Rapport d'Activité - GDR Programmation
Pôle Programmation par objets

Responsable: Christophe Dony [1]

Le pôle "Objets" du GDR Programmation est divisé en deux gros projets: ELO (Evolution des Langages à Objets) et INTERLOO (Interopérabilité des Langages à Objets).

Nés avec les années 70, expérimentés intensément dans les années 80, les langages à objets sont sortis des laboratoires et devraient atteindre leur pleine maturité d'ici la fin de ce siècle. En particulier, ils ont déjà démontré combien ils pouvaient apporter en ce qui concerner l'amélioration de la qualité du logiciel et du processus de production.

Aujourd'hui, la sémantique et les techniques d'implémentation de langages comme Smalltalk, CLOS, Objective-C ou Eiffel sont stabilisées - certains de ces langages sont même en voie de normalisation - et permettent leur mise en œuvre quasi industrielle.

Cependant le succès des langages à objets a contraint les concepteurs à en figer trop tôt certains aspects. Certains résultats de recherches n'ont pas (encore) été intégrés dans les langages les plus utilisés industriellement, certains sujets ont été peu ou incomplètement traités, de nouveaux problèmes apparaissent nés de nouvelles applications. Le projet ELO regroupe des équipes travaillant sur diverses évolutions des modèles et des langages à objets.

Par ailleurs - en l'absence d'un modèle objet universel - les modèles et les langages à objets proposés sont souvent très différents. A tel point, qu'une application développée dans l'un de ces langages est encore très difficilement portable dans un autre langage. Ainsi, il apparaît qu'il est beaucoup plus facile de traduire une grosse application de Pascal en ADA, en C ou en Fortran que de traduire une grosse application de Smalltalk en Eiffel par exemple. L'objectif du projet INTERLOO sont de synthétiser les différentes méthodes de conception et d'étudier puis de proposer des mécanismes d'inter-opérabilité permettant de créer de bibliothèques de composants logiciels réellement partageables et interchangeables.

Le pôle regroupe plus de quarante chercheurs, il est représenté dans les comités de programme des conférences internationales et nationales sur la programmation par objets (ECOOP, OOPSLA, TOOLS, LMO, RPO). Ce rapport présente les résultats recherche du pôle ainsi que les applications industrielles de ces travaux. Parmi celles-ci, le système *Resyn* a obtenu en 1994 le prix ADER-Banques Populaires *Entreprises-Innovation* qui récompense chaque année un travail de recherche alliant un laboratoire et une entreprise.

1. Michel Habib a été responsable du pôle jusqu'à octobre 1994, il a démissionné. Christophe Dony a été choisi, lors de la réunion du pôle à Grenoble, pour le remplacer

A - Projet ELO: Evolution des langages à objets

Resonsables: Christophe Dony, Michel Habib

Objectifs:

Ce projet a pour thème l'évolution de la programmation par objets (langages modèles et techniques).

Les objectifs scientifiques du projet sont l'explorations de certains aspects mal définis ou nouveaux des langages et modèles à objets.

- La modélisation de l'héritage (multiplicité, exceptions, codage, réorganisation, classification), section 1.
- La sémantique de la programmation sans classes, son intérêt dans l'étude de la reflexivité et ses connections avec la représentation de connaissances, section 2.
- La représentation de connaissances complexes et les outils qui lui sont nécessaires dans les langages à objets: représentation de points de vues, de versions, d'objets composites, de contraintes. Section 3
- La spécification formelles de programmes à objets, section 4.

Le projet à également des objectifs de collaboration nouvelle entre les équipes. Nos sujets de recherche sont tous assez directement connectés (prenons par exemple le codage et la réorganisation de hiérarchies et la modélisation de l'héritage multiple et de l'héritage avec exceptions ou la programmation par prototypes et les points de vues. Le projet a permis à des équipes qui n'avaient pas toujours l'habitude de collaborer de confronter leurs points de vues et de mener à bien des recherches communes.

Sites et Participants:

- CNET - Centre National d'Etude des Télécommunications (Issy-les-Moulineaux):
  M. Dao (Ingénieur), Milene Shteto (doctorante).
- EMN - Ecole des Mines de Nantes:
  P. Cointe (professeur) et P. Mulet (doctorant EMN).
- IRIN (institut de recherche en informatique de Nantes), (ERTO) Equipe de Recherche en Technologies Objet:
  J.C. Royer (mdc).
- LIFL - Laboratoire d'Informatique Fondamentale de Lille:
  B. Carré (mdc), L. Dekker, Gilles Vanwormhoudt (doctorant).
- LIRMM - Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier:
  C. Dony (mdc), R. Ducournau (Professeur) M. Habib (professeur), M. Huchard (mdc), T.Libourel (mdc), M.L. Mugnier (mdc), J. Quinqueton (DR INRIA), Daniel Bardou (doctorant), Stéphane Pelenc (Doctorant).
- LGI2P [2] (Nîmes, Ales):
  M. Oussalah (professeur), M. Magnan (mdc),
- CRIN - Centre de Recherche en Informatique de Nancy: A. Napoli (CR CNRS)

2. Le LGI2P est un laboratoire commun de l'EERIE (Ecole pour les Etudes et la Recherche en Informatique et Electronique de Nîmes) et de l'EMA (Ecole de Mines d'Ales)

1 Sémantique et algorithmique de l'héritage

1.1 Héritage multiple

PArticipants: Michel Habib (LIRMM), Roland Ducournau (LIRMM), Marianne Huchard (LIRMM), Marie-Laure Mugnier (LIRMM), Amédéo Napoli (CRIN-CNRS).

Les travaux entrepris depuis 1987 par M. Habib (LIRMM) et R. Ducournau (Sema Group, puis LIRMM) ont abouti :

- à une nouvelle modélisation de l'héritage, avec la distinction entre *héritage de nom* et *héritage de valeur* [2, 4] qui permet de distinguer les conflits syntaxiques de nom des conflits "sémantiques" de valeur.
- à la construction d'une linéarisation *monotone* [3], dont la propriété de monotonie, annoncée dans des travaux précédents [2, 2], assure que la linéarisation d'une sous-classe "contient" la linéarisation de ses super-classes, ce qui est particulièrement utile pour la combinaison des méthodes. Cette nouvelle linéarisation étend les linéarisations précédemment étudiées de LOOPS et de CLOS dans la mesure où elle leur est identique quand ces deux linéarisations le sont.

L'ensemble de ces travaux fait l'objet d'un article de synthèse à paraître dans TSI [4].
Les techniques développées ne sont pas sans application : les bonnes propriétés de cette nouvelle linéarisation en font un bon candidat pour remplacer celle de CLOS et l'une des perspectives de notre travail consiste à étudier des applications de CLOS existantes, pour voir l'impact qu'aurait l'adoption par CLOS de cette nouvelle linéarisation. De plus, la technique d'héritage cumulatif non-monotone décrite dans [4] est effectivement utilisée dans le projet *RESYN* (cf. section 6.1).

Références

[1] R. Ducournau, M. Habib, M. Huchard, M-L. Mugnier, and A. Napoli. L'héritage multiple dans tous ses états. Rapport de Recherche 92-021, L.I.R.M.M., Montpellier, Juillet 1992.

[2] R. Ducournau, M. Habib, M. Huchard, and M-L. Mugnier. Monotonic conflict resolution mechanisms for inheritance. In A. Paepcke, editor, *Proceedings of the Seventh ACM Conference on Object-Oriented Programming, Languages and Applications, OOPSLA'92*, pages 16–24, Vancouver, 1992. ACM Press.

[3] R. Ducournau, M. Habib, M. Huchard, and M-L. Mugnier. Proposal for a monotonic multiple inheritance linearization. In *Proceedings of the Ninth ACM Conference on Object-Oriented Programming, Languages and Applications, OOPSLA'94*. ACM Press, 1994.

[4] R. Ducournau, M. Habib, M. Huchard, M-L. Mugnier, and A. Napoli. Le point sur l'héritage multiple. *Technique et Science Informatiques*, 1995. (à paraître).

1.2 Classification

Participants: B.Carré (LIFL), L.Dekke (LIFL), R.Ducournau (LIRMM), A.Napoli (CRIN-CNRS), J.Quinqueton (LIRMM).

Les travaux effectués dans le cadre du thème "Classification et objets" essaient de mettre en valeur le rôle de la classification dans le contexte d'un environnement à base d'objets, en tant que processus d'aide à la conception de hiérarchies et en tant que processus de raisonnement.

Plusieurs directions de recherches sont ouvertes, liées en particulier aux études sur les logiques terminologiques et sur les systèmes de représentation de connaissances à base d'objets. L'atelier de travail *Object-based Representation Systems* tenu pendant la conférence mondiale IJCAI'93 (Chambéry, août-septembre 1993) a montré l'importance de la classification dans les systèmes de représentation [1]. Toutefois, le problème d'ajouter des composants procéduraux dans des systèmes comme les logiques terminologiques reste encore entier [2].

