Projet INTERLOO

Pierre Cointe & Roger Rousseau

1 Description du projet

L'avenir du Génie Lociciel réside dans la définition de bibliothèques de composants logiciels réellement partageables et interchangeables et ce indépendamment du langage avec lequel ils ont été conçus. Bien sûr idéalement - au niveau le plus externe - ces bibliothèques seraient manipulables au travers d'un unique environnement de programmation interactif incluant interprètes, flâneurs et inspecteurs génériques. De même - au niveau le plus interne - des objets persistants constitueraient une base de données unique interfaçable avec les différents langages.

L'objectif de ce projet est un premier pas dans cette direction. Il s'agit d'étudier puis de proposer des mécanismes d'interopérabilité. Cette étude porte sur les architectures logicielles nécessaires à l'interopérabilité, sur la prise en compte de l'interopérabilité dès les phases d'analyse et de conception, sur le cas concret de deux langages présentant un intérêt industriel mais également représentatifs des problèmes techniques à résoudre: Eiffel et Smalltalk.

1.1 Responsable

Pierre Cointe, École des Mines de Nantes, 4 rue Alfred Kastler, La Chantrerie, 44070 Nantes Cedex 03, Téléphone: +33 51 85 82 00, Télécopie: +33 51 81 89

1.2 Equipes

- UNSA (Univ. de Nice - Sophia Antipolis) & I3S (URA CNRS 1376) - Equipe OCL (Objets et Composants Logiciels).
- EMN - Ecole des Mines de Nantes

1.3 Résumé

L'un des enjeux majeurs dans le domaine de la fabrication du logiciel est la possibilité de produire des composants logiciels à flux tendu. A court terme, les techniques objets sont les seules permettant d'envisager sérieusement une production à flux tendu qui respectent par ailleurs le critère de qualité. Ces techniques objets se basent essentiellement sur le principe de la réutilisabilité des composants, que ces composants soient définis dans un même langage, ou dans des langages différents.

Le but initial de ce projet consistait à étudier l'interopérabilité entre les langages Eiffel et Smalltalk. Comme le montre la suite de ce texte, les principaux résultats portent sur des modèles d'évolution de ces deux langages permettant d'envisager à moyen terme un modèle à objets "unifié" qui satisfasse au mieux les exigences du génie logiciel et qui soit supporté efficacement par ces deux langages. A plus long terme, ce modèle unifié devrait permettre l'assemblage de composants décrits dans des langages différents.

1.4 Animation scientifique

Au niveau scientifique, les membres du projet jouent un rôle majeur dans l'animation de la communauté objet nationale et internationale. Ils participent activement aux différentes conférences, écoles d'été et revues sur les objets.

Au niveau industriel, les relations des membres du projet avec les sociétés ISE (Eiffel), O2 Technology (O2) et OTI (Smalltalk) leur permettent de participer activement au transfert de technologie vers l'industrie.

2 Description des sous-projets

2.1 SLO – Systèmes et langages à objets

2.1.1 Responsable

Pierre Cointe, professeur École des Mines de Nantes

2.1.2 Chercheurs impliqués

Isabelle Borne, Annya Romanczuck (Maîtres assistante), Philippe Mulet (chercheur OTI), Noury Bouraqadi, Thomas Ledoux, Frédéric Rivard (Doctorants).

2.1.3 Thème et objectif scientifiques

Le projet de "l'équipe objet" de l'École des Mines de Nantes s'inscrit dans le cadre général des technologies à objets. Il étudie plus particulièrement les langages et leur environnement de programmation pour s'intéresser à leurs implémentations, à leurs extensions et à leur mise en œuvre. Les principaux thèmes abordés sont les suivants :

- étude de la réflexivité comme outil de description, d'implémentation et d'extension des langages à objets et de leurs bibliothèques,
- étude des problèmes liés à l'inter-opérabilité entre langages à objets,
- étude de la composabilité des (méta)-objets [MMC95],
- étude d'outils d'aide à l'analyse et à la compréhension de bibliothèques de classes [Borne95],
- étude des langages à prototypes par opposition aux langages de classes (cf. projet ELO [Mul95]).

Nous ne développerons dans ce rapport que les deux premiers points qui sont directement liés au projet INTERLOO du Gdr.

