environnements classiques Java) ;

- vérifications de sémantique statique qui permettent d'assurer un degré supplémentaire dans la sureté du développement.
- simulation d'une ou plusieurs applets (avant le chargement effectif sur une carte à puce), avec séparation du contexte d'exécution. Désormais, les classes JCRE, CAD, APDU et Applet sont spécifiées en Typol (au lieu d'être écrites en Java, et donc simulées), ce qui permet de ne garder dans le programme source à simuler que le code de l'applet à mettre au point. Les descriptions sémantiques ont ainsi été complétées par l'ajout de la manipulation de tableaux ainsi qu'une partie significative des APIs Java Card.
- construction interactive des APDUs de commande ; une analyse de dépendances de données du programme source permet de déduire les APDUs de commande (et leur format) comprises par cette application, puis de transmettre ces commandes, saisies de manière interactive, au simulateur [32].
- visualisation et décodage des APDUs de commande et de réponse (transactions entre carte et lecteur).

6.2 Analyses statiques et transformations de programmes

Participants : Isabelle Attali, Gilles Barthe, Denis Caromel, Romain Guider, Didier Parigot, Bernard Serpette.

Nos travaux sur l'analyse statique des programmes à objets ont conduit à deux articles en cours de soumission.

Le premier étend la réduction statique pour un langage à objets comportant des aspects impératifs : affectations, exceptions et effets de bords (l'analyse proposée dans [10] n'était définie que pour un langage purement fonctionnel). La partie objet, via l'envoi de message, a été abstraite par une fonction *Lookup* qui, selon sa définition et les types dynamiques qu'elle reçoit en argument, peut concrétiser aussi bien l'héritage simple que multiple en passant par les fonctions génériques. Le résultat principal de cet article est de montrer que les aspects impératifs introduits dans le langage n'imposent pas d'approximation particulière. La qualité de l'analyse reste factorisée sur un seul point : la fonction d'approximation.
D'autre part, dans le cadre de la thèse de Romain Guider, nous avons étudié une méthode d'analyse statique de programmes à objets qui permet de découvrir automatiquement des propriétés des graphes d'objets. On peut notamment, avec cette technique, prouver qu'à un point de programme donné, deux chemins distincts dans le graphe d'objets ne conduisent jamais à une même cellule de mémoire, ou encore qu'un objet ne sera plus accessible après la prochaine opération du programme. Ces propriétés permettent différentes manipulations de programmes.

Une application possible est la parallélisation, c'est à dire utiliser les propriétés prouvées par l'analyse des programmes pour transformer un programme à objets afin de produire un programme parallèle (exécutant plusieurs tâches simultanément) et distribué (s'exécutant simultanément sur plusieurs machines physiques distinctes) [30].

Une autre application est l'optimisation de programme dans un compilateur. Nous utilisons Java comme langage témoin et avons étudié comment appliquer et intégrer ces techniques

d'analyses statiques dans le compilateur TurboJ fourni par le Silicomp Research Institute (équipe de Vania Joloboff, Grenoble). Ce compilateur permet de compiler un programme destiné à être exécuté sur une machine virtuelle Java de manière à ce qu'il s'exécute plus vite et à ce qu'il consomme un espace mémoire raisonnable, la mémoire étant une ressource précieuse dans un système informatique embarqué. Ces travaux ont été menés dans le cadre d'un externat industriel et permettent d'envisager pour l'évolution de TurboJ des optimisations plus puissantes.

Nos travaux sur les transformations de programmes fonctionnels à l'aide des techniques de grammaires attribuées ont abouti à la publication de trois articles [26, 27, 28] et à la réalisation d'un prototype par Loic Correnson qui devrait soutenir sa thèse en tout début de l'année 2000 (Université de Marne la Vallée). Le résultat important de ces travaux est de montrer que les techniques de transformation de programmes issues des grammaires attribuées donnent de meilleurs résultats par rapport aux techniques classiques de la communauté fonctionnelle. Plus précisément, ce type de transformation, appelée *déforestation*, permet de transformer un programme fonctionnel utilisant des structures intermédiaires en un nouveau programme, de même sémantique, mais où les structures intermédiaires ont été éliminées. Ces travaux ont abouti à la définition d'un nouveau formalisme, appelé *Sémantique Equationnelle*. Les motivations et les objectifs associés à cette nouvelle forme de sémantique sont les suivants :

1. unification des diverses techniques de transformation (déforestation, évaluation partielle) ;
2. correction de la transformation exprimée simplement, indépendamment de toutes sémantiques opérationnelles ;
3. sémantique simple pour les grammaires attribuées dynamiques.

6.3 Spécifications sémantiques et vérifications

Participants : Isabelle Attali, Gilles Barthe, Simao De Sousa, Didier Parigot, Shen-Wei Yu.