Par ailleurs, le raisonnement par classification peut être employé comme une aide à la recherche d'information dans le cadre de données hiérarchisées, que ce soit dans le contexte de la gestion de bases de données ou dans celui de la recherche d'analogues dans le contexte du raisonnement à partir de cas [3]. La modélisation de la synthèse en chimie organique est un des domaines d'application privilégié pour les techniques décrites ci-dessus (cf. section 6.1).

De nombreux points restent ouverts dans la recherche sur les systèmes de représentation à base d'objets, et notamment : la programmation doit-elle être autorisée dans le contexte de tels systèmes, et si oui, sous quelle forme et avec quelle sémantique, comment représenter sous forme "déclarativé" des composants procéduraux ou fonctionnels, qui deviennent alors des objets qui peuvent être traités par le processus de classification. Ces discussions sont abordées en profondeur dans le groupe *classification et objets* du PRC-IA dont certains des membres appartiennent aussi au *pôle objet* du GDR de programmation [4].

Références

[1] L. Dekker and A. Napoli. Report on the IJCAI-93 Workshop on "Object-Based Representation Systems". *SIGART Bulletin*, 5(1):57–59, 1994.

[2] A. Napoli. Studies about the Integration of Classification-Based Reasoning and Object-Oriented Programming. In F. Baader, M. Lenzerini, W. Nutt, and P.F. Patel-Schneider, editors, *Working Notes of the 1994 Description Logics Workshop*, pages 60–62. DFKI Saarbruecken, 1994.

[3] A. Napoli. Catégorisation, raisonnement par classification et raisonnement à partir de cas. In *Journées Acquisition – Validation – Apprentissage (JAVA'94), Strasbourg, 21–25 mars*, pages E1–E14, 1994.

[4] B. Carré, R. Ducournau, J. Euzenat, A. Napoli, and F. Rechenmann. Classification et objets : programmation ou représentation ? In *Journées nationales du PRC-IA, Nancy*. 1995. (à paraître).

1.3 Cache des méthodes

Participants: P.André (ERTO), J.C.Royer(ERTO), R.Ducounau(LIRMM).

Le point suivant dans les problèmes liés à l'héritage est l'efficacité de l'invocation des méthodes et plus particulièrement de la recherche de la méthode à invoquer suite à un envoi de message. Ce problème est particulièrement important dans le cas des langages typés dynamiquement. Les techniques de cache consistent à effectuer un codage de la hiérarchie d'héritage.

1. Equipe de J.C. Royer
   Un premier travail d'implantation d'un système d'analyse statistique (EROS) a été fait par P. André. Ce système a permis de faire des mesures sur Smalltalk et d'étudier l'utilisation d'une technique de coloriage de graphe. Après une étude plus théorique de cette technique dans le cadre des langages dynamiques, un algorithme de coloriage incrémental est proposé et appliqué au langage Smalltalk [1]. Ce travail a donné lieu à des comparaisons avec le système Yafool de Roland Ducournau. Nous avons échangé des données communes pour comparer plus finement les résultats : les techniques mises au point dans Yafool présentent une très nette amélioration.

2. Technique de YAFOOL : R. Ducournau.
   A la suite de la parution de [1], la technique utilisée en YAFOOL pour le calcul de la structure des objets a été appliquée au calcul du cache des méthodes en SMALLTALK, avec des résultats très satisfaisants [2]. Une réflexion est en cours (stage de DEA) pour essayer de comprendre la qualité de ces résultats, de prouver leur optimalité et de les améliorer, en particulier compte-tenu de travaux récents (J. Vitek à ECOOP'94).

Références

[1] P. André and J.-C. Royer. Optimizing method search with lookup caches and incremental coloring. In A. Paepcke, editor, *Proceedings of the Seventh ACM Conference on Object-Oriented Programming, Languages and Applications, OOPSLA'92*, pages 110–126, Vancouver, 1992. ACM Press.

[2] R. Ducournau, M. Habib, M. Huchard, M-L. Mugnier, and A. Napoli. L'héritage multiple dans tous ses états. Rapport de Recherche 92-021, L.I.R.M.M., Montpellier, Juillet 1992. Citén section 1.1

1.4 Représentation efficace de hiérarchies

Participants: Michel Habib, Marianne Huchard et Lhouari Nourine (LIRMM).

Dans de nombreuses applications, il est important de pouvoir stocker efficacement les hiérarchies d'héritage qui sont des graphes sans circuit ou des treillis. Ce problème est un cas particulier de celui du codage d'un ordre.

Un codage d'un ordre est une application f d'un ordre $P = (X, \leq)$ dans un ordre $Q = (Y, \leq)$ telle que $f : X \rightarrow Y$ et vérifie: $x \leq_P y$ si et seulement si $f(x) \leq_Q f(y)$. Un tel codage est intéressant lorsque la relation d'ordre sur Q est plus facile à manipuler que celle sur P, où lorsque la place mémoire utilisée est réduite.

Les codages les plus fréquents sont ceux où Q est un treillis booléen, on obtient alors des codages par vecteurs de bits. Les problèmes d'optimisation sous-jacents sont NP-difficiles, mais aucun argument théorique n'interdit de construire des codages optimaux sur des classes particulières d'ordres ou de concevoir des heuristiques très efficaces pour le cas général. Les principales heuristiques actuelles ont été proposées par Aït-Kaci, Caseau, Elis et Habib-Nourine. L'importance pratique du problème est telle qu'il y aura un colloque en 95, KRUSE, dédié à ces questions.

Les premiers résultats [1] concernent les bornes théoriques sur la taille en nombre de bits d'un codage, la NP-complétude du problème, l'égalité de la taille du codage d'un ordre et de son complété, un codage optimal pour les treillis distributifs et certains treillis décomposables, ainsi qu'une heuristique "paresseuse" pour les treillis quelconques.

Références

[1] M. Habib, L. Nourine: Bit-vector Encoding for Partially Orderes Sets, Proc. ORDAL, Lyon 94, Lecture Notes in Computer Science, 831, p. 1-13, 1994.

1.5 Réorganisation de hiérarchie

Participants: H. Dicky, C. Dony, M. Huchard, T. Libourel.

Les hiérarchies d'héritage sont le plus souvent construites "à la main" par un concepteur ou un groupe de concepteurs selon des règles empiriques qui ne sont pas toujours généralisables.

Lorsque la complexité et la quantité des objets à structurer devient trop importante, ou lorsque la hiérarchie a beaucoup évolué depuis sa conception initiale, il devient nécessaire d'automatiser au moins en partie le processus de (re)structuration. Il paraît difficile, en raison du nombre de critères à prendre en compte pour obtenir une "bonne" hiérarchie et du caractère discutable de ces critères, de prétendre produire des méthodes infaillibles, l'enjeu consiste donc à proposer de bons outils d'aide à la conception.

On peut réorganiser localement une hiérarchie avec des règles de modification inspirées du génie logiciel, ou prendre un point de vue plus général, et s'appuyer sur le treillis de Galois de la relation liant classes et attributs. Nous avons ainsi proposé [1, 2] un algorithme permettant de construire de fac con incrémentale un ordre partiel bien défini par rapport au treillis de Galois, la "sous-hiérarchie de Galois". Un mémoire de DEA [3] a par ailleurs été consacré à l'étude de système similaires comme par exemple le système DEMETER et à l'implantation de l'algorithme.

Références

[1] H. Dicky, C. Dony, M. Huchard, T. Libourel: "ARES, un algorithme d'Ajout avec REStructuration dans les hierarchies de classes". Conférence LMO'94 (Langages et Modeles a objets), Grenoble, octobre 1994.

[2] H. Dicky, C. Dony, M. Huchard, T. Libourel: "ARES, un algorithme d'Ajout avec REStructuration dans les hierarchies de classes" Rapport de Recherche LIRMM No 94039, mai 1994. Contient les preuves et la complexité de l'algorithme.

[3] Florent Pages: Etude de Methodes de Reorganisation des Hiérarchies de Classes. Rapport de DEA, LIRMM, Septembre 1994.

2 Langages à prototypes et réflexion

Participants: Pierre Cointe (EMN), Christophe Dony, Jacques Malenfant (Univ. Montréal), Daniel Bardou (LIRMM), Philippe Mulet (EMN), Marco Jacques (Univ. Montréal).

Les systèmes à prototypes sont utilisés aussi bien en représentation des connaissances, dans les langages de Frame, qu'en programmation dans les langages à prototypes apparus plus récemment. Ce projet initié dans l'équipe RXF-LITP[3] avait pour but premier de mieux comprendre les langages à prototypes comme Self, Garnet, Actra, KEVO et les comparer aux langages de frames (Y3). Nous avons ainsi proposé une première taxinomie de ces langages (OOPSLA'92). Cette étude est maintenant menée en collaboration entre l'EMN, le LIRMM et l'équipe de Jacques Malenfant à l'Université de Montréal.

Le travail sur la taxonomie se poursuit et porte sur les différentes sémantiques du lien de délégation, sur les problèmes de partage d'objets et sur les applications à la gestion des points de vues, il fait l'objet du travail de thèse de Daniel Bardou.

