Projet CRISTAL

Programmation typée, modularité et compilation

Localisation : *Rocquencourt*

Mots-clés : compilation, compilation séparée, environnement de programmation, génération de code, lambda-calcul, programmation fonctionnelle, typage.

1 Composition de l'équipe

Responsable scientifique

Michel Mauny, Directeur de Recherche Inria

Responsable permanent

Pierre Weis, Directeur de Recherche Inria

Secrétaire (Commun avec COQ et PARA)

Sylvie Loubressac, TR Inria

Conseillers scientifiques

Guy Cousineau, Professeur, Univ. Paris 7
Christian Queinnec, Professeur, Univ. Paris 6

Personnel Inria

Xavier Leroy, Chargé de Recherche
Daniel de Rauglaudre, Ingénieur de Recherche
Didier Rémy, Chargé de Recherche
François Rouaix, Chargé de Recherche

Ingénieurs experts

Jean-Marie Geffroy, jusqu'au 31 mai 1996
Valérie Ménissier-Morain, depuis le 1er octobre 1996

Chercheurs doctorants

Jun Furuse, boursier du gouvernement Japonais, depuis octobre 1996
Robert Harley, allocataire MENESR, depuis octobre 1996
François Pottier, allocataire normalien (ENS-Paris)
Christian Rinderknecht, allocataire MENESR (Univ. Paris 6)
Émilie Sayag, allocataire MENESR (Univ. Paris 7)
Jérôme Vouillon, allocataire normalien (ENS-Paris)

Stagiaires

François Pessaux, DEA, Univ. Paris 6
Pascal Cuoq, Magistère, ENS-Lyon

2 Présentation du projet

Le projet Cristal s'intéresse à différents formalismes de typage statique des langages de programmation et à leur utilisation dans la conception et la mise en œuvre d'outils de programmation typée robustes et efficaces.

Le typage statique accroît la sécurité de programmation, la rapidité de développement d'applications et facilite leur maintenance. Par ailleurs, les systèmes de types sont aussi parmi les formalismes de base de recherches de preuves de programmes où les types sont vus comme des spécifications. Il s'agit là d'autant d'arguments montrant la nécessité d'environnement de programmation typée alliant fiabilité, sécurité et efficacité.

Le typage statique des langages de programmation impose aux programmes de satisfaire un certain nombre de propriétés. L'objectif de nos travaux est de tirer parti de ce fait dans la recherche d'une compilation efficace, dans la conception d'extensions de ces langages et dans la conception d'environnements de programmation typée. Nous nous intéressons aussi aux liens et aux interactions entre systèmes de preuves de programmes et environnements de programmation. Nos travaux se situent donc au carrefour de la théorie des types, de la conception et la mise en œuvre de langages de programmation et de la programmation proprement dite.

Nos implantations de Caml, un langage fonctionnel typé et puissant de la famille ML, développées au sein de l'équipe, sont utilisées à la fois comme contexte d'étude, comme outils d'implantation et comme banc d'essai des outils que nous développons.

Les directions de recherche suivies durant l'année 1996 ont été la poursuite de l'effort autour du langage Caml, l'exploitation des bonnes propriétés de ce langage (et plus généralement des langages statiquement typés) dans le domaine du Web, et notamment du code mobile, ainsi qu'un certain nombre de travaux plus en amont concernant notamment les systèmes de types.

Cette année 1996 a été marquée par les faits suivants :

- La mise en distribution d'Objective Caml, la première extension orientée-objets de ML supportant l'inférence de types. Cette implantation concrétise les travaux de recherche de Didier Rémy sur le typage des objets, ainsi que ceux de Xavier Leroy sur la compilation et les systèmes de modules.
- La réalisation et la mise en distribution par François Rouaix, en collaboration avec Bernard Lang du projet ATOLL du *proxy* individuel V6 qui permet à l'utilisateur d'installer des filtres entre son navigateur et les serveurs Web. V6 est programmé en Objective Caml.

3 Actions de recherche

3.1 Implémentation de Caml

Une nouvelle implémentation de Caml a vu le jour en 1996 : il s'agit d'Objective Caml, doté notamment d'objets et de classes admettant la synthèse de types. Cette implémentation a été portée vers Microsoft Windows, réutilisant le portage de Caml-Light réalisé par Jean-Marie Geffroy, permettant à un grand nombre d'utilisateurs de bénéficier du langage ainsi que des applications développées en Objective Caml. Nous avons par ailleurs effectué des travaux d'intégration d'outils syntaxiques (extension ou spécialisation du langage) apparaissant comme une contribution indépendante du langage Objective Caml.

3.1.1 Objective Caml

Participants : Xavier Leroy, Jérôme Vouillon

Objective Caml (précédemment connu sous le nom de Caml Special Light) est une réimplémentation quasi-complète de Caml Light, entreprise par Xavier Leroy en 1995, auquel s'est joint Jérôme Vouillon en 1996 pour l'implémentation des objets et des classes. Les principales innovations d'Objective Caml sont :

- un système complet d'objets et de classes, mettant Caml au niveau des meilleurs langages orientés-objets existants ;
- un calcul de modules puissant, mais néanmoins compatible avec la compilation séparée ;
- un compilateur produisant du code assembleur de hautes performances pour la plupart des processeurs du marché (Pentium, Alpha, PowerPC, Sparc, Mips, HPPA).

Le développement majeur de cette année, décrit plus en détails dans le reste de ce rapport, a été l'intégration par Jérôme Vouillon du calcul d'objets et de classes conçu par Didier Rémy et lui-même. En parallèle, Xavier Leroy a poursuivi le développement du système de modules, assuré la maintenance de l'ensemble du système, et effectué de nombreux portages du compilateur. C'est au total 11 *releases* du système que nous avons réalisées entre septembre 1995 et novembre 1996.

Objective Caml a reçu un accueil très favorable. C'est la première extension orientée-objet du langage ML, ou, de manière équivalente, le premier langage orienté-objet à bénéficier de l'inférence de types à la ML, à être rendu publique. La présence de modules puissants et d'un compilateur très efficace a aussi beaucoup contribué à effacer l'aspect "petit langage uniquement pour l'enseignement" parfois attaché à Caml Light, notre précédente implémentation de Caml.

Objective Caml a été utilisé dans plusieurs projets importants, aussi bien à l'intérieur de l'INRIA (le prouveur Coq, le navigateur MMM, le cache Web intelligent V6) qu'à l'extérieur (*cf.* section 5.2.2). Nos contacts étroits avec ces utilisateurs importants ont beaucoup contribué à la qualité générale de l'implémentation Objective Caml et au développement de bibliothèques répondant à leurs besoins, en particulier dans le domaine du parallélisme (*threads*).