Nous avons développé autour de Java Card une méthodologie combinant des systèmes de preuves (avec la modélisation en Coq de la Machine Virtuelle Java Card JCVM et du μ-calcul) et des *model-checkers* afin d'étudier le partage des objets. Le model-checker, implementé en Caml, a la particularité de construire un terme de preuve Coq, cette approche permettant d'assurer une plus grande fiabilité de l'outil de vérification.

Nous avons également initié une réflexion sur les formalismes pour les spécifications sémantiques et suggéré l'utilisation des mécanismes de sous-typage, et notamment le sous-typage par constructeurs [3, 19, 21], afin de simplifier la modélisation des langages de programmation.

Finalement, nous avons poursuivi nos travaux sur l'inférence de types et les transformations de preuves dans les langages sous-jacents des systèmes de preuve [16, 22, 17].

6.4 Outils et méthodologie pour la modélisation et la vérification de politiques de sécurité

Participants : Isabelle Attali, Gilles Barthe, Denis Caromel, Carine Courbis, Simao De Sousa, Alexandre Fau, Refik Molva, Didier Parigot, Yves Roudier, Marjorie Russo, Shen-Wei Yu.

Nous souhaitons étudier d'une part des critères qui garantissent certaines propriétés de sécurité, et d'autre part les outils qui pourraient permettre de formaliser ces propriétés. Le cadre de ces recherches s'applique naturellement à Java et Java Card.

L'ensemble de ces propriétés constitue ce qu'on appelle une « politique de sécurité », qui prend en compte la nature des acteurs agissant sur le système (droits d'un acteur en fonction du moment, de son historique, de l'état du système, etc.). Ces propriétés sont spécifiques (à un acteur) et génériques (par exemple relatives au flux d'information). Une politique de sécurité peut par exemple garantir l'intégrité, la confidentialité des données, ou encore la non-répudiation.

Nous avons étudié le nouveau modèle de sécurité Java et l'avons comparé avec les modèles formels existants. A partir de cette étude, nous avons pu formaliser la sécurité en Java 2 (rapprochement avec le modèle de sécurité par matrice d'accès), ainsi que les domaines de protection. Un calcul de treillis des domaines de protection nous permet de déduire des propriétés de circulation d'information (un logiciel qui calcule et visualise le treillis pour un fichier de securité donné a été réalisé) [31].

Nous étudions également l'utilisation des types pour sécuriser le code mobile. Dans une ligne de travail, nous développons des systèmes de types non-standard tels que les programmes bien typés préservent la confidentialité et/ou l'intégrité des données (propriété de Non-Interférence [20]) ; nous souhaitons à terme enrichir ces systèmes de types pour manipuler des exemples de programmes plus réalistes. Dans une autre ligne de travail, nous étudions l'utilisation des systèmes de preuve pour établir l'adhérence d'un programme donné à une certaine politique de sécurité donnée.

6.5 Etude formelle des modèles à objets distribués

Participants : Isabelle Attali, Denis Caromel, Romain Guider, Didier Parigot, Bernard Serpette, Julien Vayssière.

Dans le cadre de travaux antérieurs, nous avons proposé, étudié et enrichi progressivement un modèle de programmation à objets distribués qui permet, entr'autres, de rendre possible la réutilisation dans le cadre de la programmation parallèle, distribuée, et concurrente. Cela se traduit en particulier par un comportement parallèle qui prolonge aussi souvent que possible celui existant en séquentiel.

On retrouve naturellement ce trait dans les aspects sémantiques formelles. D'une certaine manière, des systèmes distribués obtenus à partir d'un système séquentiel existant conservent en grande partie leur sémantique séquentielle. Dans le cas de systèmes construits directement, leur sémantique reste proche d'une sémantique séquentielle. Nous avons pu d'ores et déjà formaliser ces propriétés dans un certain nombre de travaux [18, 15], soit sur des sous-ensembles des langages considérés, soit sur des études de cas.

6.6 Implémentation des langages à objets

Participants : Denis Caromel, Didier Parigot, David Sagnol, Bernard Serpette, David Teller, Sébastien Villemot.

Nos études sur les langages à objets ont débuté par l'implémentation d'un outil d'expérimentation sur le calcul des objets d'Abadi et Cardelli qui est aux langages à objets ce que le λ-calcul est aux langages fonctionnels. Cet outil [36] a été défini de manière modulaire afin de permettre des extensions ultérieures. Notamment, grâce à la notion de Visiteurs (*Pattern Visitor*), nous avons pu facilement expérimenter [37] les analyses d'évaluation partielle et de non-interférence décrites dans [20].

6.7 Bibliothèques pour la répartition

Participants : Françoise Baude, Alexandre Bergel, Denis Caromel, Florian Doyon, Fabrice Huet, Stéphane Jasinski, Arnaud Poizat, David Sagnol, Julien Vayssière.