Un second thème d'intérêt est l'étude d'un modèle de prototypes réflexif dans lequel a été mise en œuvre une réflexion de comportement permettant de paramétrer au niveau de chaque objet la sémantique de l'envoi de messages. Ce modèle a d'abord été expérimenté en grandeur nature sur une extension réflexive du langage Self [1], dont un interprète avait été réalisé au préalable, en Smalltalk, conjointement par P. Mulet et F. Rivard.
Ces travaux prennent aujourd'hui les directions suivantes :

- Formalisation d'une sémantique de ce modèle à partir d'un système de ré-écriture [2, 6].

  Cette formalisation démontre "la convergence" du processus de résolution de message et ce malgré la régression du lien méta.

- Implémentation en Scheme du langage Moostrap, dérivé de ce premier modèle [4, 3].

  Moostrap est construit autour d'un noyau réduit de primitives, et son caractère réflexif lui permet d'être profondément étendu sans en modifier la machine virtuelle. Moostrap doit être vu comme une plate-forme d'étude des langages à prototypes permettant de définir de nouvelles sémantiques au travers d'une bibliothèque de méta-objets.

  Nous étudions, en outre, une méthodologie de composition de méta-objets qui permet notamment de composer des méta-objets élémentaires de manière incrémentale en conservant la monotonie du système [7]. Certains résultats de cette étude s'appliqueront également à la composition des métaclasses.

  Parallèlement, nous avons établi une analogie entre les concepts d'environnements Scheme et de prototypes Self. Cette analogie nous a conduit à proposer un outil pédagogique de simulation de prototypes en Scheme [5].

Références

[1] Pierre Cointe, Jacques Malenfant, Christophe Dony et Philippe Mulet. Etude de la réflexion de comportement dans le langage Self. Dans les *Actes des Premières journées Représentation par Objets*, pages 213–224, La Grande Motte, France, Juin 1992. EC2.

[2] Jacques Malenfant, Pierre Cointe et Christophe Dony, "Behavioral Reflection in a Protoype-Based Languagé'. *Int'l Workshop on reflection and Meta-Level Architectures.* ISE and IPA (Japan) + ACM SIGPLAN, JSSST & IPS, pp 143-153. Edité par A. Yonezawa et B. Smith Tokyo, Japan, November 92.

[3] Philippe Mulet et Pierre Cointe. Définition d'un noyau réflexif pour un langage à prototypes. Dans les *Actes des Secondes Journées Représentation par Objets*, pages 101–116, La Grande Motte, France, Juin 1993. EC2-IA2.

3. à laquelle appartenaient entre autres P. Cointe, C. Dony, J. Malenfant et P. Mulet.

[4] Philippe Mulet et Pierre Cointe. Definition of a Reflective Kernel for a Prototype-Based Language. Edité par Shojiro Nishio et Akinori Yonezawa, *Lecture Notes in Computer Science vol.742: First International Symposium on Object Technologies for Advanced Software*, pages 128–144, Kanazawa, Japan, Novembre 1993. JSSST-JAIST, Springer-Verlag.

[5] Philippe Mulet and Marco Jacques. De la parenté entre les environnements de MIT Scheme et les prototypes de Self. Dans les *Actes de Langages et Modèles à Objets'94*, pages 167–178, Grenoble, France, Octobre 1994.

[6] Jacques Malenfant, Christophe Dony et Pierre Cointe, A Semantic of Introspection in a Reflective Prototype-Based Language, Journal on Lisp And Functional Programming (LASC), à paraitre en 1995.

[7] Philippe Mulet et Jacques Malenfant. Towards a Methodology for Composing MetaObjects. *Soumis à ECOOP'95*, Aarhus, Danemark, Août 1995.

3 Supports à la représentation de connaissances complexes et évolutives

Les recherches regroupées sous ce thème traitent de l'implantation au sein des langages à objets de traits permettant la représentation de connaissances complexes et évolutives (objets avec point de vues, objets composites, objets contraints) ainsi que de la façon dont ces connaissances peuvent décrites et manipulés. Il est bon de noter qu'il s'agit dans tous les cas de problématiques générales posées au paradigme objet, sans distinction de techniques particulières : programmation par objets, représentation des connaissances, bases de données a objets.

3.1 Points de Vues

Participants: Bernard Carré (LIFL), Lenneke Dekker (LIFL), Gilles Vanwormhoudt (LIFL). Suite aux réunions de travail du GDR, une collaboration a débuté avec Christophe Dony (LIRMM) et Daniel Bardou (LIRMM) pour étudier la possibilité d'utiliser les prototypes pour représenter les points de vues.

L'approche par points de vue permet d'associer plusieurs représentations (intensions) à une même entité (le référent, d'extension unique dans le monde réel). De tels besoins se révèlent dans de nombreux problèmes abordés, parfois reformulés, au travers du paradigme objet.

Il existe essentiellement deux approches du problème : l'une reposant sur la composition d'objets (agrégation de perspectives), l'autre considérant qu'il s'agit à la base d'un probleme d'héritage multiple. Le groupe s'est intéressé plus particulièrement à l'étude de cette dernière approche autour du langage ROME.

Dans ce langage, la classe est retenue comme seul moyen de décrire des objets. La représentation multiple (parfois appelée improprement "instanciation multiple") permet à un objet, instance d'une seule classe initiale, d'être membre de plusieurs classes et, par héritage, des surclasses de celles-ci. L'ensemble de ces classes détermine le graphe de représentation de l'objet. Toute classe dtermine alors un point de vue sur ses objets. L'opération de sélection de point de vue effectue une projection du graphe de représentation de l'objet et, partant, de sa description.

Le langage de représentation des connaissances à base de frames FROME [1, 2], obtenu par méta-extension de ROME, "hérite" des capacités précédentes qu'il adapte et enrichit pour les structures d'objets plus complexes que sont les frames. Ceci autorise en particulier l'homonymie d'attributs et sa prise en charge dans le processus de classification d'objets. Une étude a été menée par ailleurs sur les possibilites d'implémenter ces fonctionnalités dans le langage Smalltalk, c'est le prototype RomeTalk, résultat d'un travail conjoint du LIRMM et du LIFL [5].

Enfin un projet plus récent tente à systématiser l'approche par point de vue à travers la notion de plan de description. Il s'agit du modèle CROME (Contextes en ROME) de conception par objets d'architectures multi-applications [4, 3]: Un "plan de base" décrit, par une hiérarchie de classes, les objets du domaine et détermine notamment leur identité, des "plans fonctionnels" détiennent modulairement les adaptations à apposer sur le plan de base, relativement aux contextes applicatifs. Un prototype a été développé en exploitant les capacités de représentation multiple par points de vue de ROME ainsi que ses aspects réflexifs. Il a été expérimenté dans le domaine de la CAO électronique.

Références

[1] Lenneke Dekker : La reification des filtres en FROME, Conference sur les Representations par Objets, RPO'93, EC2 Ed, La Grande Motte, 17-18 juin 1993.

[2] Lenneke Dekker : FROME : Representation multiple et classification d'objets avec points de vue, These de l'Universite des Sciences et Technologies de Lille. 13 Juin 1994.

[3] Bernard Carré et Lenneke Dekker: CSCW et partage d'objets. Actes des journees Ganymede sur le travail cooperatif et la communication avancee, Lille, Juin 1994. Publication interne LIFL no 159.

[4] Gilles Vanwormhoudt et Philippe Rio CROME, Contextes en ROME. Mémoire de DEA du LIFL. Juin 1993.

[5] Cyril Bourdin. Points de vue et Représentation Multiple et Evolutive dans les Langages à Objets. Mémoire de DEA du LIRMM. Juin 1994.

3.2 Objets Composites, Evolution d'objets

Participants: M.Oussalah, M.Magnan, G.Talens, L.Torres (LGI2P).

L'équipe de Mourad Oussalah (LGI2P) s'intéresse aux différentes techniques d'évolution d'objets simples et composites [5]. Sont présentés ici les résultats concernant d'une part la réutilisation de classes via les objets composites et d'autre part les versions d'objets.

Réutilisation de classes via les objets composites.

De nombreuses applications utilisent la relation de généralisation/spécialisation. Certains systèmes à objets offrent en plus la relation de composition qui permet l'utilisation d'objets à structures complexes appelés objets composites.

- Modèle d'objets composites:

  Les systèmes à objets permettant la description d'objets composites se répartissent en deux groupes: les systèmes ayant une relation de composition fixe et ceux, sur lesquels nous travaillons, qui permettent au concepteur de décrire la nature de la relation de composition. Dans le modèle réalisé[2, 9], les liens de composition entre un objet composite et ses composants sont explicitement nommés et typés, leur cardinalité peut être spécifiée, ils peuvent être exclusifs ou partagés, dépendants ou indépendants, prédominants ou non.

- Exceptions dans un graphe de composition.

  Les exceptions dans les graphes de composition ont également été introduites[3] car elles favorisent la réutilisabilité et la conception incrémentale. Nous distinguons trois types d'exceptions: les exceptions à l'héritage (horizontal) d'une propriété que nous appelons exceptions classe-propriété [7] et les exceptions par annulation d'un lien de la relation de composition que nous appelons exceptions classe-classe ou exceptions instance-classe [2].

Versions d'objets (simples et composites)

L'objectif est d'étudier et de développer un modèle de représentation et de gestion de versions d'objets simples et composites permettant aux utilisateurs de modéliser et gérer l'évolution complète de leur conception.