2.1.4 État d'avancement

Sous-projet ClassTalk ClassTalk est une extension de Smalltalk conçue d'abord comme l'ajout à Smalltalk du modèle d'ObjVlisp de P. Cointe & J.P. Briot [Coi93] Ce projet a ensuite évolué vers la réalisation d'une bibliothèque de métaclasses explicites permettant de disposer de différents modèles de classes. Ces différents modèles constituent autant de variations sur les thèmes de l'encapsulation, de la représentation des variables d'instance, de l'héritage, de l'agrégation, de la résolution de l'envoi de messages . . . .
Ces modèles peuvent également être vus comme des "patterns" définissant des règles de construction pour des classes possédant des propriétés explicites. Partant de ce constant, nous avons menés les études suivantes :

- rétro-conception manuelle de hiérarchies de classes Smalltalk (Boolean, Magnitude, ...) en utilisant des métaclasses explicites ClassTalk [LeCo96],

- réalisation de métriques sur l'ensemble des classes de l'environnement Smalltalk standard pour mesurer les propriétés implicites des classes en tant qu'objet (par exemple, classes non instanciables, classes abstraites, classes non derivables, classes ayant une seule instance, . . .). Ces mesures valident à posteriori l'idée d'utiliser des métaclasses explicites [Bour95],

- premières propositions pour élaborer un environnement de programmation exploitant les propriétés des classes explicitement déclarées dans leur métaclasses.

Sous-projet ClassTalk-Ada95 Ce projet mené en collaboration avec la société Object Technology International Inc (OTI) a consisté à définir en ClassTalk un sous-ensemble du langage Ada95 directement exécutable par les machines virtuelles Smalltalk.

Ce projet avait trois buts. Il s'agissait avant tout d'expérimenter les capacités d'extensibilité du compilateur Smalltalk (réifié sous forme d'un objet), de proposer un système de types pour le langage Smalltalk et de mettre en oeuvre la plate-forme **Envy/ClassTalk** sur un exemple significatif.

Le compilateur et l'analyseur syntaxique Smalltalk étant représentés par des hiérarchies de classes, nous avons pu les étendre pour proposer de nouvelles formes d'envoi de messages, réaliser un premier mécanisme d'inférence de types, et finalement traiter directement des expressions Ada95 [Riv95a] [RC95].
Par ailleurs, cette modélisation d'Ada95 a été l'occasion d'enrichir de manière très significative les bibliothèques de métaobjets de ClassTalk. La description de cette réalisation constituera l'un des chapitres de la thèse de F. Rivard.

Sous-projet Extension réflexive du langage Smalltalk Le but de ce projet est d'abord de répertorier les différents aspects réflexifs du système Smalltalk et d'en proposer une première taxinomie. Dans [Riv96b] [Riv96c] nous reprenons la classification de P. Deutsch pour distinguer les classes agissant sur la structure et le comportement des objets, le contrôle de l'exécution, la gestion de la mémoire, la gestion de l'espace de nommage, l'environnement de programmation, l'organisation des classes dans le flâneur, le processus de compilation (analyse syntaxique et génération de codes).

Il s'agit ensuite d'utiliser ces aspects réflexifs pour étudier deux extensions du système Smalltalk :

1. Extension du compilateur pour introduire de nouvelles sémantiques d'envoi de messages (par exemple, introduction de pré-post conditions à la Eiffel [Riv95a]) mais également permettre l'exécution - par la machine virtuelle Smalltalk - d'autres langages (en particulier Ada95 [RC95]),

2. réification du lien d'instanciation et possibilité de changement dynamique de la classe d'un objet [Riv96a].

Ce projet constitue le corps de la thèse que F. Rivard doit soutenir en décembre 1996. Celle-ci décrira également **Militsa**, une extension de ClassTalk supportant le changement dynamique de classe.

Sous-projet SOM - DSOM - SomTalk Les principaux constructeurs informatiques dont SUN et HP ont créé (1991) un consortium (The Object Management Group) devant produire un modèle objet unifié pour l'inter-opérabilité de réseaux hétérogènes répartis. Leur but principal étant de permettre aux industriels d'échanger des composants logiciels distribués, réutilisables et intégrables. Les principaux résultats de ce consortium portent sur la définition du langage de description d'interface IDL (Interface Description Language) et de l'architecture CORBA (a Common Object Request Broker Architecture).

En 1992, IBM a proposé le modèle SOM (a System Object Model) compatible avec IDL et avec CORBA dans sa version distribuée (DSOM). SOM est aujourd'hui disponible sur le marché sous la forme d'un produit (le SOMObjects Developer Toolkit). Ce produit est plus particulièrement destiné à l'élaboration de bibliothèques de composants (classes) binaires pouvant être partagées par différents langages et en particulier C et C++. Le modèle SOM permet également d'inter-opérer avec IBM-Smalltalk et le système VisualAge.