Nous développons un modèle de programmation concurrente à objets ainsi que deux implémentations sous la forme de bibliothèques : l'une pour C++, appelée C++// (`http://www.inria.fr/oasis/c++ll`), l'autre appelée ProActive, 100 % Java (`http://www.inria.fr/oasis/proactive`), faisant suite aux travaux autour d'Eiffel// . ProActive et C++// ont fait l'objet de démonstrations au stand INRIA de SuperComputing'99 [38].
Les qualités essentielles du modèle proposé sont :

- de rendre transparent à l'utilisateur le fait que :
  - les objets de son application sont ou non distants,
  - et sont supportés ou non par des threads,

  permettant ainsi la réutilisation de code, par exemple pour un déploiement sur Internet (un de nos objectifs étant de parvenir à une conservation de la sémantique) ;
- de fournir un modèle haut niveau de gestion de la concurrence entre objets actifs.

Le jeu de primitives que fournit le MOP (Méta Object Protocol) est utilisable pour la définition de mécanismes plus élaborés que le « simple » appel distant de méthode. En particulier, nous avons introduit un mécanisme de migration d'objets utilisable par le programmeur grâce à un ensemble minimal de primitives, mais dont la gestion est totalement transparente [33]. Il devient dès lors possible de construire des API d'agents mobiles grâce à la construction d'itinéraires. Une application envisageable est l'administration de systèmes en réseau (sujet de stage de DEA RSD pour 1999-2000).
L'implémentation du MOP est aussi le lieu idéal pour injecter des optimisations d'exécution, comme par exemple le recouvrement du transfert de certains paramètres d'un appel distant de méthode, avec l'exécution de cette méthode distante, que nous avons expérimenté dans le cadre de C++//. La seule implication du programmeur est de déclarer certains des paramètres d'appel de la méthode comme étant des *later*. Les récentes expériences qui utilisent la version de C++// au dessus de Nexus donnent des résultats bien meilleurs qu'en utilisant la version de C++// au dessus de Schooner [23, 13]. La raison est que C++// au dessus de Nexus utilise des processus légers, donc permet de réaliser de manière concurrente l'exécution de la méthode distante et la réception des paramètres retardataires.

Une extension à C++// a été proposée afin d'autoriser le partage d'objets en lecture entre objets actifs [24, 25]. Ces travaux ont donné lieu à une application de calcul matriciel creux. Il s'agissait de montrer que C++// pouvait être aussi performant que MPI tout en simplifiant

la réutilisation du code séquentiel. Ce travail a été effectué en collaboration avec la société Adulis.

A titre d'illustration de ProActive, nous avons développé DIVA (Distributed and Interactive Virtual world in Java), une application répartie collaborative pour un monde virtuel interactif entièrement écrite en Java en utilisant Java3D, RMI, et ProActive [35, 34].

6.8 Sécurisation des applications à objets distribuées

Participants : Françoise Baude, Denis Caromel, Fabrice Huet, Refik Molva, Yves Roudier, Bernard Serpette, Julien Vayssière.

Il s'agit d'intégrer, en suivant les approches appliquée et pragmatique, des caractéristiques de sécurité dans les bibliothèques pour la répartition, et par là-même, aux applications développées en partant par exemple d'une application collaborative existante. Notamment, les communications sur Internet circulant en clair par défaut, les applications peuvent requérir que les informations échangées (que ce soient des données ou du code) soient protégées.

Ces recherches, en phase de démarrage, seront menées en collaboration avec R. Molva (Eurecom) et G. Castagna (ENS) dans le cadre du projet financé par le CNRS (voir 8.2.2).

7 Contrats industriels (nationaux, européens et internationaux)

7.1 Environnements de développement et vérification pour Java et Java Card

Participants : Isabelle Attali, Denis Caromel, Carine Courbis, Henrik Nilsson, Didier Parigot, Marjorie Russo, Shen-Wei Yu.

Il s'agit d'un contrat de recherche avec Bull/CP8 (1998 - 2001) d'un montant de 1 362 KF (pour une description des activités scientifiques, voir 6.1).

7.2 SmartTools : outils génériques pour la construction d'environnements de développement

Participants : Isabelle Attali, Denis Caromel, Franck Chalaux, Pascal Degenne, Alexandre Fau, Stéphane Jasinski, Francis Montagnac, Didier Parigot.

Action Dyade SmartTools démarrée en Juin 1999 sur le thème « outils génériques pour la construction d'environnements de développement » (voir 5.1), avec la participation de Microsoft (pour un montant de 440 KF).

Le support des différents partenaires de l'action SmartTools nous permettent de nous appuyer sur une équipe composée de 5 chercheurs ou enseignants-chercheurs et quatre ingénieurs (dont un mis à disposition par Bull pour la durée de l'action).

Deux réunions pleinières (25 participants) ont été organisées à l'INRIA Sophia Antipolis : journée de lancement le 2 Juin et journée de travail après 6 mois le 10 Décembre.

7.3 Analyse de formes pour l'optimisation de la compilation Java

Participants : Isabelle Attali, Denis Caromel, Romain Guider.

Externat industriel de Romain Guider à Open Group (Silicomp Research Institute), à Grenoble de Février à Juin 1999 (voir 6.2). Montant 55 KF, supporté également par l'Anvar.