Dans ce but ont été réalisés:
- une étude des différentes solutions connues en matière de versions et une synthèse traitant uniquement le cas des objets simples et composites[1],
- une taxonomie de versions [8],
- une représentation explicite des différents types de liens de dérivation [6]
- une représentation explicite de liens d'équivalence et de liens sensitifs pour éviter la prolifération de versions, (5) un modèle de propagation de versions d'objets simples et composites [4].

Applications industrielles:

Ces travaux sont utilisés dans le cadre de diverses applications en collaboration avec des industriels:
- CNET(Paris)-ITESOFT(Nîmes), Presage : Un environnement orienté objet pour la modélisation de réseaux de télécommunication, cf. section 6.3.
- CNET(Paris), Filtrage et Classification de composants matériels et logiciels.
- Matra Marconi Space (Toulouse), MOSAIC : Modélisation objet pour la conception de centre de contrôle de satellites.

Références

[1] G. Talens, Gestion de versions d'objets simples et composites (Thèse soutenue à l'Université de Montpellier II, Fevrier 94) .

[2] M. Magnan, Réutilisation de composants : un modèle d'objets composites avec exceptions, (Thèse soutenue à l'Université de Montpellier II, Septembre 94) .

[3] M. Magnan, C. Oussalah., Multiple inheritance systems with exceptions, in Artificial Intelligence Review Vol. 6 , Kluwer Acad.Pub.(1994) pp. 3-24.

[4] G Talens, C. Oussalah, Versions of Simple and Composite Objects. in Proc. of the 19th Intl. Conf. on VLDB, Dublin, Ireland, Aug. (1993), pp. 62-72.

[5] M. Magnan, C. Oussalah. Object Evolution. in Proc. of the 5th IEEE Intl Conf. on Software Engineering and Knowledge Engineering, San Francisco, June (1993), pp. 546-554. (nominated as one of the best five papers of SEKé93).

[6] C. Oussalah, G Talens , Concepts and Methods for version Modeling. in Proc. of the 3rd IEEE/ACM EuroDAC , Hambourg, Germany, Sep.. (1993), pp. 332-338.

[7] C. Oussalah, M. Magnan, L. Torres. A method for the management of exceptions in multiple inheritance systems, in Proc. of the 4th IEEE International Conference on TOOLS with Artificial Intelligence, Arlington, Virginia, USA, November (1992), pp. 222-229.

[8] C. Oussalah, G Talens. Version management for the modelling of complex systems. In Proc. of the E/C Information Technology, Tokyo, Japan, september (1992), pp. 146-152. p

[9] M.Magnan, S. Benmahmoud, C. Oussalah. Classification d'objets composites. In Proc. of the RFIA'94,Palais de Congrès, Paris (1994), pp. 240-246.

[10] X.Vincent, Implantation d'un système de stockage de versions d'objets, rapport de DEA soutenu en septembre 1993 à l'USTL Montpellier-II.

[11] K. Abderrezak, Mécanismes de partage des propriétés dans les graphes de composition, rapport de DEA soutenu en juin 1994 à l'USTL-Montpellier.

3.3 Contraintes et Objets

3.3.1 Objets et CSP

Participants: R.Ducournau, M.Habib, T.Kokeny, M.C.Vilarem (LIRMM).

Ce travail traite de l'introduction de la définition et de la résolution des Problèmes à Satisfaction des Contraintes (CSP) dans un environnement Orienté Objets. Il a fait l'objet de la thèse de Tibor Kokeny.

- En utilisant les structures et les mécanismes offerts par un langage à objets, une représentation modulaire et extensible des CSP ont été proposée. Cette représentation comprend non seulement la définition statique des éléments des CSP par les objets (classes de variables et de contraintes), mais également les outils algorithmiques de résolution de CSP. [6, 1].

- Pour les CSP dans lesquelles les domaines sont des ensembles d'objets, un algorithme de filtrage de consistance d'arc et un algorithme calculant des ensembles de solutions ont été proposés en prenant en compte la hiérarchie de spécialisation/généralisation des objets [2, 3, 4, 5].

– Pour pouvoir permettre la définition des contraintes sur les chemins d'attributs, une nouvelle architecture de communication entre variables contraintes et objets a été proposée, basée sur les variables flottantes et sur les contraintes d'interface.

Le système **CSPOO**[6, 7] est la synthèse de ces études, il offre des outils pour définir et résoudre des problèmes de contraintes dans l'environnement orienté objets Y3.

3.3.2 Objets et contraintes directionnelles

Participants: R.Ducournau (LIRMM), Michel Dao (CNET), Stephane Pelenc (LIRMM), Milene Shteto (CNET)

Développée dans le cadre du projet RODIN du CNET (Paris), l'application CONRAD étudie l'utilisation des techniques informatiques à base d'objets pour modéliser et déterminer la topologie et le dimensionnement des réseaux téléphoniques à commutation de circuits avec acheminement adaptatif. Pour plus de détails voir la section 6.2.

Références

[1] T. Kökény. CSPOO : un système de résolution de contraintes orientées objet. In M. Habib and M. Oussalah, editors, *Actes des journées Représentation Par Objets, RPO'93*, La Grande Motte, juin 1993. EC2.

[2] Tibor Kökény. A new arc-consistency algorithm for CSPs with hierarchical domains. In *Proceedings of IEEE Conference of Tools for Artificial Intelligence, ICTAI'94*, pages 439–445, New Orleans, USA, 1994.

[3] Tibor Kökény. Yet another object-oriented constraint resolution system: an open architecture approach. In *Proceedings of Technology of Object-Oriented Languages and Systems Conference USA, TOOLS USA'94*, pages 103–114, Santa Barbara, USA, 1994.

[4] A. Bellicha, C. Capelle, M. Habib, T. Kökény, and M-C. Vilarem. CSP techniques using partial orders on domain values. In *Proceedings of ECAI'94 Workshop on Constraint Satisfaction Issues Raised by Practical Applications*, Amsterdam, 1994.

[5] Tibor Kökény. HAC-6: an arc-consistency algorithm for hierarchical CSPs. Research Report 94-040, LIRMM, 1994.

[6] T. Kökény. *Satisfaction de contraintes dans un environnement orienté objets.* Thèse d'informatique, Université Montpellier II, déc. 1994.

[7] Tibor Kökény. Constraints and objects: a new way of integration. Research Report 94-041, LIRMM, 1994.

4 Spécifications formelles

Participants:n J.C Royer(IRIN), P.André (ERTO)

L'intégration de spécifications formelles dans le cadre de la programmation par objets est un enjeu important en génie logiciel. En effet ces préoccupations sont à la base d'une bonne inter-opérabilité et de la spécification de composants réutilisables. Un premier axe d'étude concerne la conception et un deuxième l'analyse.

Au niveau de la conception un premier travail établit des liens précis entre la notion de classe et les types abstraits algébriques. Ceci a permis de définir le modèle des classes formelles [2], c'est un modèle algébrique avec une sémantique opérationnelle, des règles de bonne conception et des outils divers. Ce modèle a servi à expérimenter quelques cas simples d'étude [6, 4, 1]. L'étude du typage a été abordé et un système avec règles covariantes a été prouvé. Ces travaux ont été fait en collaboration avec l'équipe de Dan Chiorean du laboratoire d'informatique de l'Université de CLUJ (Roumanie) et de Corina Cirstea actuellement à Computing Laboratory, Oxford (England).

L'axe analyse à objets et spécifications formelles est la partie principale de la thèse de Pascal André. Un formalise plus abstrait que les classes formelles a été élaboré : les Types Abstraits Graphiques. Ce formalisme est destiné à permettre une meilleure analyse à objets en associant une description fonctionnelle et une description dynamique d'un système. Dans le cadre des systèmes séquentiels nous utilisons un automate avec gardes pour décrire le comportement dynamique. Cet automate sert ensuite à produire la description fonctionnelle qui est une spécification algébrique dans le style d'Asspegique. L'intérêt est une facilité d'écriture indéniable et méthodique et des possibilités d'outils performants notamment pour la conception en classes formelles [3, 5].

Références

[1] Pascal André, Dan Chiorean, Corina Cîrstea, and Jean-Claude Royer. Object-Oriented Design With Formal Classes. In *ConTI'94: International Conference on Technical Informatics*, 1994. 16-19 November, Timisoara, Romania.

[2] Pascal André, Dan Chiorean, and Jean-Claude Royer. The Formal Class Model. In *Joint Modular Languages Conference, Modula, Oberon & friends*, Ulm, Germany, 1994. GI, SIG and BCS. September, 28-30.

[3] Pascal André and Jean-Claude Royer. Introduction de concepts formels dans le développement objet. Journées du GDR de Programmation, Lille, September 1994.

[4] Pascal André and Jean-Claude Royer. La modélisation des listes en programmation par objets. In Pierre Cointe, Christian Quiennec, and Bernard Serpette, editors, *Journées Francophones des Langages Applicatifs (JFLA'94)*, number 11 in Collection Didactique, pages 259–285, Noirmoutier, 31 janvier - 1 février 1994. INRIA.