3.1.2 Objective Caml sous Microsoft Windows

Participants : Xavier Leroy, François Rouaix, Pascal Cuoq

Objective Caml a été initialement développé sous Unix. Un premier portage pour Windows NT et Windows 95 a été réalisé par l'équipe Graphics de Microsoft Research, pour leurs besoins internes, puis complétée et intégrée dans la distribution principale par Xavier Leroy.

Dans le cadre de son stage de première année de magistère, Pascal Cuoq, élève de l'ENS Lyon, a adapté à Windows NT et 95 deux bibliothèques importantes d'Objective Caml : l'interface avec le système d'exploitation et la bibliothèque de parallélisme. Cette dernière a nécessité une réimplémentation complète, utilisant les *threads* fournis par Windows au lieu de l'approche à base d'entrelacement au niveau de la machine abstraite Caml utilisée dans la version Unix. En collaboration avec François Rouaix, Pascal Cuoq a ensuite mis son travail en application en faisant tourner le *proxy* Web V6 sous Windows.

3.1.3 Outils syntaxiques

Participants : Michel Mauny, Daniel de Rauglaudre

Les années précédentes, Michel Mauny et Daniel de Rauglaudre ont mené une étude sur l'intégration en ML d'outils de manipulation de syntaxes concrètes (analyse syntaxique fonctionnelle, quotations,

grammaires, etc.). Ces travaux se sont concrétisés en 1996 par la distribution du langage Chamau [67], un prototype offrant un mécanisme de quotations ainsi que des grammaires extensibles.

Dans le but d'intégrer au mieux ces mécanismes aux évolutions récentes de Caml, sans pour autant modifier profondément la définition du langage, Daniel de Rauglaudre a développé un préprocesseur d'Objective Caml nommé Camlp4 (pour *preprocessor-pretty-printer*) intégrant ces fonctionnalités. Cette intégration a été effectuée grâce à la possibilité offerte par le compilateur Objective Caml de recevoir en entrée des arbres de syntaxe abstraite, et donc de court-circuiter l'analyse syntaxique des programmes. Camlp4 peut donc se substituer à l'analyseur syntaxique d'Objective Caml, permettant ainsi d'étendre le langage ou bien de le restreindre, et de le spécialiser à telle ou telle application.

La réalisation de Camlp4 a été terminée durant l'année 1996, et une documentation est en cours de rédaction.

3.2 Outils pour le Web

Le langage Caml est bien adapté à la programmation d'outils liés au Web, ainsi que comme langage de programmation d'*applets*. En effet, le système de modules et la sécurité de programmation offerte par le typage statique autorisent le développement d'applications de taille conséquente. Le typage statique ainsi que des techniques de cryptographie garantissent l'innocuité d'*applets* programmées en Caml.

L'action du projet Cristal dans ce domaine s'est vue renforcée cette année par l'arrivée de Valérie Ménissier-Morain, ingénieur-expert financée par le GIE Dyade.

3.2.1 MMM

Participant : François Rouaix

François Rouaix a poursuivi ses travaux sur l'usage de Caml pour le Web : le navigateur MMM, commencé l'année dernière, a été porté vers Objective Caml. Ce navigateur supporte des *applets* sûres en Caml, grâce aux propriétés de typage du langage Caml et à l'utilisation d'authentification cryptographique. Le système d'applets Caml a également été implanté pour le navigateur Web de Netscape sous forme de *plug-in*.

3.2.2 V6

Participant : François Rouaix

En collaboration avec Bernard Lang (projet Atoll), François Rouaix a développé V6, un relai (*proxy*) individuel programmable pour les applications Web. Grâce à V6, l'usager peut installer des filtres entre son navigateur et les serveurs ; ces filtres peuvent travailler aussi bien sur les requêtes émises par le navigateur que sur les documents retournés par les serveurs. V6 fournit ainsi un cadre uniforme pour expérimenter des outils de navigation « intelligente » et autres accessoires facilitant la pratique du Web, et ceci indépendamment du navigateur utilisé.

3.2.3 Les protocoles et la sécurité sur le Web

Participante : Valérie Ménissier-Morain

Depuis le 1er octobre 1996, Valérie Ménissier-Morain occupe un poste d'ingénieur-expert au projet Cristal, financé par le GIE Dyade. Elle travaille sur la certification des protocoles cryptographiques, qui permettent de garantir l'intégrité et la confidentialité de certaines données transmises sur le réseau Internet. Elle étudie également les modèles de sécurité pour le code mobile (*applets* Java ou Caml).

3.2.4 Verified Internet Protocols

Participants : Xavier Leroy, François Rouaix

Nous avons commencé une réflexion commune avec le projet Coq au sujet de la définition d'une couche basse de la représentation de programmes Caml, apte à servir d'interface avec le logiciel Coq. L'un des objectifs de cette étude est, à terme, de pouvoir certifier des morceaux de code Caml transitant sur le réseau sous forme d'*applets*. C'est aussi dans ce cadre que se situe le travail de François Rouaix sur les *applets* en Caml.

3.3 Modularité

La modularité d'un langage de programmation et l'expressivité de ce système de modules sont des facteurs importants dans la réalisation d'applications de taille conséquente, et dans la « réutilisabilité » du code.

3.3.1 Systèmes de modules

Participant : Xavier Leroy

Le travail de Xavier Leroy sur la formalisation de systèmes de modules dans le cadre des formalismes classiques de typage a débouché sur la conception et l'implémentation du système de modules du langage Objective Caml. Ce travail dote donc Caml d'un système de modules expressif (il atteint la puissance de celui de Standard ML, la référence en la matière), mais qui se prête cependant à la compilation séparée dans le style de Modula-2 (avec détection immédiate des erreurs de types inter-fragments, au contraire des modules de Standard ML, qui repoussent la détection de ces erreurs au moment de l'édition de liens).

Une année d'utilisation de cette implémentation a confirmé la viabilité du système de modules et a permis de le mettre à l'épreuve sur des programmes réalistes (le navigateur MMM, le *proxy* Web V6, le compilateur lui-même) et de le compléter sur certains points sous-estimés lors de l'étude théorique mais importants en pratique.

En parallèle, Xavier Leroy a poursuivi l'étude théorique de ce système de modules, en étudiant d'une part les conditions de complétude de l'algorithme de typage, et d'autre part en étendant la plupart de ses résultats antérieurs au cas des foncteurs applicatifs, une extension simple de son système qui augmente considérablement l'expressivité des foncteurs d'ordre supérieur [86].