D'un point de vue recherche, le modèle SOM est d'autant plus intéressant qu'il met en œuvre une architecture de métaobjets très proche de celle d'ObjVlisp et de ClassTalk. Dès lors, l'inter-opérabilité entre C++ et Smalltalk devrait se trouver largement facilitée par l'utilisation de ClassTalk. En effet, ce dernier dispose déjà du modèle de métaclasses explicites ainsi que du modèle d'héritage multiple de SOM.

Une première expérimentation de couplage SOM - ClassTalk a été réalisée par D. Barbaron dans le cadre de son stage de DESS [Bar94] [MLBRC94]. La suite de cette expérimentation et son compte-rendu constitue une partie du travail de thèse de T. Ledoux.

Cette année nous avons également expérimenté la version distribuée de SOM: DSOM. Pour ce faire nous sommes partis d'un "framework" test constitué de trois classes développées en C et qui sont utilisables indifféremment en C++ et/ou Smalltalk dans l'environnement OS/2. Cette expérience met en oeuvre l'inter-opérabilité entre ces trois langages. La prochaine étape sera de tester cette inter-opérabilité au travers de trois systèmes différents: OS/2, AIX et Windows.

2.1.5 Relations nationales et internationales

- Collaborations régulières avec : Jacques Malenfant (Université de Montreéal), Gregor Kiczales (Xerox Parc), Akinori Yonezawa, Satoshi Matsuoka (Université de Tokyo), Mario Tokoro (Keio Univiersity & Sony Lab), Trevor Hopkins (Manchester University),
- Organisation de workshops et conférences : Programme Chairman d'ECOOP'96, président de l'Association Internationale pour les Technologies Objets (AITO), co-organisateur du workshop de l'IJCAI "On Reflection and Meta Level Architecture and their Application in AI" [ICCFM95], co-organisateur de la conférence Objet'96 à Nantes (Mai 96).
- Participation à des comités de programmes de conférences sur les objets : ECOOP (1994, 1995, 1996), OOPSLA (1996), ISOTA(1996), REFLECT'96, LMO (1993, 1994, 1995), Numéro spécial de TSI Objets (à paraître en 1996), Revue l'OBJET . . . ,
- Réalisation de séminaires de formation dans des Écoles d'été : Sixth European Summer School in Logic, Language and Information (ESSLI'94) [Coi94b], MCSEAI'94 [Coi94a], European Smalltalk Users' Group Summer School (ESUG) [Coi95a],
- Participation à des tutoriaux et tables rondes : TOOLS'96 [Coi96a], Forum Objet'95 [Coi95d], Stratégies Objets'95 [Coi95c].

2.2 OCL – Objets et Composants Logiciels (I3S)

2.2.1 Responsable

Roger Rousseau, Maître de Conférences,

2.2.2 Chercheurs impliqués

Robert Chignoli, Philippe Lahire (Maîtres de Conférences), Philippe Brissi, Philippe Collet (Doctorants) Ces chercheurs sont tous membres du thème OLEM (Objets, Langages, Environnements et Méthodes) du laboratoire I3S de Sophia Antipolis (URA CNRS/UNSA 1376).

2.2.3 Thème et objectif scientifiques

Cette équipe travaille sur les *apports de la technologie par objets au génie logiciel, étendu au domaine des bases de données* [Rou95], c'est-à-dire principalement le cas de *la programmation par réutilisation de composants*, dans un cadre volatile ou persistant.

L'approche repose sur l'expérience pratique acquise par l'équipe avec le langage Eiffel depuis sa création, tant sur le plan de l'enseignement que celui de développements industriels importants. Quatre types de problèmes sont considérés en priorité :

- les *aspects sémantiques* pour l'expression (spécification, requêtes de sélection...), la documentation et le contrôle des programmes,
- l'*évolution des composants*, des bibliothèques, des applications et des objets composites persistants,
- l'*efficacité*, aussi bien pendant les phases de construction que d'utilisation des logiciels (applications ou bibliothèques),
- la *structuration* (héritage, encapsulation, généricité) et sa perception selon différents points de vue.

L'objectif à moyen terme est de *définir un modèle à objets* qui satisfasse au mieux les exigences du génie logiciel, des bases de données — voire de la CAO — et d'étendre au besoin le langage Eiffel pour faciliter l'*interoperabilité* de ces domaines.