7.4 Formavie - Modélisation Formelle et Certification Sécuritaire pour Machine Virtuelle Embarquée

Participants : Isabelle Attali, Gilles Barthe, Simao De Sousa, Didier Parigot, Shen-Wei Yu.

Projet OPPIDUM, démarré le 1er juin 1999 d'un montant de 1 360 KF.
Les partenaires industriels sont Bull CP8 et Schlumberger SC & T.
Le travail effectué par l'équipe dans le cadre de ce projet est développé dans 6.3 et 6.4.

8 Actions régionales, nationales et internationales

8.1 Actions régionales

8.1.1 Programmation répartie et collaborative pour Internet

Participants : Denis Caromel, Julien Vayssière.

Cette action, financée par la Région PACA, en partenariat avec la société Questel (fournisseur de recherche d'antériorité pour le dépôt de brevets), permet de supporter la thèse de doctorat de Julien Vayssière.

8.2 Actions nationales

8.2.1 Action de Recherche Coopérative Java Card

Participants : Isabelle Attali, Gilles Barthe, Denis Caromel, Carine Courbis, Alexandre Fau, Romain Guider, Bernard Serpette, Shen-Wei Yu.

L'action de recherche coopérative Java Card : sémantique, optimisations et sécurité (voir `http://www.irisa.fr/lande/jensen/javacard.html`) a démarré en Janvier 1998 pour deux ans.

L'action est coordonnée par le projet LANDE, et a pour partenaires le projet CRISTAL et deux partenaires extérieurs : le CERT, laboratoire de l'ONERA et l'action VIP de Dyade. Deux groupes de travail réunissant l'ensemble des partenaires ont eu lieu en Mars et en Juillet.

L'action Java Card vise à coordonner les travaux effectués à l'INRIA autour des aspects formels de Java. L'action se focalisera sur la sémantique et la vérification des programmes Java Card, la version de Java dédiée aux cartes à puces.

La participation du projet OASIS repose sur la description formelle des applications Java Card. Cette année, deux réunions pléinières ont eu lieu : à Saint Malo les 8 et 9 Juillet et à l'ENS Ulm le 8 Décembre. Les activités de recherche sont décrites en 6.1 et 6.3.

8.2.2 Programmation répartie collaborative et sécurisée pour Internet

Participants : Isabelle Attali, Françoise Baude, Denis Caromel, Fabrice Huet, Refik Molva, Yves Roudier, Julien Vayssière.

Financement CNRS (montant 250 KF pour deux ans) qui démarre en décembre 1999, en partenariat avec R. Molva (Eurecom) et G. Castagna (ENS-Ulm). Ce projet de recherche vise à développer une méthodologie et les outils associés afin de faciliter le développement d'applications réparties, collaboratives sur Internet. Ces applications présentent par ailleurs des exigences de sécurité à la fois fortes et aussi variées que sont la confidentialité ou la détection d'intrusion.

8.2.3 Spécification, Vérification, Sémantique

Participants : Isabelle Attali, Françoise Baude, Denis Caromel, Carine Courbis, Marjorie Russo.

Participation au groupe de travail « Spécification, Vérification, Sémantique » du pôle Objets du GDR ALP : Algorithmique, Langages et Programmation. Responsable Giuseppe Castagna, ENS-Ulm.

8.3 Actions internationales

8.3.1 Spécification formelle et transformation de programmes parallèles

Participants : Isabelle Attali, Denis Caromel, Romain Guider.

Contrat NSF - INRIA - CNRS (1997-2000, montant 165 KF) sur la spécification formelle et la transformation de programmes parallèles concernant J.-L. Gaudiot (USC), en partenariat avec Andrew Wendelborn, University of Adelaide, Australie.

8.3.2 Programmation à objets

Participants : Isabelle Attali, Denis Caromel, Marjorie Russo.

Collaboration INRIA-Brésil (1995-1999, montant 60 KF) avec Fabio Da Silva, Paolo Borba (Recife) et Sidi Ould Ehmety (São Luis) sur la programmation à objets.

8.3.3 Concurrence et applications

Participants : Denis Caromel, Julien Vayssière.

Denis Caromel est co-responsable du groupe de travail « Concurrence et applications » du Java Grande Forum (voir `http://www.javagrande.org`).