Xavier Leroy a également publié une implémentation de référence de son système de modules, mettant en valeur sa grande indépendance vis-à-vis du langage de base [82].

3.3.2 Ajout d'une construction conditionnelle dans les modules

Participants : Xavier Leroy, François Rouaix, François Pessaux

Dans le cadre de son stage de DEA, François Pessaux a étudié l'ajout d'une construction conditionnelle dans le système de modules obtenu par Xavier Leroy, permettant aux modules paramétrés de discriminer sur les types effectifs qu'ils reçoivent en argument. Cette extension permet la définition de modules paramétrés "ad-hoc", qui sont l'équivalent au niveau des modules des fonctions génériques précédemment étudiées par Francois Rouaix et Pierre Weis au niveau du langage ML de base.

Une application immédiate de la construction conditionnelle dans les modules est la réalisation de bibliothèques génériques optimisées pour certains types d'arguments particuliers. À plus long terme, cette construction semble aussi nécessaire pour exprimer en termes de modules des hiérarchies de structures algébriques comme dans le système de calcul formel Axiom.

François Pessaux a formalisé cette extension du système de modules, a conçu un algorithme d'inférence de types de modules adapté, et en a réalisé une maquette.

3.4 Systèmes de types

L'équipe Cristal mène un certain nombre de travaux dans le domaine des systèmes de types. Le typage des objets et le sous-typage sont des domaines de recherche très actifs sur le plan international, à cause du succès des techniques de programmation par objets.

Nous menons aussi quelques travaux liés au polymorphisme, autorisant l'utilisation d'algorithmes génériques. Nous étudions les types intersection, qui forment un système de types très riche et très expressif, et dont nous explorons les liens avec les formes plus classiques de polymorphisme. Nous étudions aussi le polymorphisme extensionnel, qui autorise des programmes se comportant différemment selon le type de leurs arguments. Nous étudions enfin la possibilité de prouver formellement l'algorithme de typage polymorphe d'un noyau de ML.

Par ailleurs, nous étudions le typage d'autres langages, notamment celui du langage ASN.1, langage de spécification de données transitant sur le réseau, très utilisé dans le domaine des télécommunications.

3.4.1 Typage des objets

Participants : Didier Rémy, Jérôme Vouillon

Didier Rémy a continué à s'intéresser au typage des objets dans les langages fonctionnels. Bien que plusieurs solutions presque satisfaisantes aient déjà été proposées pour des langages explicitement typés, l'ajout d'objets dans les langages avec synthèse de types restait un problème difficile mais essentiel pour la crédibilité de ML.

Didier Rémy et Jérôme Vouillon ont conçu une extension de ML avec des objets [75]. La proposition reprend les idées du langage ML-ART, mais en fournissant maintenant des constructions primitives pour les objets et les classes. Cette proposition a été intégrée par Jérôme Vouillon dans le langage Objective Caml [80]. Objective Caml est le premier langage à objets fortement typé avec synthèse des types à être publiquement distribué.

Les choix de conception du langage Objective ML sont un compromis minutieusement équilibré entre l'expressivité, la simplicité et la conformité aux opérations standards. De ce fait, les constructions sur les objets d'Objective Caml sont très proches de celles que l'on peut trouver par exemple dans le langage Java. Mais à la différence de Java, les classes peuvent être naturellement paramétriques, et les types sont synthétisés. Simultanément le langage conserve la simplicité du langage ML. Cela fait d'Objective Caml un langage très attractif.

Le langage a toutefois une limitation qui peut a priori paraître importante : l'affaiblissement de l'interface des objets n'est pas automatique ; elle doit être demandée par une opération de sous-typage explicite. Cependant, le polymorphisme de l'envoi de messages réduit considérablement le besoin de sous-typage. De plus, si les travaux de Francois Pottier sur la synthèse du sous-typage s'avèrent positifs, ils devraient permettre de corriger ce défaut immédiatement. Parallèlement, Didier Rémy a travaillé en collaboration avec Jacques Garrigue (Université de Tokyo) sur un mécanisme autorisant des méthodes polymorphes dans les objets qui devrait réduire encore davantage le besoin de sous-typage.

Les recherches concernant les objets de ML vont se poursuivre dans principalement deux directions. D'une part, il est actuellement possible de cacher dans une sous-classe certaines variables d'instance d'une classe parente mais pas ses méthodes. Pour des raisons de symétrie, il est nécessaire de choisir entre simplifier davantage le language en renonçant aux deux aspects, ou les autoriser simultanément en perdant un peu en simplicité. D'autre part, le langage Objective Caml possède maintenant des objets et des modules expressifs. Il est devenu nécessaire de bien comprendre l'interaction entre ces deux paradigmes de la programmation à grande échelle, pour si possible les intégrer dans un mécanisme unifié.

3.4.2 Sous-typage

Participants : Didier Rémy, François Pottier

Didier Rémy s'est intéressé à la synthèse des types avec contraintes qui permet en particulier d'étendre le système de typage de ML avec une notion de sous-typage. Cette extension est très prometteuse et devrait s'appliquer également aux utilisations du typage pour effectuer des analyses statiques des programmes. Didier Rémy a étudié avec François Pottier les possibilités de simplification des contraintes, un problème crucial tant pour la lisibilité des types synthétisés que pour l'efficacité des algorithmes. L'effort porte sur la mise au point d'algorithmes de simplification des types inférés, suffisamment rapides pour être utilisables en pratique.

Didier Rémy a aussi repris ses travaux sur les enregistrements et montré comment le formalisme des types enregistrement permet, en présence de sous-typage, d'exprimer des informations plus précises. Par exemple, il est possible d'avoir une opération d'extension sur les objets en présence de sous-typage.

3.4.3 Types intersection

Participants : Michel Mauny, Émilie Sayag

Dans le cadre de sa troisième année de thèse, Émilie Sayag a poursuivi son étude d'un système de types basés sur une restriction des types intersection classiques.

En effet, dans la perspective d'obtenir un système de types qui donne une meilleure compréhension des liens entre β-réduction et typage polymorphe des termes du λ-calcul, Émilie Sayag avait proposé un système de types et montré plusieurs résultats concernant les termes en forme normale. S'appuyant sur ce travail, elle a étendu son étude aux termes généraux. Émilie Sayag a démontré la propriété de typage principal dans ce système pour tous les termes normalisables et normalisables en tête. Elle a ainsi retrouvé une propriété fondamentale des systèmes de types intersection classiques. De cette manière, Émilie Sayag obtient un système de types intersection conceptuellement plus simple, qui possède les mêmes propriétés que les systèmes classiques mais dont les preuves sont plus simples.