2.2.4 État d'avancement

De 1991 à 1994, l'essentiel des activités de recherche de cette équipe s'est déroulée dans le cadre de trois projets internationaux qui ont nécessité des développements de logiciels d'assez grande taille, au total environ 1 100 classes développées, 160 000 lignes de code :

FLOO : pour la définition d'un modèle de persistance pour le langage Eiffel, avec l'implémentation d'un prototype par un couplage fort entre Eiffel et O2 (Contrat européen ESPRIT II 5311 *"Business*

Class").

IREC: pour la définition d'un modèle d'évolution de classes Eiffel, d'instances persistantes, de bibliothèques et d'applications, avec l'implémentation d'un prototype de réactualisation semi-automatique des entités qui dépendent de celles qui évoluent, en utilisant une base de connaissances qui réifie toutes les constructions du langage Eiffel (Contrat européen ESPRIT II 5311 *"Business Class"*).

K2: pour la conception et l'implémentation d'un logiciel de CAO appliqué à l'économétrie pour la prévision des besoins alimentaires des pays du Tiers Monde (Contrat avec la FAO de l'ONU).

Depuis 1995, l'équipe exploite l'expérience pratique acquise dans les conceptions et les développements des années précédentes pour développer des modèles plus théoriques.

Seules les activités plus directement liées au projet *INTERLOO* sont présentées dans ce rapport, essentiellement les projets *FLOO*, *IREC* et un ensemble de travaux préliminaires à la définition d'un modèle à objets, commun pour le génie logiciel et les bases de données, appelé *GOODY*. Le lecteur trouvera dans les rapports d'activité du laboratoire I3S (1992, 1994, 1996) des informations complémentaires sur les autres activités de cette équipe.

Sous-projet OCL/FLOO. Ce projet propose un modèle de persistance intégré au langage Eiffel qui satisfait les règles d'Atkinson pour les bases de données, et d'autres nécessaires au génie logiciel pour les besoins de la réutilisation et de la fiabilité, lesquels sont contrôlés par le typage statique, les assertions et les exceptions.

La gestion des échanges entre les mémoires volatile et persistante est transparente au programmeur, sans modification du langage et sans la nécessité d'utiliser des classes ou des méthodes spécifiques. Les accès sont réalisés de manière automatique et incrémentale, soit de manière navigationnelle par redéfinition de l'opérateur d'accès (en Eiffel, la notation pointée `a.x.g(k).h...`), soit par des requêtes sélectives unifiées avec les méthodes générales de parcours des collections génériques de la bibliothèque Eiffel. Le statut de persistance s'obtient par des requêtes explicites sur des racines de persistance, ou implicitement par "contagion" du statut de persistance. La destruction physique des instances est retardée, réalisée par les ramasse-miettes volatiles et persistants. Enfin, *FLOO* repose sur un système transactionnel de type clients/serveurs, pour prendre en compte les accès simultanés, le partage d'objets et la gestion des reprises en cas d'incidents.

Le prototype implémenté pour le projet *FLOO* s'appuie sur le SGBD O2. Cette solution a l'inconvénient d'une redondance des schémas (Eiffel et O2) et a une certaine lourdeur de mise en oeuvre : traduction complète des classes Eiffel avec leurs méthodes, en C par le compilateur Eiffel, puis en O2 par le compilateur FLOO[1]. Le bénéfice est cependant clair pour ce qui est de l'efficacité et de l'interopérabilité entre des applications écrites en Eiffel et dans d'autres langages couplés avec O2, notamment C++.

Par rapport aux modèles de l'OMG (*ODMG'93, ODMG'95*), le modèle *FLOO* va plus loin pour ce qui est de la transparence des accès et permet un traitement beaucoup plus efficace des instances volatiles, tout en utilisant les mêmes notations pour manipuler les instances volatiles ou persistantes. Cependant, il peut encore être amélioré pour le traitement des associations et de la sélection des champs. Ce travail se poursuit actuellement par une réflexion sur l'utilisation de vues pour étendre les mécanisme d'héritage d'Eiffel et permettre l'expression des sélections de champs (deux stages de DEA en 94 et 95, et démarrage de la thèse de Pierre Crescenzo en novembre 1995).

Ce projet a conduit à la thèse de Philippe Lahire en 1992, à de nombreux rapports techniques, séminaires de présentation, démonstrations et comités d'évaluation, notamment dans le cadre du projet ESPRIT, à plusieurs publications dans des conférences [CFLR93, CFLR93, CFLR95b] et à un article soumis à TSI pour le numéro spécial sur les objets, qui devrait paraître en 1996 [CFLR95a].