[5] Pascal André and Jean-Claude Royer. Formal Concepts and Tools for Object-Oriented Analysis and Design. In *5th ICC: Object-Oriented Technology: Methods and Applications*. Tata McGraw-Hill, January 1995. 9-11 January, Hyderabad, India.

[6] Jean-Claude Royer. Un exercice de spécification formelle de preuve et de conception à objets. Rapport de recherche 93-39, IRIN, Faculté des Sciences et des Techniques, Université de Nantes, September 1993.

5 Manifestations

Les conférences RPO (Représentation par objets - Le point sur la recherche et ses applications) 1992 et 1993 à la Grande-Motte ont été organisés par la société EC2 et IA2 en collaboration avec les membres du pôle objet et du groupe ELO.
Le groupe a participé à l'organisation La conférence LMO (Langages et Modèles à Objets) tenue à Grenoble en Octobre 1994. La prochaine est prévue a Nancy en 1995.

6 Applications Industrielles

Outres les applications déjà citées, certaine des recherches que nous venons de décrire sont utilisées dans des applications d'envergure: l'héritage non cumulatif non-monotone (cf. section 1.1et les techniques de classification (cf. section 1.2)dans le projet *RESYN*, les implantations des techniques de représentation de connaissances (cf. section 3) et les contraintes à objets dans les projet *RODIN* et *PRESAGE*.

6.1 RESYN: Applications des objets en Chimie Organique

Cette application résulte de la coopération entre un industriel de la chimie (Roussel Uclaf), un industriel de l'intelligence artificielle (Framentec) et de groupes de recherche publics (LIRMM et CCIPE). Ces travaux se font dans le cadre du GDR 1093 du CNRS-Chimie et du groupe ELO du GDR Programmation.
Le système *Resyn* a obtenu en 1994 le prix ADER-Banques Populaires *Entreprises-Innovation* qui récompense chaque année un travail de recherche alliant un laboratoire et une entreprise.

Les systèmes de synthèse assistée par ordinateur sont connus depuis une vingtaine d'années. Ils ont pour but d'aider les chimistes à trouver un ou plusieurs chemins de synthèse possibles pour une molécule donnée.

Schématiquement, on peut dire que les systèmes connus ont été le plus souvent développés d'un point de vue encyclopédique, la connaissance étant représentée de façon procédurale. S'ils sont performants au niveau tactique, ils le sont beaucoup moins au niveau stratégique en raison du manque de formalisation des stratégies de synthèse.

RESYN (REtroSYNthèse[3]) est un système d'aide à l'élaboration de plans de synthèse en Chimie Organique. Le but est de concevoir le plan de synthèse d'une molécule organique en s'appuyant sur une représentation à base d'objets et en combinant raisonnement par classification et raisonnement à partir de cas [1] [2].
De manière classique, le travail a comporté trois phases: (1) extraction de la connaissance d'un expert (E. Toromanoff), (2) modélisation de la connaissance et du raisonnement, (3) implémentation.

Relativement aux systèmes antérieurs, RESYN présente deux aspects spécifiques :

1. il est conçu pour résoudre des problèmes de synthèse organique industrielle.
2. il a été entièrement écrit dans un langage à base de frames (Y3), ce qui a permis de concevoir une base de connaissance structurée hiérarchiquement dont la mise à jour est aisée, et il fait appel à un mode de raisonnement par classification.

Depuis, RESYN sert de base à des travaux de fond sur la modélisation de la synthèse organique[4], et le rôle du raisonnement par cas dans l'élaboration d'un plan[2]. Un important travail de fond a été réalisé, permettant de dégager des principes régissant la conception de systèmes d'aide à la synthèse en chimie organique, et a conduit à la réalisation de prototypes nouveaux, dans lesquels la modélisation du raisonnement est poussée plus à fond[1].

Références

[1] A. Napoli, C. Laurenço, and R. Ducournau. An object-based representation system for organic synthesis planning. *International Journal of Human-Computer Studies*, 41(1/2):5–32, 1994.

[2] A. Napoli and J. Lieber. A First Study on Case-Based Planning in Organic Synthesis. In S. Wess, K.-D. Althoff, and M.M. Richter, editors, *Topics in Case-Based Reasoning – First European Workshop (EWCBR'93), Kaiserslautern*, Lecture Notes in Artificial Intelligence 837, pages 458–469. Springer Verlag, Berlin, 1994.

[3] J. Quinqueton, P. Vismara, J.C. Régin, M. Py, L. Lapied, and C. Laurenço. Resyn: un système d'aide à la synthèse en chimie organique. In *Systèmes Experts et Applications 12 Avignon'92*, pages 305–318, Juin 1992.

[4] Philippe Vismara and Joel Quinqueton. Design of planification strategies in organic chemistry. In *Proceedings of IASTED'95 (Int. Association of Science and Technology for Development Conference*, Innsbruck, 1995.

6.2 RODIN: Planification de réseaux

L'équipe de Michel Dao fait partie d'un groupement du CNET [4] qui développe des méthodes et des outils pour la planification de réseau. Cette équipe a la responsabilité du projet RODIN qui a pour objectif d'étudier, au travers notamment de la réalisation de prototypes, l'apport des techniques à objets (représentation, programmation, conception, bases de données) pour le dimensionnement des réseaux, qui constitue un des aspects de la planification.
Le support commun aux travaux du projet RODIN est CONRAD-OBJETS [1, 4] qui est un prototype d'outil de dimensionnement de réseaux téléphoniques commutés à acheminement adaptatif. Cet outil comporte une modélisation en classes du domaine concerné utilisant les traits de représentation des connaissances offerts par Y3 ; l'implémentation des algorithmes [5] de dimensionnement est réalisée de façon répartie dans les différentes classes, et le contrôle de l'exécution est en partie déporté dans certaines d'entre elles. CONRAD-OBJETS est développé avec CASSIS [2, 3] qui est un environnement de conception d'applications dédiées réseaux, développée au dessus de Y3.

4. Centre National d'Etudes des Télécommuncations, centre d'étude et de recherche de France Télécom
5. il s'agit d'algorithmes mettant en œuvre des résolutions d'équations reliant les divers paramètres du réseau dans un état stationnaire.

Deux activités de ce projet intéressent le groupe ELO :

- **Contraintes et objets**
  CONRAD-OBJETS gère un grand nombre de dépendances complexes entre les différentes propriétés (principalement numériques) des objets. L'utilisation de mécanismes de propagation de contraintes sur les propriétés d'objets offre une possibilité de gestion automatique de ces dépendances d'une manière plus déclarative. Des contraintes fonctionnelles unidirectionnelles de type agrégat sur les attributs d'objets sont maintenues paresseusement par des réflexes attachés à ces attributs. Ces travaux ont été effectués dans le cadre du stage de DEA de Stéphane Pelenc [5], et se prolongent actuellement par une thèse.

- **Représentation objets de comportements**
  L'implémentation d'algorithmes de dimensionnement de réseaux sur une représentation par objets de ceux-ci fait l'objet du travail de thèse de Milène Shteto au CNET. Ce travail consiste à concevoir deux taxonomies de classes du domaine étudié : une pour les différentes architectures de réseaux, et l'autre pour les méthodes (aux sens mathématique et algorithmique) de dimensionnement, et à étudier la manière de mettre en relation ces deux taxonomies, afin d'appliquer les comportements de la seconde sur la première.

Références

[1] M. Dao and M. Lebourges. Integration of Object-Oriented Programming and Distributed Algorithms for Network Design. In *Networks94*, pages 161–166. Hungarian Scientific Society for Telecommuncations, september 1994.

[2] M. Dao and J.-P. Richard. CASSIS: An Open Network Design Environment. In *NATO's Advanced Research Workshop on Topological Network Design*, june 1989.

[3] M. Dao and J.-P. Richard. CASSIS*: manuel de référence.* France Télécom CNET/PAA/ATR, 1.4 edition, june 1990.

[4] Milene Shteto and Michel Dao. An Object-Oriented Tool for Partial Network Sizing. In Michel Dao, editor, *Workshop on the Use of Object-Oriented Technology for Network Design and Management*, pages 75–85. ECOOP'94, july 1994.

[5] S. Pelenc. *Application de techniques de propagation de contraintes en langage à objets.* Mémoire de DEA, Université Montpellier II, LIRMM, 1994.

6.3 PRESAGE: Modélisation de réseaux de télécommunication

CNET(Paris)-ITESOFT(Nîmes), Presage : Un environnement orienté objet pour la modélisation de réseaux de télécommunication [1], (sept. 90, octobre 96).
Ce projet porte sur l'étude et le développement d'un environnement de conception et d'exécution de logiciel de planification de réseaux, il s'intéresse essentiellement à la partie descriptive du processus de planification de réseaux TAR (traffic- acheminement-routage).
Dans ce projet, les bases théoriques sont fondées sur des modèles complexes utilisant des représentations d'objets composites. Pour ce faire le noyau du langage **Yafool** a été étendu pour y intégrer les mécanismes de représentation et de gestion d'objets composites ainsi que les mécanismes de gestion des versions décrits précédemment (cf. section 3.2).