Cette nouvelle présentation des types intersection constituant un résultat complet, elle a fait l'objet de plusieurs publications. Les résultats préliminaires sur les formes normales ont été publiés dans les actes de la conférence FSTTCS'96 [77]. Une présentation longue de l'ensemble de ces travaux ont donné lieu à un rapport de recherche [85]. D'autre part, Émilie Sayag a complètement rédigé la première partie de sa thèse.

Pouvant désormais s'appuyer sur un formalisme et une base théorique solides, Émilie Sayag poursuit sa recherche d'un calcul sur les types permettant la mise à jour des liens entre β-réduction et typage polymorphe et ce qui rendra possible une reconstruction dynamique des types polymorphes dans un cadre général.

3.4.4 Polymorphisme extensionnel

Participants : François Rouaix, Pierre Weis, Jun Furuse

Jun Furuse est arrivé en France à la fin du mois de Septembre pour entamer une thèse au sein du projet Cristal. Cette thèse porte sur le *polymorphisme extensionnel*, une discipline de typage des langages fonctionnels qui permet l'écriture de programmes dont l'évaluation repose sur un calcul fondé à la fois sur la manipulaion des valeurs et des types. Ce calcul permettrait d'obtenir des langages d'une souplesse et d'une puissance inconnue actuellement pour les langages fortement typés statiquement. Le travail de Jun Furuse consistera à faire l'étude théorique et l'implémentation pratique de ce typage. Le premier travail va consister à ajouter à Caml des types à l'exécution, de façon à pouvoir disposer de valeurs à types dynamiques et à implémenter le filtrage sur les types qu'exige la sémantique des fonctions génériques du polymorphisme extensionnel.

3.4.5 Preuves formelles de synthétiseurs de types

Participants : Valérie Ménissier-Morain, Pierre Weis

L'algorithme W est l'algorithme de synthèse de types du cœur des langages ML. Son originalité se trouve dans la gestion du polymorphisme, qui consiste en la généralisation de types contenant des variables de type, ainsi qu'en l'instanciation de types polymorphes en instances particulières. La correction et la complétude par rapport au système de types de ML ont déjà été prouvées, mais jamais vérifiées par machine. Valérie Ménissier-Morain et Pierre Weis se sont attaqués à ce problème en collaboration avec Samuel Boutin (projet COQ) et Catherine Dubois (Univ. d'Évry).

Après une spécification commune du problème, Pierre Weis et Samuel Boutin d'une part, et Valérie Ménissier-Morain et Catherine Dubois, d'autre part ont choisi de poursuivre deux voies parallèles pour s'attaquer à la preuve de W en Coq.

V. Ménissier-Morain et C. Dubois ont choisi de partir d'une description fonctionnelle de l'algorithme, alors que P. Weis et S. Boutin sont partis d'une description prédicative. Chacune de ces deux approches possède ses avantages et ses inconvénients (dans le cadre de l'utilisation du logiciel Coq) : disponibilité immédiate et automatique de l'algorithme d'une part, une plus grande liberté dans le choix des inductions d'autre part.

Ces travaux sont toujours en cours de réalisation, et les deux approches sont actuellement confrontées au problème du traitement des schémas de types (c'est-à-dire les types polymorphes), qui forment l'originalité de l'algorithme W.

L'approche fonctionnelle est décrite dans [68, 78], et a obtenu une preuve de correction et de complétude dans le cas monomorphe. L'approche prédicative, modulo quelques axiomes qui écartent la difficulté de traitement des schémas de types égaux modulo α-conversion, a permis d'obtenir une preuve de correction dans le cas général.

3.4.6 ASN.1

Participants : Michel Mauny, Christian Rinderknecht

Dans sa troisième année de doctorat, Christian Rinderknecht a terminé la réalisation en Objective Caml d'un analyseur sémantique pour le langage ASN.1:1990 (standard ISO). Cet analyseur a été distribué publiquement (par `ftp`) et a été favorablement accueilli par les industriels (notamment la compagnie Southwestern Bell Technology Resources, Inc., qui a réutilisé ce travail pour en faire un navigateur de documents ASN.1). La partie sémantique de cet analyseur est fondé sur la formalisation en sémantique naturelle que Christian Rinderknecht a conçue pour ASN.1 sans sous-types. Il poursuit ses recherches pour spécifier le sous-typage et donner ainsi une formalisation complète de ASN.1:1990, allant de la syntaxe à la sémantique, en passant par le système de modules.

3.5 Arithmétique

Participants : Michel Mauny, Valérie Ménissier-Morain, Robert Harley

Les langages tels Caml sont de bons candidats à la programmation d'algorithmes dont on veut prouver formellement la correction. Les opérations fondamentales des mathématiques discrètes, d'algèbre et de théorie des nombres sont de bons candidats, et leur intégration au langage Caml jetterait les bases à un système de calcul formel fondé sur un langage de programmation expressif, efficace, et possédant de solides bases théoriques.

C'est dans ce contexte que Robert Harley commence son travail de thèse, poursuivant son travail de DEA durant lequel il a développé un algorithme de primalité. Pour son stage de DEA, Robert Harley a d'abord étudié l'algorithme des sommes de Jacobi dans sa version optimisée avec la théorie de Galois dans les anneaux. Par la suite, il a développé un nouvel algorithme, semblable au test de Hendrik Lenstra par sa simplicité et par la théorie sous-jacente, mais qui est utilisable en pratique. Robert

Harley a implémenté une maquette de cet algorithme ce qui a permis de vérifier sa rapidité sur des exemples de plusieurs centaines de chiffres.

3.6 Calcul distribué

3.6.1 Join-calcul distribué

Participant : Didier Rémy

Didier Rémy a participé au groupe de travail du projet Para sur les calculs distribués. Le but de ces travaux est de fournir un langage de programmation où la distribution est intégrée au noyau du langage, plutôt qu'ajoutée comme une couche externe. Cette intégration est indispensable, notamment pour pouvoir raisonner sur les programmes distribués. Le travail a abouti à la conception du join-calcul distribué [69], en ajoutant la distribution et une notion primitive d'erreurs au join-calcul[1]. Un prototype de ce langage a été réalisé par Luc Maranget et Cédric Fournet, membres du projet Para. Didier Rémy s'intéresse plus particulièrement aux problèmes de typage ainsi qu'aux objets distribués.