Sous-projet OCL/IREC. Ce projet vise la gestion de l'évolution d'entités logicielles interdépendantes, développées dans un monde Eiffel persistant : classes, instances persistantes, bibliothèques, applications.

L'étude de l'état de l'art sur les problèmes d'évolution montre que la plupart des travaux portent sur l'identification, la mémorisation, et la gestion des variations (gestion de versions ou de configurations), dans un cadre sémantique faible, ce qui explique un manque d'aptitude pour le problème fondamental de la *réactualisation des entités dépendantes.*

1. Essentiellement l'oeuvre de Jacques Farré, de l'équipe "langages" du thème OLEM de l'I3S

Pour réactualiser, de la manière la plus automatiquement possible, les entités qui dépendent de changements, il faut un cadre qui permette :

– une perception des changements à un grain assez fin, éventuellement jusqu'au niveau des instructions des routines,
– une compréhension de la signification des changements et leurs conséquences sur les objets dépendants.

Le langage Eiffel offre un cadre bien adapté à ces deux prérequis :

– En tant que langage à objets, il donne la possibilité de réifier n'importe quelle construction ou concept du langage. Cela permet une expression fine des des changements *(deltas)* en exploitant l'identité des objets, ce qui manque aux outils traditionnels d'expression des deltas, tels que la commande `diff` de Unix.
– Sur le plan sémantique, le langage Eiffel a une approche pragmatique originale qui repose en grande partie sur l'utilisation d' *assertions exécutables* embarquées dans le code. À défaut de proposer une sémantique complète comme dans le cas des spécifications algébriques, la technique des assertions à le mérite d'être à la portée — aussi bien en lecture qu'en écriture — de la plupart des programmeurs et peut donc s'utiliser *systématiquement*, même pour des codes volumineux comme celui du projet *K2*.

Dans le projet *IREC*, les assertions jouent un double rôle pour :

– comprendre l' évolution de classes et la compatibilités des entités qui en dépendent, grâce à la sémantique de l'héritage renforcée par les assertions placées dans le code des applications,
– exprimer les métarègles sémantiques du langage Eiffel, grâce aux assertions placées dans les classes qui réifient toutes les constructions d'Eiffel.

Le caractère partiel des assertions d'Eiffel et leur mode d'expression ne permettent pas d'expliquer tous les cas d'évolution, mais on peut déjà interpréter de nombreux cas, parmi les plus fréquents. Pour réifier toutes les constructions d'Eiffel, ce qui constitue une base de métaconnaissances, le projet *IREC* a adapté un interprète du langage Eiffel, écrit en Eiffel en 1991 dans cette équipe. Cette réification n'utilise pas de métaclasse, concept absent de façon explicite en Eiffel, mais construit des vues complètes ou partielles, selon divers points de vue, par des classes ordinaires en Eiffel qui jouent le rôle de **composants de langage**.

Les instances des composants *IREC* (environ 250 pour Eiffel) décrivent ainsi, non seulement l'équivalent d'un arbre de syntaxe abstraite décoré d'attributs sémantiques, mais aussi les méthodes des principales actions de manipulations des composants logiciels durant leur cycle de vie : expression des *"deltas sémantiques"*, méthodes de réactualisations semi-automatiques, évaluation des assertions, interprétation du code, calcul de vues, etc... Cette approche *"tout objet"*, qui rassemble au niveau de chaque *composant de langage* toute la connaissance et le savoir faire en matière de manipulation, se démarque des approches classiques comme Centaur qui manipulent les programmes de manière globale à partir de métadescriptions syntaxico-sémantiques dans des langages spécifiques (par exemple Typol).

La réification *IREC* du langage Eiffel a d'autre part été réécrite en Smalltalk par l'équipe de Jean-François Perrot du Laforia, à partir de la version de l'I3S. L'objectif était d'explorer des possibilités d'interopérabilité entre les environnements Eiffel et Smalltalk, notamment pour bénéficier des outils et du savoir faire de l'équipe du Laforia dans le domaine de l'intelligence artificielle (notamment le système de règles NeOpus), pour une explication plus fine de l'évolution, des contrôles statiques renforcés, un optimisation plus poussée du code...

En dehors des communications au sein du projet ESPRIT *Business Classs* (rapports techniques, démonstrations, comités d'évaluation) et de présentations diverses en séminaires, le projet IREC, n'a encore fait l'objet que d'une publication [BR95b], car il est momentanément arrêté depuis septembre 1994, suite à la maladie de Philippe Brissi qui est tragiquement décédé en mai 1995.