8.4 Visites, Participations à des conférences, et invitations de chercheurs

- Visites :
  - Julien Vayssière a visité l'équipe de Andrew Wendelborn à l'Université d'Adelaide de Avril à Octobre.
  - Isabelle Attali et Denis Caromel ont visité USC et ISI dans le cadre du contrat NSF - INRIA - CNRS du 13 Juillet au 6 Août 1999.
  - Visite de Gilles Barthe à SICS (Stockholm), 7 et 8 Juin.
- Participations conférences écoles :
  - Denis Caromel a participé à la conférence ERC à (San Francisco, 23-26 février).
  - Carine Courbis a participé à l'Ecole des Jeunes Chercheurs, (Lille, 20-30 mars).
  - Isabelle Attali et Denis Caromel ont participé à IPPS/SPDP 99, ainsi qu'aux workshops "Java for Parallel and Distributed Computing" et "Formal Methods for Parallel Programming, Theory and Applications" (Puerto Rico, Avril).
  - Arnaud Poizat, David Sagnol, Stéphane Jasinski ont participé à la conférence JES à (Paris, 27-29 avril).
  - Shen-Wei Yu a soutenu son PhD à Durham le 5 Mai.
  - Denis Caromel a été conférencier invité à la "European Science Foundation Conference: Problem-Solving Environments: Infrastructure and Prototypes" San Feliu de Guixols (Espagne, 12-17 Juin).
  - Denis Caromel a participé à la conférence ECOOP 99 (Lisbonne, 14-18 juin).
  - Julien Vayssière a participé aux conférences Java'One et Java Grande Forum (San Francisco, 15-18 juin).
  - David Sagnol a participé à HPCN'99 (Amsterdam) et aux journées RenPar11 (Irisa) en juin.
  - Henrik Nilsson a participé au 17th Modelica design meeting (Aveiro, Portugal, 6-8 mai), et au 19th Modelica design meeting, à Dresde, (Allemagne, 18-20 novembre).
  - Isabelle Attali, Denis Caromel, Carine Courbis, Romain Guider et Alexandre Fau ont participé à la réunion Java Card de St Malo (9-10 Juillet).
  - Isabelle Attali et Stéphane Jasinski ont participé au Workshop Dyade « cartes à puces et sécurité », (Rocquencourt, 3 Septembre).
  - Henrik Nilsson a participé à TYPES SUMMER SCHOOL' 99,(Giens, du 30 août au 10 septembre).
  - Isabelle Attali a participé au User Meeting Group TLA+ dans le cadre de FM99 (Toulouse le 20 Sept).
  - Carine Courbis, Bernard Serpette, Shen-Wei Yu ont participé à l'Université d'été Bull sur le GSM (Sophia Antipolis, 13-14 Septembre).
  - Henrik Nilsson a participé à PLI COLLOQUIUM (Paris du 27 septembre au 1er octobre), ainsi qu'à ICFP'99 (pour présenter son article [29] et PPDP'99.
  - Présentation Isabelle Attali de SmartTools à Microsoft Research (Cambridge, 28 oct).

  - Denis Caromel, Fabrice Huet et David Sagnol ont participé à SC'99 Portland (13-19 nov)(stand INRIA [38]) ; Denis Caromel a participé au Panel "The Role of Java in High Performance Computing".
  - Carine Courbis a participé à CARTES 99, (Paris, 17-18 novembre).
  - Denis Caromel, Fabrice Huet et Julien Vayssière ont participé aux journées PRO (Lille, 25-26 Novembre).
  - Gilles Barthe a participé aux conférences ETAPS (Amsterdam), Workshop Types (Lökeberg), FLOPS (Japon), en Novembre.
  - Gilles Barthe, Carine Courbis, Bernard Serpette, Shen-Wei Yu ont participé à la reunion Java Card de l'ENS Ulm (8 Décembre).
  - Denis Caromel a participé à la Conférence ISCOPE'99, (San Francisco, Décembre).
- Visiteurs étrangers :
  - Ewen Denney (post-doc Lande, IRISA, ARC JavaCard) 26-28 Mai.
  - Simao De Sousa et José Bacelar (doctorants Univ. Minho, Portugal) en Juillet.
  - John Hatcliff et Matthew Dwyer (Kansas State University) 16-17 septembre.
  - Maria Joao Frade (doctorante Universidade do Minho, Portugal) 11-24 Septembre.
  - Jean-Christophe Pazzaglia (CRS4, Sardaigne) le 19 octobre.
  - Giuseppe Castagna (ENS Ulm) 16-17 novembre.

9 Diffusion de résultats

9.1 Animation de la Communauté scientifique

- Isabelle Attali est membre du comité de direction du GDR ALP : Algorithmique, Langages et Programmation.
- Isabelle Attali et Denis Caromel ont participé au comité d'édition du numéro spécial de la revue « L'Objet » (éditions Hermès) sur le thème "Formal Methods For Object Systems".
- Françoise Baude participe au comité d'édition de la revue Calculateurs Parallèles, éditions Hermès (depuis Mars 96).
- Françoise Baude a participé au comité de programme de RenPar'11 (8 au 11 juin 1999) à Rennes.
- Denis Caromel participe au comité d'édition de la revue L'OBJET, éditions Hermès (depuis Janvier 97).
- Denis Caromel co-organise avec Jean-Marc Geib (LIFL) et Patrick Sallé (IRIT) l'action Inter-GDR PRO : Parallélisme, Répartition et Objets (GDR ALP : Algorithmique, Langages et Programmation et GDR ARP : Architecture, Réseaux et systèmes, Parallélisme)
  Organisation des journées PRO : Recherches en cours et Applications Lille, 25-26 Novembre 1999.
  Voir `http://www.inria.fr/oasis/PRO-Lille99`.