3.6.2 Calcul distribué et le langage Scheme

Participant : Christian Queinnec

C. Queinnec a poursuivi ses réflexions linguistiques sur la distribution de calcul de deux manières différentes.

D'une part il a proposé un nouveau modèle de macro-expansion fondé sur l'utilisation d'une tour de macro-expanseurs emboîtés [74]. Cette tour supprime la confusion de temps et d'espace qu'entretiennent de nombreux systèmes de macros avec leur langage cible. Ce modèle s'assortit d'une proposition d'environnement de première classe [72] permettant de décrire finement le comportement de multiples espaces de variables modifiables ainsi que leurs connexions. Ces deux propositions autorisent de multiples programmeurs (ou utilisateurs) à combiner finement et précisément des fragments de code éventuellement migrant ainsi qu'écrits dans des langages différents.

D'autre part, il a poursuivi l'implantation d'un système de mémoire partagée répartie, DMEROON. Ce système procure une bibliothèque de fonctions permettant de décrire des structures de données (pouvant contenir des pointeurs), de les manipuler et de les échanger au dessus d'Internet. Des capacités de description de ces structures en HTML sont intégrées au prototype qui peut jouer le rôle d'un serveur HTTP d'objets. L'implantation est réflexive et s'auto-décrit. La bibliothèque a été conçue pour être multi-lingue (C et Bigloo pour l'instant) et multi-usagers. Ce système n'est pas encore publiquement disponible.

Par ailleurs, C. Queinnec a reciblé le compilateur Scheme vers C, décrit dans son dernier livre [63], vers le langage Java. Le noyau d'un interprète de Scheme en Java a suivi et est disponible sur le réseau sous forme d'*applet*.

4 Actions industrielles

4.1 GIE Dyade

Participants : Valérie Ménissier-Morain, François Rouaix

Valérie Ménissier-Morain participe au projet de Dominique Bolignano qui utilise Coq pour la formalisation de protocoles sécuritaires pour le commerce électronique. François Rouaix est membre du projet

[1] Cédric Fournet et Georges Gonthier. « The Reflexive Chemical Abstract Machine and the Join-Calculus ». In *Proceedings of the 23rd ACM Symposium on Principles of Programming Languages*, pp. 372–385, janvier 1996.

VIP (*Verified Internet Protocols*) : c'est dans ce cadre que se situe son travail sur le code mobile en Caml (cf. section 3.2.3).

5 Actions nationales et internationales

5.1 Actions nationales

5.1.1 Activités éditoriales

Christian Queinnec est membre du comité de rédaction de la revue Techniques et Sciences Informatiques.

5.1.2 Contacts industriels

Nous avons des contacts réguliers avec le CNET-Lannion et la société Dassault, où le langage Objective Caml est utilisé pour certaines réalisations. Par ailleurs, nous sommes en contact avec l'Aérospatiale.

5.1.3 GDR Programmation

Le projet Cristal joue un rôle important dans le GDR Programmation, où Michel Mauny et Christian Queinnec sont chacun responsables d'un pôle :

Pôle Programmation Fonctionnelle du GDR
Michel Mauny est responsable de l'équipe "Typage et Programmation en ML" (INRIA, LRI, LIENS, LITP) et du Pôle "Programmation Fonctionnelle". Il est aussi membre de l'équipe de direction du GDR "Programmation".

Pôle Parallélisme et Distribution du GDR
Christian Queinnec est responsable de ce pôle, et membre de l'équipe de direction du GDR "Programmation".

Nous avons organisé nos cinquièmes "Journées de Pôle" à Orléans en novembre.

5.1.4 Autres responsabilités

Michel Mauny est responsable scientifique du Comité des Bourses de l'INRIA-Rocquencourt. Il a été membre du jury du concours de recrutement CR à l'INRIA-Rocquencourt.

Par ailleurs, Michel Mauny fait partie du groupe d'experts "Sciences et technologies de l'information" (DSPT4) auprès de la Mission Scientifique et Technique du MENESR.

Pierre Weis est membre du Comité de l'UR de Rocquencourt (CUR).

5.2 Actions internationales

5.2.1 Activités éditoriales

Xavier Leroy a été membre du comité de sélection du symposium ICFP 1997 (*International Conference on Functional Programming*).

Michel Mauny est membre du comité de lecture de la revue *Fifth Generation Computing*.

Didier Rémy a été membre du comité de sélection du symposium ICFP 1996 (*International Conference on Functional Programming*), d'ESOP 1996 (*European Symposium On Programming*), des JFLA 1996 (*Journées Francophones des Langages Applicatifs*), ainsi que de celui de l'atelier FOOL 96.

5.2.2 Contacts industriels

Sagem C. Queinnec a été consultant auprès de la société Sagem durant l'année 96.

CEA C. Queinnec a aussi été amené à participer à un groupe de travail du CEA sur la sémantique d'un langage d'extension de bibliothèques mathématiques.

François Rouaix a participé à la rédaction de l'étude de faisabilité du projet G7 Tel-Lingua (section sur le code mobile).

Par ailleurs, quelques industriels étrangers réalisent des développements importants en Objective Caml. Citons entre autres Microsoft Research (prototypage du système d'animation tridimensionnelle ActiveVRML), VLSI Libraries (conversions de descriptions de circuit), Next Solution (formatage de documents SGML utilisant les feuilles de style au standard ISO DSSSL), l'ENAC (développement d'algorithmes génétiques pour le contrôle du traffic aérien).

C. Queinnec a poursuivi sa participation au groupe de normalisation de l'AFNOR sur Lisp.

5.2.3 Europe

Christian Queinnec participe au réseau d'excellence "Parallel Virtual MultiComputer" (VIM) du programme HCM de la CEE. Ce réseau a été créé en 1993 et est animé par l'université de Bath. À ce titre, il a passé trois semaines à l'université de Southampton en compagnie de David de Roure et Luc Moreau.

5.2.4 Amérique

Xavier Leroy est membre invité du comité ML2000, qui réunit un certain nombre de spécialistes travaillant sur ML et la sémantique, dans le but de concevoir une version modernisée du langage ML intégrant les avancées récentes dans ce domaine. Xavier Leroy a participé aux réunions de janvier et août 1996 de ce comité.

6 Diffusion des résultats

6.1 Actions d'enseignement

6.1.1 Enseignement universitaire et grandes écoles

Universités – 3ème cycle

Guy Cousineau est responsable du DEA "Sémantique, Preuves et Programmation" (Paris 6-7-11, X, ENS), et y assure un cours de mise à niveau d'une semaine en programmation fonctionnelle.

Didier Rémy, Xavier Leroy et Michel Mauny donnent un cours intitulé "Typage et Programmation" dans le tronc commun de ce DEA (20 heures au total).

Xavier Leroy assure un cours d'option de ce DEA sur la compilation des langages fonctionnels et impératifs (20h).