Sous-projet OCL/GOODY. L'expérience acquise dans les projets *FLOO*, *IREC* et *K2* met en évidence l'intérêt de disposer d'un modèle à objets général qui étende les possibilités actuelles du langage Eiffel, afin qu'il convienne mieux, dans un premier temps aux domaines du génie logiciel et des bases de données, puis progressivement aux domaines de la CAO et de l'IA.

Pour préparer cet objectif ambitieux, nous avons commencé quelques actions préliminaires :

– l'étude de divers modèles à objets, notamment ceux de ODMG (1993 et 1995),

- la participation au workshop OOPSLA'94 sur le thème des moteurs à objets organisé par Fred Cummins [RBC94a],
- la réalisation d'un cours de DEA sur le thème d'un modèle commun pour le GL, les BD et la CAO,
- un début de réflexion sur les problèmes d'efficacité, particulièrement sur l'exploitation de la persistance et de nouvelles possibilités d'introspection pour donner plus de souplesse au cycle de vie des composants et permettre une meilleure efficacité des contrôles et des exécutions. Cette réflexion se poursuit actuellement avec la collaboration de Jean-Claude Royer (IRIN).
- enfin, le démarrage de la thèse de Philippe Collet en octobre 1994, sur le thème des assertions en Eiffel. Il s'agit en particulier d'étendre le langage assertionnnel d'Eiffel et d'opérer un rapprochement entre la notion d'invariant de classe et celle de critère de sélection dans les bases de données.

2.2.5 Relations nationales et internationales

- FAO/ONU (*Food Agriculture Organization of the United Nations*, depuis 1991), projet *K2*) : conception et implémentation d'un logiciel de CAO pour la prévision des besoins alimentaires des pays du Tiers Monde. Deux des principaux modules de ce logciel ont été livrés en décembre 1995 (800 classes, 100 000 lignes de code en Eiffel 3). La nature de ce projet est plus du développement que de la recherche, mais présente l'intérêt de permettre le test en vraie grandeur de nos outils pour l'évolution et d'alimenter la réflexion pour le modèle *GOODY*[LJ95].
- Projet Européen ESPRIT II N° 5311 *Business Class* (1991-1994), avec huit autres sociétés ou laboratoires français, britanique, italiens et espagnols : Télésystèmes et SOL (F), ALR (UK), Etnoteam et Datamont (I)...
- Bertrand Meyer (ISE, Santa Barbara, Cal.) et Jean-Marc Nerson (SOL, Paris) : discussions sur le langage Eiffel (réunions, mails, news...), consultations, séminaires de formation continue, partenariat dans le projet *Business Class* précédent.
- Participation à des comités de programmes de conférences sur les objets : Tools'Europe (1993,1994,1995), LMO (1994, 1995), Numéros spéciaux de TSI Ada [Rou94] et Objets (à paraître en 1996).
- Collaborations avec des chercheurs du pôle Objets du GDR de programmation : Jean-François Perrot (Laforia, Paris) pour le projet *IREC* et Jean-Claude Royer (IRIN, Nantes) pour le début d'une réflexion sur les problèmes d'efficacité.
- IUT de Vannes (Daniel Devaux), Université de Fribourg (Jacques Pasquier-Boltuck), Technische Hochschule Darmstadt (Hans-Jürgen Hoffmann) [BR95a]: consultations sur l'enseignement d'Eiffel.
- Réalisation de séminaires de formation dans des Écoles d'été : Fribourg (1993), Vannes (1993), Nice (INRA, 1994).
- Séminaires de recherche : GDR de programmation (pôle objets, Orsay 1993, Grenoble 1995), Groupe AFCET Modélisation des objets en cours de construction (Paris, 1994), ENSIMAG (Grenoble 1994).

Références

[BR95a] Jean-Claude Boussard & Roger Rousseau. Why and How to teach Eiffel in Software University Education: Our Experience at UNSA for Seven Years. In Hans-Jürgen Hoffmann (Ed.), *Eiffel in Schule und Hochschule; Ausbildung in objektorientiertem Programmieren*, Technische Hochschule Darmstadt, `http://www.pu.informatik.th-darmstadt.de/eis/`, Darmstadt, 26–27 Mai 1995.

[BR95b] Philippe Brissi & Roger Rousseau. IREC: An Object-Oriented Abstract Representation to Handle Software Components in a Persistent Framework. In Vangalur S. Alagar and Rokia Missaoui (Eds.), 62nd Congress of l'Association Canadienne Française pour l'Avancement des Sciences (ACFAS), Montréal, 16–17 Mai 1994. Selected demonstration at European Conference on Object Oriented Programming (ECOOP'94), Bologna, 4-8 July 1994. *Object-Oriented Technology for Database and Software Systems*, World Scientific Publishing Corporation (Singapore), pages 79–94, Septembre 1995.