- Denis Caromel a participé aux comités de programme de :
  - LMO'99, Villefranche sur Mer, 27-29 Janvier 1999.
  - International Conference and Exhibition on High-Performance Computing and Networking (HPCN Europe'99 (Amsterdam) ; actes Springer Verlag, Lectures Notes in Computer Science (LNCS).
- Denis Caromel a participé aux comités d'organisation de :
  - Workshop "Java for Parallel and Distributed Computing", Co-chair with S. Chaumette, G. Fox, April 1999, actes Springer, Lectures Notes in Computer Science (LNCS) [11], within IPPS/SPDP 99.
  - ISCOPE'99, Décembre 1999, San Francisco, actes Springer, Lectures Notes in Computer Science (LNCS).
- Didier Parigot a organisé et Isabelle Attali a participé au Comité de Programme du workshop WAGA'99 [12] (2nd International Workshop on Attribute Grammars and their Applications), voir `http://www.inria.fr/oscar/waga99.html`.
- Plusieurs membres du projet ont participé à :
  - des jurys de thèse de doctorat en tant que :
    - directeur : Christophe Roudet (UNSA), Nathalie Furmento (UNSA, projet SLOOP)
    - rapporteur : Ralf Lämmel (Rostock), Jean-Yves Vion-Dury (Grenoble), Abdelkrim Nimour (ENST Paris), Pascale Launay (IRISA), Houari TINE (INSA Lyon)
    - président : Silvano Dal-Zilio (UNSA)
  - des jurys d'habilitation à diriger des recherches en tant que rapporteur : Jean-Louis Sourrouille (INSA LYON)

  Plusieurs membres du projet ont évalué des articles soumis aux :
  - conférences suivantes :
    WAGA'99, TYPES'98, ECOOP'99, HPCN'99, IPPS/SPDP'99, ICFP'99, ISCOPE'99, LICS'99, RTA'99, SCCC'99, ESOP'2000
  - et revues suivantes : TSI, TAPOS, Concurrency Practice and Experience, Software Practice and Experience, The Computer Journal, TCS, Mathematical Structures in Computer Science, Journal of Functional Programming, Journal of Logic and Computation.

9.2 Enseignement

- Isabelle Attali est responsable du module de tronc commun « Sémantique et Sécurité » du DEA d'Informatique de l'UNSA, module dans lequel intervient également Gilles Barthe (module de 36h).
- Isabelle Attali est membre du conseil scientifique de ce DEA.
- Isabelle Attali a été responsable de la Formation pour l'INRIA Sophia Antipolis jusqu'en Septembre 1999.

- Isabelle Attali est responsable scientifique depuis 1995 de l'Ecole Jeunes Chercheurs en Programmation, qui réunit environ 40 doctorants pour 15 jours de cours chaque année. En 1999, l'Ecole a eu lieu à l'ENIC à Lille (voir `http://www-sop.inria.fr/EJC99`).
- Isabelle Attali est responsable depuis Novembre 99 de la commission Formation Emploi de l'association Télécom-Valley.
- Françoise Baude
  - est responsable de l'enseignement de licence d'informatique de l'UNSA « Concepts des systèmes d'exploitation et gestion de la concurrence », ainsi que de celui « Utilisation avancée de systèmes d'exploitation » ;
  - participe à l'enseignement de Java en Licence MASS ;
  - participe au module de tronc commun du DEA RSD « Algorithmique parallèle et distribuée » ainsi qu'à celui « Algorithmique parallèle et distribuée - outils de programmation » de la filière SAR de l'ESSI 3ème année.
- Denis Caromel est responsable des modules de Maitrise « C++ » et « Programmation distribuée et Administration Système ».
- Denis Caromel est responsable de la filière « Systèmes Distribués » du DEA RSD (Réseaux et Systèmes Distribués) de l'UNSA (en collaboration avec CMA, CNET, Eurécom, INRIA Sophia Antipolis), depuis Septembre 1995.
- Denis Caromel est responsable depuis 1995 du Module « Langages de Programmation Concurrente, Parallèle, Distribuée » commun aux DEAs Informatique et RSD de l'UNSA.
- Denis Caromel est coordinateur au Département Informatique du DESS Télécommunications, Université de Nice - Sophia Antipolis.
- Didier Parigot est chef de travaux pratiques à l'Ecole Polytechnique.
- Gilles Barthe a été responsable du module LOI « Introduction à AWK et C » au 2e semestre pour le DEUG MASS 2e année à l'UNSA.

10 Bibliographie

Ouvrages et articles de référence de l'équipe

[1] I. Attali, D. Caromel, S. O. Ehmety, S. Lippi, « Semantic-based visualization for parallel object-oriented programming », *in : Proceedings of OOPSLA'96, ACM Sigplan Notices, 31, 10*, ACM Press, San Jose, CA, octobre 1996.