École Normale Supérieure (Paris)

En collaboration avec Roberto Dicosmo (ENS-Paris), Didier Rémy a enseigné un cours d'initiation aux objets à l'École Normale Supérieure de la rue d'Ulm, en s'appuyant sur le langage Objective Caml (16 h).

École Polytechnique

Christian Queinnec a une demi-charge de Maître de Conférences à l'École Polytechnique où il enseigne l'interprétation de Scheme.

Michel Mauny, Didier Rémy et Pierre Weis sont Chefs de Travaux Dirigés d'Informatique à temps partiel à l'École Polytechnique. À ce titre, ils enseignent dans le tronc commun de première année (52h). Pierre Weis participe activement à l'introduction du langage Caml dans les cours d'informatique de l'École Polytechnique. Il a écrit une annexe de cinquante pages au polycopié du cours de tronc commun d'informatique. Pierre Weis y fait une introduction didactique au langage Caml et traduit en Caml les algorithmes du cours.

François Rouaix a participé à l'enseignement de Systèmes et Réseaux de la Majeure 2 d'Informatique de l'Ecole Polytechnique (20h).

CNAM

Pierre Weis est professeur associé au CNAM (contrat Barre) où il assure des cours de Caml en amphithéâtre avec 450 étudiants (2 heures hebdomadaires).

Écoles d'Ingénieurs

Michel Mauny a donné une semaine de cours de Programmation Fonctionnelle à l'ISIA (École des Mines de Paris, Sophia-Antipolis).

Universités – 1er et 2ème cycle

Valérie Ménissier-Morain a occupé un poste d'ATER à l'université d'Évry Val d'Essonne. Elle y a enseigné (cours/TD/TP) l'algorithmique avec Caml en licence de mathématiques (30h) ainsi que la programmation fonctionnelle et impérative en Caml en DEUG MIAS première année (52,5h de cours, 90h de TD). Ce dernier cours était nouveau et a donné lieu à la rédaction d'un support de cours de plus de 140 pages, en collaboration avec Catherine Dubois, qui devrait se transformer l'an prochain en un livre [87].

François Pottier est chargé de travaux pratiques d'algorithmique à l'Université Paris 7 (64h).

Christian Rinderknecht, dans le cadre de sa troisième année de monitorat, a enseigné l'algorithmique et la programmation en seconde année de DEUG (Paris 7) et en seconde année à l'IUP (Paris 7). (64 h)

Émilie Sayag a assuré des travaux dirigés de Caml Light à l'université d'Evry-Val d'Essonne (24h) au cours du premier semestre 1996.

Depuis septembre 1996, Émilie Sayag occupe un poste d'Attaché Temporaire d'Enseignement et de Recherche à l'université Paris 7.

6.1.2 Encadrement de stages, thèses

Michel Mauny encadre les thèses d'Émilie Sayag (en collaboration avec Guy Cousineau), de Christian Rinderknecht (en collaboration avec Bernard Lorho) et de Robert Harley.

Didier Rémy encadre les thèses de François Pottier et Jérôme Vouillon.

Pierre Weis encadre la thèse de Jun Furuse.

Xavier Leroy et François Rouaix ont encadré le stage de D.E.A. de François Pessaux.

Michel Mauny et Valérie Ménissier-Morain ont encadré le stage de DEA de Robert Harley.

Xavier Leroy a encadré le stage de magistère de Pascal Cuoq.

6.1.3 Jurys de thèse

Michel Mauny a été rapporteur des thèses de Rémi Douence (Rennes) et de Jean-Michel Moreno (Paris 7).

C. Queinnec a participé aux jurys de thèse de Paulo Ferreira (Paris 6), Younes El Amrani (Orléans), Serge Heiden (Paris 6) ainsi qu'au jury d'habilitation de Christine Paulin-Mohring (Lyon).

6.2 Participation à des colloques

POPL Xavier Leroy et Pierre Weis ont assisté au congrès *Principles of Programming Languages* (St Petersburg, Floride, 21–24 janvier 1996).

ICFP François Pottier, Christian Queinnec et Didier Rémy ont participé à la conférence ICFP (International Conference on Functional Programming) qui s'est tenue en mai à Philadelphie.

François Pottier y a exposé ses résultats[71]. A cette occasion, il a également rendu visite à Scott Smith et Valery Trifonov, de l'Université John Hopkins.

Christian Queinnec a présenté [72] à ICFP et a assisté au *Scheme Workshop*.

Didier Rémy était membre du comité de programme d'ICFP. Il a également assisté à l'atelier FLIC (Functional Languages in the Introductory Computing Curriculum) satellite de la conférence ICFP. À l'occasion de ce voyage, il a rendu visite à Carl Gunter à l'université de Pensylvanie où il a présenté ses travaux avec Jérôme Vouillon sur les objets.

FST-TCS Michel Mauny et Émilie Sayag ont participé à la conférence FSTTCS'96 à Hyderabad en Inde. Émilie Sayag y a présenté ses travaux sur la caractérisation du type principal des formes normales dans un système de types intersections.

TPHOL Valérie Ménissier-Morain et Catherine Dubois ont présenté leurs travaux sur la preuve de l'algorithme W en Coq lors de la conférence TPHOL'96 fin août à Turku (Finlande).

Reflection 96 Christian Queinnec s'est rendu à cette conférence (San Francisco, USA) où il y a présenté [74].

Reals Valérie Ménissier-Morain a participé à la conférence Reals'96 au début du mois d'avril au CIRM à Marseille.

FOOL Didier Rémy et Jérôme Vouillon ont participé à l'atelier FOOL qui s'est tenu en juillet à New Brunswick. Didier Rémy était membre du comité de programme, et Jérôme Vouillon y a présenté les résultats de son travail [75]. Jérôme Vouillon a profité de ce séjour aux États-Unis pour rendre visite à John Riecke aux laboratoires Bells où il a également exposé son travail, et il a assisté à la conférence LICS.

MPR Christian Queinnec a assisté aux Journées sur les Mémoires Partagées Réparties à Bordeaux où il a présenté son travail sur la bibliothèque DMEROON. [73].

Web Conference François Rouaix a participé à la cinquième conférence sur le *World Wide Web* (Paris, 6–10 mai 1996) où il a présenté son travail[76] sur le navigateur MMM et les applets Caml, et y a proposé une démonstration. Il a également participé au Workshop *Programming the Web*, ou son travail[91] sur le proxy V6 a été présenté par Bernard Lang.