Références

[1] A. Caminada, C. Oussalah, Modelling concepts for telecommunications network planning. In Artificial Intelligence, Expert Systems and Symbolic Computing, Edited by : E.H. Houstis and J.R. Rice, Elsevier Science Pub., Amsterdam, The Netherlands(1992), pp 42-54.

Projet INTERLOO: interopérabilité

Responsable: Pierre Cointe

Objectifs:

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'inter-opérabilité. 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.

Sites et Participants:

- EMN - Ecole des Mines de Nantes:

  - Pierre Cointe (depuis juillet 92, professeur),
  - Isabelle Borne (depuis octobre 94, maître assistante),
  - Thomas Ledoux (depuis mai 94, doctorant, boursier CIFRE IBM),
  - Philippe Mulet (depuis octobre 92, doctorant boursier EMN),
  - Frédéric Rivard (depuis décembre 93, doctorant, boursier EMN),
  - Nick Edgar (depuis octobre 94, chercheur détaché, Object Technology International)

- Université de Nantes - ERTO (Equipe de Recherche en Technologies Objet):

  - Jean Bézivin (Professeur),
  - Frank Barbier (Maître de Conférences),
  - Richard Lemesle, Christophe Beaudet (dea),
  - Jérôme Lanneluc (Ingénieur société OTI, du 1/1/94 au 1/07/94),
  - Charles Nguyen (Ingénieur société Qualitech),
  - Régis Berteaud (CNAM),
  - ainsi que de nombreux étudiants du DESS de Nantes.

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

  - Roger Rousseau (Maître de Conférences),
  - Jean-Claude Boussard (Professeur),
  - Robert Chignoli (Maître de Conférences),
  - Philippe Lahire (Maître de Conférences, depuis octobre 1993),
  - Philippe Brissi (Doctorant, depuis octobre 1991),
  - Philippe Collet (Doctorant depuis octobre 1994),

- Jean-Marc Jugant (Ingénieur contractuel depuis août 1993 - contrat FAO).

- Mireille Blay-Fornarino, de l'équipe de Paul Franchi Zannettacci de l'I3S qui fait aussi partie du GDR de Programmation dans un autre pôle, a récemment souhaité son rattachement au pôle sur les objets. Ses travaux ne sont pas mentionnés dans ce rapport.

– Université Paris-6 (LAFORIA):

- Jean-François Perrot (Professeur),

- Gilles Blain (Maître de Conférences),

- François Pachet (Maître de Conférences).

7 Architectures réflexives et Interopérabilité

Les techniques de méta-représentation et d'auto-description sont utilisées depuis longtemps dans différents domaines de l'intelligence artificielle, elles sont apparues plus récemment (1987) dans le monde de la programmation par objets et sont aujourd'hui expérimentées pour construire et décrire des architectures objets extensibles et paramétrables. Ces expérimentations sont effectuées principalement dans les domaines des langages, des environnements de programmation, des nouveaux systèmes d'exploitation répartis et des interfaces homme-machine.
Une forme bien connue de réflexivité dans les langages à classes est la notion de méta-classe telle qu'elle est apparue en Smalltalk, puis en Loops et ObjVlisp.
Le système objet de CommonLisp (Clos) systématise cette approche en explicitant un protocole pour manipuler ses méta-objets (le "MOP" ou Meta Object Protocol). Ces méta-objets représentent non seulement les classes mais également les méthodes, les combinaisons de méthodes, les fonctions génériques et les définitions des slots.

ClassTalk est une extension de Smalltalk-80 conçue d'abord comme l'intégration à Smalltalk du modèle ObjVlisp. Ce projet a ensuite évolué vers la réalisation d'une bibliothèques de métaclasses significatives permettant de proposer 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, et de la résolution de l'envoi de messages.
Aujourd'hui, ClassTalk sert de base au développement d'un protocole de méta-objets pour Smalltalk. On trouvera dans [1] une comparaison détaillée de ClassTalk et du MOP de Clos. Ce protocole est utilisé et expérimenté par deux autres projets de l'équipe : les projets **ClassTalkAda9X** et **SomTalk**.
Récemment ClassTalk a été porté dans le système Envy, une plate-forme de développement Smalltalk multi-utilisateurs développée par la société canadienne Object Technology International (OTI).

7.1 Le projet ClassTalk-Ada9X

Ce projet mené en collaboration avec la société OTI consiste à définir en ClassTalk un sous-ensemble du langage Ada9X directement exécutable par la machine virtuelle Smalltalk.

Ce projet a trois buts. Il s'agit 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, il devient possible de les étendre pour réaliser un mécanisme d'inférence de types, proposer de nouvelles formes d'envoi de messages et finalement traiter directement des expressions Ada9x [4].
Par ailleurs, cette modélisation d'Ada9x est l'occasion d'enrichir de manière très significative les bibliothèques de méta-objets de ClassTalk. Cette réalisation devrait constituer l'une des partie importantes de la thèse de F. Rivard.

7.2 Le projet 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).

Par ailleurs, IBM viens de proposer le modèle SOM (a System Object Model) compatible avec IDL et avec CORBA dans sa version distribuée (DSOM). SOM est déjà disponible sur le marché sous la forme d'un produit (le SOMObjects Developper 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 doit également inter-opérer avec Smalltalk (en particulier grâce à sa nouvelle "incarnation" VisualAge).

D'un point de vue recherche, le modèle SOM est d'autant plus intéressant qu'il met en oeuvre une architecture de méta-objets très proche de celle d'ObjVlisp et 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 méta-classes explicites ainsi que du modèle d'héritage multiple de SOM.
Une première expérimentation de couplage SOM - ClassTalk a été réalisée par [3] dans le cadre de son projet de DESS. La suite de cette étude constituera une partie de la thèse de T. Ledoux.

7.3 Actions Industrielles

- [1993-] Object Technology International
  Intégration de ClassTalk dans le système ENVY (un environnement de développement Smalltalk multi-utilisateur)
  Utilisation d'EnvyClassTalk pour définir un prototype d'interprète Ada9X

- [1994-1996] IBM & AD-Consultants
  Application des protocoles de méta-objets et de la réflexion aux problèmes de l'inter-opérabilité entre ClassTalk et SOM.

7.4 Stages Dea

- Métaclasses explicites : application à la rétro-conception de bibliothèques Smalltalk
- Métaclasses explicites et Héritage : Définition de règles de compatibilité
- Architectures réparties et réflexion: application à la distribution d'objets Smalltalk

7.5 Thèses en cours

- P. Mulet: Réflexion et langages à prototypes: le langage Moostrap.
- F. Rivard, Réflexion et langages à classes: application à Smalltalk-80 et ClassTalk.
- T.Ledoux: Inter-opérabilité et langages à objets: application au couple SOM - ClassTalk.

Références

[1] 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.

[2] 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.

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

[4] Fréderic Rivard, "Extension du compilateur Smalltalk : application à la paramétrisation de l'envoi de message. A paraître dans les actes des JFLA'95.

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

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

8 Ateliers de Génie Logiciel et Interopérabilité

Dans le cadre de l'Équipe de Recherche en Technologie Objet de l'Université de Nantes (ERTO), l'AGL expérimental OSMOSIS, en cours de construction, est destiné à étudier la nature du cycle de vie à objets et les nombreux problèmes d'interopérabilité qui s'y posent.

Le cycle de vie à objets est considéré comme composé d'un certain nombre de processus construisant ou utilisant des produits ou modèles. Ces modèles sont souvent considérés comme similaires voire équivalents, car ils s'appuient en général sur un formalisme à objets. La propriété de continuité entre modèles (*seamlessness*) amène certains auteurs à considérer l'analyse, la conception et la mise en oeuvre comme des activités de nature identique se situant à des niveaux d'abstraction différents. Contrairement à cette approche, nous avons proposé dans [1] une définition originale du **processus de conception** considéré comme une composition d'objets métiers et d'objets techniques, opérée pour réaliser un but précis spécifié par un **script de besoins**.

À partir de ce travail, il est possible de recenser un grand nombre de modèles à objets dans le cycle de vie : de besoin, de domaine, de moyen, de conception, de mise en oeuvre, de test, de spécification formelle, etc. Le nombre de formalismes différents qu'il est nécessaire de faire cohabiter est déjà très important, même si l'on ne considère que les langages de programmation (par exemple Smalltalk, Eiffel, C++) et les méthodes d'analyse et de conception (par exemple OMT, OOA, HOOD, CRC, OBJECTORY, SYNTROPY, etc.).