[2] I. Attali, D. Caromel, S. O. Ehmety, « A Natural Semantics for Eiffel Dynamic Binding », *ACM Transactions on Programming Languages and Systems (TOPLAS) 18*, 5, novembre 1996.

[3] G. Barthe, « Order-sorted inductive types », *Information and Computation 149*, 1, février 1999, p. 42–76.

[4] D. Caromel, F. Belloncle, Y. Roudier, « The C++// System », *in : Parallel Programming Using C++*, MIT Press, 1996, ISBN 0-262-73118-5.

[5] D. Caromel, W. Klauser, J. Vayssiere, « Towards Seamless Computing and Metacomputing in Java », *Concurrency Practice and Experience 10*, 11–13, novembre 1998, p. 1043–1061.

[6] D. Caromel, « Towards a Method of Object-Oriented Concurrent Programming », *Communications of the ACM 36*, 9, septembre 1993, p. 90–102.

[7] L. Correnson, E. Duris, D. Parigot, G. Roussel, « Generic Programming by Program Composition (position paper) », *in : Workshop on Generic Programming*, Marstrand, Sweden, juin 1998. conjunction with MPC'98, `ftp://ftp-sop.inria.fr/oasis/Didier.Parigot/publications/Correnson98a.ps.gz`.

[8] N. Furmento, F. Baude, « Schooner: An Object-Oriented Run-time Support for Distributed Applications », *in : In K. Yetongnon and S. Hariri, editors , Proceedings of Parallel and Distributed Computing Systems (PDCS'96), Dijon, 1*, International Society for Computers and their Applications (ISCA), p. 31–36, septembre 1996.

[9] D. Parigot, G. Roussel, M. Jourdan, E. Duris, « Dynamic Attribute Grammars », *in : Int. Symp. on Progr. Languages, Implementations, Logics and Programs (PLILP'96)*, H. Kuchen, S. D. Swierstra (éditeurs), *Lecture Notes in Computer Science, 1140*, Springer-Verlag, p. 122–136, Aachen, septembre 1996, `ftp://ftp-sop.inria.fr/oasis/Didier.Parigot/publications/plilp96.ps.gz`.

[10] B. Serpette, « Approximations d'évaluateurs fonctionnels », *in : Proceedings of WSA (Workshop on Static Analysis)*, Bigre, p. 79–90, 1992.

Livres et monographies

[11] D. Caromel, S. Chaumette, G. Fox (éditeurs), *Java for Parallel and Distributed Computing*, IPPS/SPDP'99, Puerto Rico, Springer Verlag No 1586, avril 1999. ISBN 3-540-65831-9.

[12] D. Parigot, M. Mernik (éditeurs), *Second Workshop on Attribute Grammars and their Applications WAGA'99*, ETAPS'99, Amsterdam, The Netherlands, INRIA Rocquencourt, mars 1999. Satellite event of ETAPS'99, `http://www-rocq.inria.fr/oscar/www/fnc2/WAGA99/accept.html`.

Thèses et habilitations à diriger des recherches

[13] N. Furmento, Schooner *: Une encapsulation orientée objet de supports d'exécution pour applications réparties*, thèse de doctorat, Université de Nice-Sophia Antipolis, mai 1999.

[14] C. Roudet, *Conception et implantation d'un langage pour les transformations de programmes*, thèse de doctorat, Université de Nice - Sophia Antipolis, mai 1999.

Articles et chapitres de livre

[15] I. Attali, D. Caromel, S. O. Ehmety, « Formal Properties of the Eiffel// Model », *in : Parallel and Distributed Objects*, Hermes Science Publications, 1999.

[16] G. Barthe, « Theoretical pearl: type-checking injective pure type systems », *Journal of Functional Programming*, à paraître.

[17] G. Barthe, J. Hatcliff, M. Sørensen, « CPS-translations and applications: the cube and beyond », *Higher-Order and Symbolic Computation 12*, septembre 1999, p. 125–170.

Communications à des congrès, colloques, etc.

[18] I. Attali, D. Caromel, S. Lippi, « From a Specification to an Equivalence Proof in Object-Oriented Parallelism », *in : FMPPTA'99: Modeling and Proving (Fourth Workshop in Formal Methods for Parallel Programming, Theory and Applications), 1586*, Springer, LNCS, 1999.

[19] G. Barthe, M. Frade, « Constructor subtyping », *in : Proceedings of ESOP'99*, D. Swiestra (éditeur), *Lecture Notes in Computer Science, 1576*, Springer-Verlag, p. 109–127, 1999.

[20] G. Barthe, B. Serpette, « Partial evaluation and non-interference for object calculi », *in : Proceedings of FLOPS'99*, A. Middeldorp, T. Sato (éditeurs), *Lecture Notes in Computer Science, 1722*, Springer-Verlag, p. 53–67, 1999.