ALEL Michel Mauny a participé au *Workshop on Compiler Techniques for Application Domain Languages and Extensible Language Models* qui s'est tenu à Linköping, au printemps, où il a présenté son travail commun avec Daniel de Rauglaudre [67].

JFLA Xavier Leroy et Pierre Weis ont participé aux Journées Francophones des Langages Applicatifs (Val-Morin, Québec, 28–30 janvier 1996). Xavier Leroy y a présenté un article sur le compilateur Caml Special Light.

Journées du GDR Environ la moitié de l'équipe a assisté aux Journées du GDR à Orléans. Jérôme Vouillon et François Pessaux y ont présenté leurs travaux.

6.3 Conférences invitées, tutoriels, cours, etc.

Didier Rémy a participé à la conférence LMO (Langage et Modèles à Objets) qui s'est tenue en octobre à Aigle, près de Lausanne. Il y a donné une conférence invitée sur la synthèse des types dans les langages à objets.

François Rouaix a présenté un exposé sur le *Code Mobile* au groupe de réflexion "Autoroute de l'information et Recherche" du MENESR.

Michel Mauny a participé à une table ronde sur le Génie Logiciel lors d'une rencontre Franco-Autrichienne organisée par le Ministère de l'Industrie à Linz, en Autriche. Il y a fait un exposé sur les retombées pratiques des études théoriques des systèmes de types.

Francois Rouaix a présenté un exposé sur la sécurité du code mobile lors de la journée "Le Web : axes de recherche, impacts sur la communication" organisée dans le cadre du grand séminaire de l'IRISA (8 novembre).

Pierre Weis a participé à la traduction d'un article consacré à Horus, publié dans le numéro de *Pour la Science* du mois de juillet 1996. Il a de plus rédigé un encadré dans l'article où il fait le point sur la recherche française en systèmes distribués et révèle les liens entre l'équipe américaine d'Horus et l'équipe de développement de Caml à l'Inria.

Valérie Ménissier-Morain a rendu visite à l'équipe *Automated reasoning* du laboratoire d'informatique de l'université de Cambridge dirigé par Lawrence Paulson. Durant cette semaine de la mi-mars, elle a exposé ses travaux sur l'arithmétique réelle exacte à l'université de Cambridge et à Imperial College à Londres.

Valérie Ménissier-Morain a été invitée par Abbas Edalat pour trois semaines en juillet 1996 au sein de "Theory and formal methods" d'Imperial College à Londres. Elle a aussi participé au groupe de travail inter-PRC "Arithmétique des ordinateurs et géométrie algorithmique" dirigé par Jean-Michel Muller.

Lors de l'atelier COMPROX, Valérie Ménissier-Morain a présenté ses travaux sur l'arithmétique réelle en précision arbitraire ainsi qu'une démonstration sur machine d'un prototype écrit en Caml.

Daniel de Rauglaudre a présenté le langage Chamau au CNET-Lannion.

François Rouaix a présenté un exposé récapitulatif sur *La surcharge dans les langages statiquement typés* au séminaire du LaMI, Université d'Evry (janvier 1996).

Didier Rémy a également participé au groupe de travail du projet Para sur les calculs distribués [69]. Il s'intéresse particulièrement aux problèmes de typage ainsi qu'aux objets distribués.

Michel Mauny et François Rouaix ont participé au Cebit de Hannovre (16–20 mars 1996), où ils ont proposé des démonstrations de MMM.

Michel Mauny a participé au Workshop LOGSEM, à Birmingham (Angleterre), afin de rencontrer les partenaires d'une proposition de *Working Group* actuellement à l'étude.

6.4 Animations scientifiques

Le séminaire Coq-Cristal-Para, organisé par Didier Rémy, a accueilli cette année une trentaine d'orateurs internationaux.

6.5 Diffusion de logiciels

Nos logiciels (Caml Light, Objective Caml, V6, MMM, Asno90) sont en accès libre par FTP anonyme à l'adresse suivante :

```
ftp://ftp.inria.fr/INRIA/Projects/cristal
```

et sont aussi accessibles sur le Web à l'URL :

`http://pauillac.inria.fr/cristal`

Les versions Macintosh et PC de Caml Light sont également distribuées sous forme de disquettes par le Centre de Diffusion de l'Inria. Les distributions de Caml sont régulièrement en tête du *hit-parade* des logiciels Inria transférés par FTP : environ 3000 transferts pour Caml Light et 2000 pour Objective Caml cette année, en très nette augmentation par rapport aux 1500 transferts de Caml Light de l'année dernière.

Environ 400 utilisateurs sont abonnés à la liste de distribution des utilisateurs de Caml `caml-list@pauillac.inria.fr`, gérée et modérée par Pierre Weis. Pierre Weis a écrit une centaine de pages Web bilingues consacrées à Caml, dont une FAQ (foire aux questions) et un ensemble de pages de présentation du langage.

En outre, Pierre Weis continue à suivre et à encourager les efforts des professeurs des classes préparatoires. Il envoie régulièrement des articles à la "Lettre de Caml", publication électronique de Laurent Chéno, professeur de Maths Sup. au Lycée Louis-le-Grand, consacrée à l'enseignement de Caml en classe préparatoires.

7 Publications

Livres et monographies

[63] C. QUEINNEC, *Lisp in Small Pieces*, Cambridge University Press, 1996.

Articles et chapitres de livre

[64] P. H. HARTEL, M. FEELEY, M. ALT, L. AUGUSTSSON, P. BAUMANN, M. BEEMSTER, E. CHAILLOUX, C. H. FLOOD, W. GRIESKAMP, J. H. G. VAN GRONINGEN, K. HAMMOND, B. HAUSMAN, M. Y. IVORY, R. E. JONES, J. KAMPERMAN, P. LEE, X. LEROY, R. D. LINS, S. LOOSEMORE, N. ROJEMO, M. SERRANO, J.-P. TALPIN, J. THACKRAY, S. THOMAS, P. WALTERS, P. WEIS, P. WENTWORTH, « Benchmarking Implementations of Functional Languages with "Pseudoknot", a Float-Intensive Benchmark », *Journal of Functional Programming 6*, 4, 1996.

[65] X. LEROY, « A syntactic theory of type generativity and sharing », *Journal of Functional Programming 6*, 6, 1996, à paraître.

[66] C. QUEINNEC, P. WEIS, « Programmation applicative, État des lieux et perspectives », *Technique et science informatiques 15*, 7, 1996, p. 1009–1013.

Communications à des congrès, colloques, etc.

[67] D. DE RAUGLAUDRE, M. MAUNY, « Chamau: an ML Dialect with Quotations, Grammars and Exensible Syntax », *in : Proceedings of ALEL'96: Workshop on Compiler Techniques for Application Domain Languages and Extensible Language Models*, avril 1996.