Afin de pouvoir disposer de l'ensemble de ces formalismes de façon opérationnelle dans un même AGL, nous avons défini les **sNets**, une notation noyau basée sur les réseaux sémantiques [5]. Ce support nous permet de représenter des modèles composites (par exemple OMT ou JSD), et de préciser les relations entre différents modèles. Une plate-forme de manipulation de sNets a été construite. Elle est constituée de la **sMachine**, qui permet la gestion des versions et des entitś primitives et du **sBrowser** [2, 4] pour la visualisation et la modification des réseaux. A partir de cet environnement de travail, différentes études sont actuellement entreprises, en particulier :

1. L'étude du passage des spécifications informelles de besoins aux spécifications formelles. Nous nous appuyons pour ceci sur le concept de *cas d'utilisation* de Jacobson que nous avons étendu et enrichi [6].
2. Le remplacement, dans le sBrowser, de la notion de lien réel par celle de lien virtuel. Un lien virtuel correspond à un chemin dans le graphe et s'exprime sous forme d'une expression régulière.
3. La prise en compte, par le formalisme des sNets, de différents formalismes dans le cadre de l'AGL OSMOSIS, et leur évaluation comparée.
4. Dans le cadre du projet *Semantor*, mené en commun avec la société Qualitech, nous avons utilisé les mêmes outils pour la définition d'une démarche de rétro-conception a partir d'applications écrites en langage Cobol [5].

Références

[1] Barbier, F., Bézivin, J. Object-Oriented Design: The OSM Scheme TOOLS USA'93, Santa Barbara, (juillet 1993)

[2] Bézivin, J., Lanneluc, J. , Lemesle, R. Un réseau sémantique au coeur d'un AGL Conférence LMO'94, Grenoble, France, (septembre 1994)

[3] Bézivin, J., Lanneluc, J. , Lemesle, R. Representing Knowledge in the Object-Oriented Lifecycle TOOLS Pacific'94, Melbourne, Australie, (décembre 1994)

[4] Bézivin, J., Lanneluc, J. , Lemesle, R. sNets: the Core Formalism for an Object-Oriented CASE Tool Rapport interne ERTO, (novembre 1994)

[5] Bézivin, J., Lennon, Y., Nguyen Huu Nhon, C. From COBOL to OMT: A Reengineering Workbench Based on Semantic Networks Rapport interne ERTO, (novembre 1994)

[6] Bézivin, J. Object-Oriented Requirement Elicitation in the OSMOSIS project Rapport interne ERTO, (décembre 1994)

9 OCL – Objets et Composants Logiciels

L'équipe de l'I3S qui participe actuellement au pôle "programmation par objets" est l'équipe OCL – Objets et Composants Logiciels – du thème OLEM, l'un des 6 thèmes du laboratoire depuis sa réorganisation de juin 1994.

De façon générale, cette équipe travaille sur les *apports de l'approche pas objets pour le génie logiciel, étendu aux domaines des bases de données et de la CAO* [35]. Ses travaux portent sur le langage Eiffel et sont articulés autour des points suivants :

1. Évaluation du langage Eiffel, propositions d'améliorations du langage, de son exécutif et de sa bibliothèque de base [20, 22, 15].
2. Définition d'un modèle de persistance intégré au langage Eiffel et implémentation d'un prototype par un couplage fort Eiffel $\leftrightarrow$ O2 : projet **FLOO** (Contrat ESPRIT II 5311 "Business Class")[7, 8, 9, 18, 10, 11, 12, 19, 23]
3. Définition d'un modèle d'évolution de classes, d'instances persistantes, de bibliothèques et d'applications pour Eiffel et implémentation d'un système de réactualisation par réification : projet **IREC** (Contrat ESPRIT II 5311 "Business Class") [1, 2, 17, 3, 4, 5, 6, 21].
4. Définition d'un modèle à objets pour le génie logiciel, les bases de données et la CAO : projet **Goody** [14];
5. Conception et implémentation d'un logiciel de CAO appliqué à l'économétrie pour la prévision des besoins alimentaires des pays du Tiers Monde : projet **K2** (Contrat FAO/ONU) [16, 13, 25, 26, 27].
6. Évaluations et démonstrations de nos travaux et produits [1, 7, 19, 3, 12, 28, 6, 11]
7. Participation à des Comités de programme et à des jurys de thèse :
   Comités de programme : conférences Tools'Europe 93, Tools'Europe 94, LMO'94.
   Jurys de thèse : Occello 93, Courdier 93, Solnon 93, Poncelet 93, Collard 93, Fafiotte 94 (rapp.), Lejouad 94 (dir.), Lafourcade 94 (rapp.).
8. Diverses interventions et consultations sur la programmation par objets : [29, 32, 30, 31, 24, 33, 34, 35]

Nous ne développerons dans cette section que les activités plus directement liées au projet INTERLOO, essentiellement les trois projets FLOO, IREC et Goody [6]. La collaboration entre l'équipe OCL et le Laforia est traitée dans la section sur le projet IREC.

9.1 Projet FLOO

Dans le cadre du projet Esprit II "Business Class" (1991-1994), nous avons proposé un modèle de persistance intégré au langage Eiffel qui satisfait les règles d'Atkinson et d'autres nécessaires au génie logiciel pour les besoins de la réutilisation et de la fiabilité contrôlée par le typage, les assertions et les exceptions.
Ce modèle propose une gestion transparente des instances, sans modification du langage et sans la nécessité d'utiliser des classes ou des méthodes spécifiques pour les chargements des instances. Ceux-ci se font de manière automatique et incrémentale, soit de manière navigationnelle par redéfinition de l'opérateur d'accès (`a.x.g(k).h`...), soit par des requêtes sélectives unifiées avec les méthodes générales de parcours de collections génériques. Le statut de persistance s'obtient soit par des requêtes explicites sur des racines de persistance, soit implicitement par "contagion". La destruction physique des instances est retardée par les ramasse-miettes volatiles et persistants.
FLOO est un système transactionnel, basé sur une approche clients/serveurs pour prendre en compte les accès simultanés, le partage d'objets et la gestion des reprises en cas d'incidents. Par rapport à ODMG'93, notre modèle va plus loin dans le domaine de la transparence des accès, mais a nécessité une modification importante de l'exécutif Eiffel et plus légère du compilateur.
Le prototype que nous avons implémenté 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 en C puis en O2). Le bénéfice est cependant de permettre à des applications Eiffel d'inter-opérer, par le biais d'échanges d'instances, avec d'autres langages couplés à O2, notamment C++.
D'autre part, cette approche nous a conduit à traduire complètement les classes Eiffel dans les langages de classe et de méthode du système O2, nous confrontant à des problèmes non triviaux de traduction [7], notamment pour la généricité, qui est présente en Eiffel de manière contrainte et n'existe pas en O2.
Ce projet a conduit à la thèse de Philippe Lahire et se poursuit par l'étude de l'introduction des vues [23], qui donnent plus de souplesse que les classes pour la manipulation des données.

6. Le lecteur pourra consulter les rapports [19, 28] pour d'autres informations sur nos activités.
7. Essentiellement l'oeuvre de Jacques Farré de l'équipe "langages" de l'I3S

9.2 Projet IREC

Ce projet est le frère jumeau de FLOO pour "Business Class", avec comme objectif la gestion de l'évolution de classes et d'instances Eiffel.
L'étude de la littérature a mis en évidence qu'au delà des techniques de gestion de versions et de configurations, il fallait être capable de **réactualiser** les entités – classes clientes ou héritières, instances persistantes, applications ou bibliothèques... –, qui dépendent de classes en évolution, chaque fois que les changements leur sont perceptibles, même si cette perception peut être limitée par l'encapsulation ou les garanties de compatibilité offertes par l'héritage. Pour parvenir à automatiser le plus possible cette réactualisation, il faut disposer d'une base de connaissance suffisante pour **comprendre les changement**.
Dans le cadre du langage Eiffel où les assertions jouent un rôle essentiel pour les contrats de clientèle ou d'héritage, il faut être capable de les évaluer dynamiquement, pour vérifier, par exemple, que l'invariant d'une instance est encore valide lorsque sa classe a été modifiée. Nous avons donc décidé de réutiliser et d'adapter un interprète du langage Eiffel écrit en Eiffel dans notre équipe en 1991, lequel a donné à Bertrand Meyer l'idée de l'environnement mixte compilé/interprété utilisé dans son atelier Eiffel-Bench pour Eiffel 3.
Nous procédons par réification pour manipuler et réactualiser les classes. Notre réification n'utilise pas de méta-classe, concept absent de façon explicite en Eiffel, mais construit des vues complètes ou partielles selon divers points de vue, des classes porteuses, d'une part de toute l'information structurelle et sémantique, et d'autre part des méthodes pour comparer, réactualiser, expliciter les différences par des *"deltas sémantiques"*, évaluer, traduire..., bref toutes les manipulations que l'on peut envisager sur des composants logiciels.
Notre réification est écrite en Eiffel par notre équipe et en Smalltalk par l'équipe de Jean-François Perrot du Laforia, à partir d'IREC. Cette approche nous permettra dans le futur d'échanger des instances Eiffel entre les mondes Eiffel et Smalltalk, décrites dans un langage de bas niveau normalisé, par exemple ASN/1.
IREC est au centre de la thèse de Philippe Brissi dont la soutenance a été retardée pour des raisons de santé.

9.3 Projet IREC - Université Paris-VI

Le système IREC développé à Nice, a fait l'objet d'une réimplementation partielle en Smalltalk-80 dans l'équipe de J-F.Perrot, comme projet d'étudiants du DESS-GLA. Le but ultime de l'opération était de mettre en oeuvre les techniques de représentation de connaissances développées au LAFORIA (notamment le système de règles NeOpus) pour aborder des questions à fort contenu de connaissances dans le cadre du génie logiciel.