[21] G. Barthe, F. van Raamsdonk, « Constructor Subtyping in the Calculus of Inductive Constructions », *in : Proceedings of FOSSACS'00*, J. Tiuryn (éditeur), *Lecture Notes in Computer Science*, Springer-Verlag, 2000.

[22] G. Barthe, « Expanding the cube », *in : Proceedings of FOSSACS'99*, W. Thomas (éditeur), *Lecture Notes in Computer Science, 1578*, Springer-Verlag, p. 90–103, 1999.

[23] F. Baude, D. Caromel, N. Furmento, D. Sagnol, « Overlapping Communication with Computation in Distributed Object Systems », *in : Proceedings of the 7th International Conference - High Performance Computing Networking'99 (HPCN Europe 1999)*, P. Sloot, M. Bubak, A. Hoekstra, B. Hertzberger (éditeurs), *Lecture Notes in Computer Science, 1593*, p. 744–753, Amsterdam, The Netherlands, avril 1999.

[24] D. Caromel, E. Noulard, D. Sagnol, « Partage en lecture pour la programmation à objets parallèle et distribuée », *in : Proceedings of 'Onzièmes Rencontres Francophones du Parallélisme' (RenPar'11)*, p. 19–24, Rennes, juin 1999.

[25] D. Caromel, E. Noulard, D. Sagnol, « SharedOnRead Optimization in Parallel Object-Oriented Programming », *in : Computing in Object-Oriented Parallel Environments, Proceedings of ISCOPE'99*, *Lecture Notes in Computer Science*, San Francisco, décembre 1999.

[26] L. Correnson, E. Duris, D. Parigot, G. Roussel, « Declarative program transformation: a deforestation case-study », *in : Principles and Practice of Declarative Programming PPDP'99*, G. Nadathur (éditeur), *Lecture Notes in Computer Science, 1702*, p. 353–369, Paris, France, octobre 1999, `ftp://ftp-sop.inria.fr/oasis/Didier.Parigot/publications/Correnson99b.ps.gz`.

[27] L. Correnson, E. Duris, D. Parigot, G. Roussel, « Equational Semantics », *in : Symposium Static Analysis SAS'99*, A. Cortesi, G. Filé (éditeurs), *Lecture Notes in Computer Science, 1694*, p. 264–283, Venise, Italie, Sept 1999, `ftp://ftp-sop.inria.fr/oasis/Didier.Parigot/publications/Correnson99a.ps.gz`.

[28] L. Correnson, « Equational Semantics », *in : Second Workshop on Attribute Grammars and their Applications, WAGA'99*, D. Parigot, M. Mernik (éditeurs), INRIA rocquencourt, p. 205–222, Amsterdam, The Netherlands, mars 1999, `http://www-rocq.inria.fr/oscar/www/fnc2/WAGA99/accept.html`.

[29] H. Nilsson, « "Tracing Piece by Piece: Affordable Debugging for Lazy Functional Languages" », *in : Proceedings of the 1999 ACM SIGPLAN international conference on Functional programming*, ACM Press, p. 36–47, Paris, France, septembre 1999.

Rapports de recherche et publications internes

[30] I. Attali, D. Caromel, R. Guider, « Static Analysis of Java for distributed and parallel programming », *rapport de recherche n° 3634*, INRIA, mars 1999, `ftp://ftp-sop.inria.fr:/pub/rapports/RR-3634.ps`.

[31] A. Fau., « Politiques de sécurité pour Java », *rapport de recherche*, DEA Informatique UNSA, juillet 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/AlexandreFauStageDEA.ps`.

[32] L. Henrio, « Tests Interactifs d'Applications Java Card », *rapport de recherche*, Stage d'Option, Ecole Polytechnique, juillet 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/LudovicHenrioStageX0699.pdf`.

[33] F. Huet, « Multicast et migration », *rapport de recherche*, Stage DEA RSD UNSA, juillet 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/FabriceHuetStageDEA0699.pdf`.

[34] S. JASINSKI, « Développement, modifications d'un logiciel collaboratif utilisant les technologies JAVA 3D et la bibliothèque ProActive PDC développée pour l'INRIA », *rapport de recherche*, ESSI - UNSA, septembre 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/JasinskiStageESSI.ps`.

[35] A. POIZAT, « Développement, modifications d'un logiciel collaboratif utilisant les technologies JAVA 3D et la bibliothèque ProActive PDC développée pour l'INRIA », *rapport de recherche*, ESSI - UNSA, septembre 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/PoizatStageDEA.ps`.

[36] D. TELLER, « Evaluation et typage du calcul des objets », *rapport de recherche*, Magister - ENS Lyon, juillet 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/TellerStageENS.ps`.

[37] S. VILLEMOT, « Applets d'inférence de types pour les calculs d'objets », *rapport de recherche*, Maîtrise- ENS Paris, août 1999, `ftp://ftp-sop.inria.fr/oasis/publications/1999/VillemotStageENS.ps`.

Divers

[38] CAPS, OASIS, PARIS, « INRIA at SC'99 (cdrom) », novembre 1999.