[CFLR93] Robert Chignoli, Jacques Farré, Philippe Lahire & Roger Rousseau Une implémentation de la persistance pour le langage Eiffel. In *Représentation par Objets (RPO'93)*, La Grande Motte, EC2, pages 131–142, 22–23 Juin 1993.

[CFLR93] Robert Chignoli, Jacques Farré, Philippe Lahire & Roger Rousseau. Floo: Basic principles and illustration of automatic persistency mechanism for eiffel. In *6th International Conference on Software Engineering and its Applications*, Paris, EC2, pages 181–190, 15–19 November 1993.

[CFLR95a] Robert Chignoli, Jacques Farré, Philippe Lahire & Roger Rousseau. FLOO: un environnement pour la programmation persistante en Eiffel. Technical Report 95-14, I3S – URA CNRS 1376 – Université de Nice – Sophia Antipolis, Mars 1995, révision Janvier 1996, 28 pages. Soumis à TSI pour le numéro spécial sur les objets (1996) (acceptation sous réserve de révisions mineures).

[CFLR95b] Robert Chignoli, Jacques Farré, Philippe Lahire & Roger Rousseau. Floo: A Strong Coupling between Eiffel Language and O2 DBMS. In Vangalur S. Alagar and Rokia Missaoui (Eds.) 62nd Congress of l'Association Canadienne Française pour l'Avancement des Sciences (ACFAS), Montréal, 16–17 Mai 1994. Selected demonstration at European Conference on Object Oriented Programming (ECOOP'94), Bologna, 4-8 July 1994. *Object-Oriented Technology for Database and Software Systems*, World Scientific Publishing Corporation (Singapore), pages 107–122, Septembre 1995.

[LJ95] Philippe Lahire & Jean-Marc Jugant. Lessons learned with Eiffel 3: the K2 Project. In Raimund Ege, Madhu Singh and Bertrand Meyer (Eds.), *International Conference on Technology of Object-Oriented Languages and Systems (TOOLS 17, USA'95)*, Santa Barbara (cal.), 31 Juillet – 4 Août 1995. Prentice-Hall, pages 207–215, Août 1995.

[RBC94a] Roger Rousseau, Philippe Brissi & Philippe Collet. From Eiffel to a General Object Model: Our Experience and Suggestions. In *9th Annual ACM Conf. on Object-Oriented Programming Systems, Languages and Applications (OOPSLA'94). Workshop on "The Object Engine: Foundation for Next Generation Architectures"*, Fred A. Cummins (Ed.), pages 29–37, 23–27 Octobre 1994.

[Rou94] Roger Rousseau. Fallait-il concevoir Ada 9x? Numéro spécial sur le langage Ada, *Technique et Science Informatique*, 13(5):723–729, Octobre 1994.

[Rou95] Roger Rousseau. Vers un modèle à objets commun pour le génie logiciel, les bases de données et la CAO. In *Journées du GDR Programmation du CNRS*, GDR de Programmation, pages 99–109, 22–24 Novembre 1995.

[Coi93] P. Cointe. CLOS and Smalltalk : a Comparison. Chapitre du livre *Object-Oriented Programming: The CLOS perspective, A. Pæpcke editor*, pages 215-250 (chapitre9), The MIT Press, 1993.

[MDC96] J. Malenfant, C. Dony & P. Cointe, A Semantic of Introspection in a Reflective Prototype-Based Language, Journal on Lisp And Symbolic Computation (LASC), à paraître en 1996.

[Riv96b] Fréderic Rivard, "Reflective Facilities in Smalltalk". Numéro spécial de Informatik/Informatique : Revue des organisations suisses d'informatique, Editeurs : SVI/FSI Fédération suisse des organisations d'informatique. Février 1996.

[Riv96c] Fréderic Rivard, "Smalltalk: a reflective language". Actes de la conférnrence Reflection'96 Editeurs : SVI/FSI Fédération suisse des organisations d'informatique. San Fransisco, 21-23 Avril 1996.

[ICCFM95] *IJCAI-95 Workshop "On Reflection and Meta Level Architecture and their Application in AI".* Organisateurs: M. Ibrahim, P. Cointe, F. Cummins, F. Giunchiglia & J. Malenfant, Montreal, Monday, August 21, 1995.