[68] C. DUBOIS, V. MÉNISSIER-MORAIN, « A Proved Type Inference Tool for ML: Damas-Milner within Coq (work in progress) », *in : Supplementary Proceedings of the 9th International Conference on Theorem Proving in Higher Order Logics: TPHOLs'96*, J. von Wright, J. Grundy, J. Harrison (réd.), Turku Centre for Computer Science, p. 15–30, août 1996. TUCS General Publication No. 1.

[69] C. FOURNET, G. GONTHIER, J.-J. LÉVY, L. MARANGET, D. RÉMY, « A Calculus of Mobile Agents », *in : Proceedings of the 7th International Conference on Concurrency Theory*, U. Montanari, V. Sassone (réd.), *LNCS*, *1119*, Springer, Pisa, Italie, août 1996.

[70] X. LEROY, « Le système Caml Special Light: modules et compilation efficace en Caml », *in : JFLA 96 – Journées Francophones des Langages Applicatifs*, p. 111–131, janvier 1996.

[71] F. POTTIER, « Simplifying subtyping constraints », *in : Proceedings of the 1996 ACM SIGPLAN International Conference on Functional Programming (ICFP '96)*, p. 122–133, janvier 1996.

[72] C. QUEINNEC, D. DE ROURE, « Sharing Code through First-class Environments », *in : Proceedings of the 1996 ACM SIGPLAN International Conference on Functional Programming (ICFP '96)*, p. 251–261, Philadelphia (Pennsylvania, USA), mai 1996.

[73] C. QUEINNEC, « Bribes de DMEROON », *in : Actes des journées de recherche sur la Mémoire Partagée Répartie — MPR 96*, C. Toinard (réd.), p. 51–56, Bordeaux (France), mai 1996.

[74] C. QUEINNEC, « Macroexpansion Reflective Tower », *in : Proceedings of the Reflection'96 Conference*, G. Kiczales (réd.), p. 93–104, San Francisco (California, USA), avril 1996.

[75] D. RÉMY, J. VOUILLON, « Objective ML: A simple object-oriented extension to ML », *in : Proc. 24th symposium Principles of Programming Languages*, ACM Press, janvier 1997. À paraître.

[76] F. ROUAIX, « A Web navigator with applets in Caml », *in : Proceedings of the 5th International World Wide Web Conference, Computer Networks and Telecommunications Networking*, *28*, 7–11, Elsevier, p. 1365–1371, mai 1996.

[77] E. SAYAG, M. MAUNY, « Characterization of principal types of normal forms in intersection type system », *in : Proceedings of Foundations of Software Technology and Theoretical Computer Science*, 1996.

Rapports de recherche et publications internes

[78] C. DUBOIS, V. MÉNISSIER-MORAIN, « A Proved Type Inference Tool for ML: Damas-Milner within Coq (work in progress) », *Rapport de recherche n°19-1996*, Université d'Évry Val d'Essonne, Laboratoire de Mathématiques et Informatique, avril 1996.

[79] R. HARLEY, *Un algorithme simple et efficace pour démontrer la primalité*, Rapport de DEA Sémantique, Preuves et Programmation, Université Paris 7, 1996.

[80] X. LEROY, D. RÉMY, J. VOUILLON, *The Objective Caml system, documentation and user's manual – release 1.02*, INRIA, octobre 1996, documentation distribuée avec le système Objective Caml.

[81] X. LEROY, *The Caml Light system, release 0.71 – Documentation and user's manual*, INRIA, février 1996, documentation distribuée avec le système Caml Light.

[82] X. LEROY, « A modular module system », *Rapport de recherche n°2866*, INRIA, avril 1996.

[83] M. MAUNY, *Functional programming using Caml Light (version 0.71)*, INRIA, février 1996, documentation distribuée avec le système Caml Light.

[84] F. PESSAUX, *Ajout de conditionnelle dans un système de modules à la Caml Special Light*, Rapport de DEA Sémantique, Preuves et Programmation, Université Paris 6, 1996.

[85] E. SAYAG, M. MAUNY, « A presentation of the intersection type discipline through principal typings of normal forms », *Rapport de recherche n°2998*, INRIA, 1996.

Divers

[86] X. LEROY, « Manifest types, modules, and separate compilation », soumis à publication aux *ACM Transactions on Programming Languages and Systems*, en cours de révision, mai 1996.

[87] V. MÉNISSIER-MORAIN, C. DUBOIS, « Cours de programmation : illustration en Caml », en préparation, 1997.

[88] V. MÉNISSIER-MORAIN, « Arithmétique exacte en Caml », *La lettre de Caml* numéro 3, pages 11–17, mars 1996.

[89] V. MÉNISSIER-MORAIN, « Arbitrary precision real arithmetic: design and algorithms », soumis au *Journal of Symbolic Computation*, septembre 1996.

[90] F. PESSAUX, «Ajout de conditionnelle dans un système de modules à la Caml Special Light», journées du GDR Programmation, novembre 1996.

[91] F. ROUAIX, B. LANG, «The V6 Web Engine», WWW5 workshop: Programming the Web - a search for APIs, mai 1996.

8 Abstract

The objective of the Cristal group is to study type systems for programming and to develop statically-typed programming languages and tools for typed programming.

Our main current topics are: design and development of the Caml programming language, the study of the usage of statically-typed programming languages for developing Web-related tools, and true theoretical study of types : objects, subtyping and polymorphism.

The Caml programming language is a statically-typed functional programming language with a non-obtrusive type system (types are inferred by the compiler), polymorphism, imperative features and a powerful exception mechanism. Caml-Light is a portable lightweight implementation, well suited for teaching and is widely used in undergraduate and graduate curricula. Caml-Light is available under Unix, MacOS and Microsoft Windows. During 1996, we started distributing Objective Caml, which features an expressive module system, an extremely efficient native code compiler, and a class-based object layer, compatible with type inference.

We have developed and distributed a Web browser called MMM and a personal Web proxy called V6, demonstrating that efficient, large-scale applications can be programmed in Caml. The originality of MMM is that it accepts Caml programs as applets, where static typechecking and cryptographic tools are used to ensure that applets are harmless.

We study type systems for objects and subtyping, and use research results for improving the Objective Caml object system. Polymorphism is one of our other interests, particularly extensional polymorphism and intersection type systems, and their relationship with lambda-calculus and beta-reduction. We are also working on mechanical proofs of type inference algorithms.

All our developments, as well as a description of our activities, are available on our Web server : `http://pauillac.inria.fr/cristal`.