[LeCo96] Thomas Ledoux and Pierre Cointe, "Explicit Metaclasses As a Tool for Improving the Design of Class Libraries". *JSSST-JAIST, proceeding to appear in Springer-Verlag LNCS* ISOTAS'96, Kanazawa, Japan 11-15 March 1996.

[MMC95] P. Mulet, J. Malenfant et P. Cointe. Towards a Methodology for Explicit Composition of MetaObjects *OOPSLA'95* pages 316-330, proceedings edited by ACM Sigplan Notices Austin, Texas, October 1995.

[MLBRC94] P. Mulet, T. Ledoux, D. Barbaron, F. Rivard & P. Cointe, "Importing SOM Libraries into ClassTalk". *International Workshop on Multi-Languages Object Models & International Workshop Experiences with CORBA: is CORBA ready for duty?* OOPSLA'94, Portland, Oregon 22-23 October 1994.

[Coi94a] Pierre Cointe, "L'approche objet dans tous ses états". En collaboration avec le professeur Mokrane Bouzeghoub, MCSEAI'94, Rabat, eécole d'été, Avril 1994

[Coi94b] Pierre Cointe, "An Introduction to Object-Oriented Programming". Sixth European Summer School in Logic, Language and Information (ESSLI'94), Copenhague, Août 1994.

[Coi95a] Pierre Cointe, "Envy/Developer". ESUG (European Smalltalk Users' Group) Summer School. Proceedings édités par IBM, ESUG & Cap Volmac. Utrecht, August 28th - 1st 1995.

[Coi95b] Pierre Cointe, "Développer des applications Smallalk en équipes". Stratégies Objets 95, Paris, La Défense, 12 Octobre 1995

[Coi95c] Pierre Cointe, "Programmer en Smalltalk avec des métaclasses explicites", Stratégies Objets 95, Paris, La Défense, 12 Octobre 1995

[Coi96a] Pierre Cointe, "Smalltalk and the ENVY developmen System" TOOLS 96, Paris, Palais des Congrès, 26-29 Février 1996

[Coi95d] Pierre Cointe, Participation à la table ronde "La forte croissance de la productivité des études informatiques grâce à l'objet : réalité ou grande illusion ?", Forum Objet 95, Paris, La Défense, 24 Octobre 1995

[Riv95a] Fréderic Rivard, "Extension du compilateur Smalltalk : application à la paramétrisation de l'envoi de message". Journées Francophones des Langages Applicatifs (JFLA'95), Pages 115-134, Inria, Collection didactique #13, C. Queinnec, V. Viguie Donzeau-Gouge, P. Weis éditeurs Bois d'Amont (Jura), 30-31 Janvier 1995.

[Riv96a] Fréderic Rivard, "Pour un lien d'instanciation dynamique dans les langages à classes". Journées Francophones des Langages Applicatifs (JFLA'96) (à paraître), Val Morin - Quebec - 28-30 Janvier 1996.

[Bar94] Denis Barbaron, "Interface SOM pour Smalltalk et ClassTalk". Actes des journées du GDR de programmation. Lille, Octobre 1994.

[LeCo95] Thomas Ledoux et Pierre Cointe, "Les Métaclasses explicites comme outil pour améliorer la conception des bibliothèques de classes". *Actes des Journées du GDR de Programmation.* GDR'95, Grenoble, France, 23-24 Novembre 1995.

[Riv95b] Fred Rivard , "Pour un lien d'instanciation dynamique dans les langages à classes". *Actes des Journées du GDR de Programmation.* GDR'95, Grenoble, France, 23-24 Novembre 1995.

[Borne95] Isabelle Borne, "Environnement de programmation par objets: une approche pédagogique," habilitation à diriger des recherches de l'université Paris-5, à paraitre en rappord EMNantes. 18 Janvier 1995.

[Mul95] Philippe Mulet, "Réflexion & Langages à prototypes", Thèse de l'Université de Nantes, à paraitre en rappord EMNantes. 5 Juillet 1995.

[RC95] Fréderic Rivard, Pierre Cointe. *From Envy-Classtalk to Ada'9x* Final Report, 64 pages. Contrat Armines-OTI à paraître en rapport de recherche Ecole des Mines de Nantes

[Bour95] Mohammed Nouraddine Bouraquadi Saadani, "Métriques & Rétro-conception des bibliothèques de Smalltalk en ClassTalk". Rapport du DEA Informatique de Nantes 94-95, Juin 1995.

[Edg95] Nicolas A. Edgar, "A Survey of Distributed Object-Bases Programming Systems Supporting Code Mobility". Rapport de DEA, Université de Nantes, Septembre 1995.