En fait, le projet n'a pu etre mené jusque-la, en partie par manque de sujets de ce genre : il aurait fallu une étude entièrement consacrée a la recherche et a l'élucidation de tels sujets avant de passer a l'expérimentation effective avec IREC-Stk.
Ce qui a ete réalisé a néanmoins illustré la faisabilité et l'intérêt de l'opération. La faisabilité en termes de réalisation en Smalltalk et de réutilisation de la partie frontale importée de Nice. L'intérêt par le simple avantage de disposer d'une batterie d'outils nouveaux, comme le couplage avec Word par liens DDE sur PC. Le système permet de lire le texte d'une classe Eiffel, d'en tirer la description de sa structure suivant la décomposition fixée par IREC, reproduite en Smalltalk, et de l'explorer de diverses manières à l'aide de l'arsenal Smalltalk. Malheureusement, par suite de problemes de santé de la part de certains etudiants de l'equipe, la redaction de la documentation afferente au systeme realisé laisse a desirer, et aucune publication n'est envisagee pour l'instant.

9.4 Projet Goody

Notre expérience des projets précédents FLOO et IREC, de CAO avec K2, le suivi de la thèse de Martine Collard[8] sur la définition d'un langage de règles pour étendre FLOO aux bases de données déductives et l'étude de divers modèles comme celui de ODMG'93, Cummins...nous ont convaincu que la définition d'un modèle à objets commun pour les domaines du génie logiciel, des bases de données et la CAO pouvait s'envisager. Nous avons participé au workshop OOPSLA'94 sur le thème des moteurs à objets [14] et avons entrepris cette année un cours de DEA sur le cahier des charges d'un tel modèle [35].
Suite aux travaux de Philippe Collet sur les améliorations de l'exécutif Eiffel pour le rendre plus efficace et introduire des facilités d'introspection par réification – notamment le parcours échantillonné des extensions de types –, nous étudions actuellement la possibilité de tester des assertions quantifiées existanciellement ou universellement. Comparé aux abondants travaux sur Larch / Smalltalk, Modula3, C++... notre approche

8. Équipe MECOSI de l'I3S

aura le mérite d'offrir des axiomes ou des assertions testables, ce qui nous parait indispensable pour révéler, non seulement les erreurs de sémantique, mais aussi celles qui ont trait à l'efficacité.

9.5 Actions Industrielles

- OTI & IBM AD Consultants (cf. section 7.3).
- [1992-] FAO/ONU
  Conception et implémentation d'un logiciel de CAO pour la prévision des besoins alimentaires des pays du Tiers Monde : projet K2. Ce produit, qui est encore en cours de développement (plusieurs centaines de classes, plus de 50 000 lignes de code Eiffel 3 spécifique) présente plusieurs intérêts pour tester en vraie grandeur nos solutions méthodologiques et techniques sur l'évolution et la réutilisation.

Références

[1] Brissi P. & Rousseau R. "Implementation of Views and Comparisons operators for an Objects and Classes Evolution Model". *Fourth Review Meeting*, Paris. Esprit II project N 5311 "Business class", 4–5 Mai 1993.

[2] Brissi P. & Rousseau R. "IREC: une représentation abstraite orientée objet pour manipuler des composants logiciels". Tech. rep. 93-39, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Juil. 1993, 13 p. *Journées du GDR Programmation (CNRS), Pôle objet.*, Université de Paris Sud, Orsay. 19–21 Oct. 1993. *Séminaire "objets"*, I3S, Sophia Antipolis, 2 Déc. 1993.

[3] Brissi P. & Rousseau R. "IREC project: evaluation and perspectives". *Final Review Meeting*, Paris. Esprit II project N 5311 "Business class", 7–11 Mars 1994.

[4] Brissi P. & Rousseau R. "IREC: an Object Oriented Abstract Representation to Handle Software Componants in a Persistant Framework". In *62nd Congress of ACFAS*, Montréal, 16–17 Mai 1994. World Scientific Publ. Co., 1994, pp. 79 – 93. "IREC: une représentation abstraite orientée objet pour manipuler des composants logiciels dans un cadre persistant". Tech. rep. 94-15, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Mars 1994, 14 p.

[5] Brissi P. & Rousseau R. "IREC: une représentation abstraite réifiée de classes Eiffel, pour la gestion de l'évolution de composants logiciels". *Séminaire du GT AFCET "Modélisation des objets en cours de conception"*, EPITA, Paris. 25 Mai 1994. *Séminaire ENSIMAG*, Grenoble. 12 Oct. 1994, 29 t.

[6] Brissi P. & Rousseau R. "IREC and classes evolution ; differences and adaptations.". *ECOOP – European Conference on Object Oriented Programming*, Bologna. 4–8 Juil. 1994. Selected demonstration.

[7] Chignoli R., Farré J. & Lahire P. "An example of object oriented inter-operability: Eiffel ↔ O2". *Fourth Review Meeting*, Paris. Esprit II project N 5311 "Business class", 4–5 Mai 1993.

[8] Chignoli R., Farré J., Lahire P. & Rousseau R. "Une implémentation de la persistance pour le langage Eiffel". In *RPO'93 – Représentation par Objets*, pp. 131–142, La Grande Motte, 22–23 Juin 1993. EC2 (Paris), Juin 1993, 2-906899-93-3. Tech. rep. 93-08, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Fév. 1993, 15 p.

[9] Chignoli R., Farré J., Lahire P. & Rousseau R. "FLOO: basic principles and illustration of automatic persistency mechanism for Eiffel". In *6th intern. conf on "Software engineering & its applications"*, pp. 181–190, Paris - La défense, 15–19 Nov. 1993. EC2 (Paris), Nov. 1993, 2-910085-00-7. "FLOO: principes et illustration des mécanismes automatiques de persistance pour Eiffel". Tech. rep. 93-40, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Juil. 1993, 15 p.

[10] Chignoli R., Farré J., Lahire P. & Rousseau R. "FLOO: a strong couplage between Eiffel language and O2 DBMS". In *62nd Congress of ACFAS*, Montréal, 16–17 Mai 1994. World Scientific Publ. Co., 1994, pp. 107–122. "FLOO: un couplage fort entre le langage Eiffel et le SGBD O2". Tech. rep. 94-16, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Mars 1994, 15 p.

[11] Chignoli R., Farré J., Lahire P. & Rousseau R. "FLOO : a strong coupling between Eiffel and O_2.". *ECOOP – European Conference on Object Oriented Programming*, Bologna. 4–8 Juil. 1994. Selected demonstration.

[12] Chignoli R., Lahire P. & Farré J. “FLOO project: evaluation and perspectives”. *Final Review Meeting*, Paris. Esprit II project N 5311 “Business class”, 7–11 Mars 1994.

[13] Lahire P. & Maetz M. “Object Oriented Design of the K2 System”. *FAO/ONU Review Meeting*, Nicosi, Chypre. May 15–18th 1993.

[14] Rousseau R., Brissi P. & Collet P. “From Eiffel to a General Object Model : our experience and suggestions”. Cummins F. A. (ed.), *OOPSLA'94 – The Ninth Annual ACM Conf. on Object-Oriented Programming Systems, Languages and Applications, Workshop on “The Object Engine: Foundation for Next Generation Architectures”*, pp. 29–37, Portland, (OR, USA), 23–27 Oct. 1994. 1994.

[15] Rousseau R. “Fallait-il concevoir Ada 9X?”. *Technique et Science Informatique*, vol. 13, no. 5 (Oct. 1994), pp. 723–729. Numéro spécial Ada.

[16] Boussard J.-C. & Lahire P. “Étude et conception du logiciel de CAO économétrique K2”. Tech. rep. 93-16, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Fév. 1993, 28 p.

[17] Brissi P. & Rousseau R. “IREC : user's guide. An object oriented abstract Representation for Software Components Evolution and General Handling”. Tech. rep. BC.R.LIS.W4.3.3, Release 1.1, Esprit II project N 5311 “Business class”, Feb. 1994.

[18] Chignoli R., Farré J., Lahire P. & Rousseau R. “FLOO: A prototype for an Eiffel Object Repository with O2 DBMS - User's guidé'. Tech. rep. BC.R.LIS.W4.1.3, Release 2.1, Esprit II project N 5311 “Business class”, Jan. 1994, 130 p.

[19] Chignoli R., Lahire P. & Rousseau R. “Technical review of basic object oriented tools and recommandations for further work”. Tech. rep. BC.R.SOL.W4, Release 1.0 (french), Release 1.1 (english), Esprit II project N 5311 “Business class”, Jan. 1994, 20 p.

[20] Collet P. *Mécanismes de l'Exécutif Eiffel ISE 2.3*. Rapport de stage, Maîtrise d'informatique, Oct. 1993, 33 p.

[21] Collet P. *Vers un modèle d'évolution de composants logiciels*. DEA d'informatique, UNSA – Univ. de Nice – Sophia Antipolis, I3S – URA CNRS 1376, Juin 1994, 70 p.

[22] Collet P. & Rousseau R. *Un exécutif efficace, avec facilités d'introspections pour Eiffel 2.3*. Rapport de recherche I3S, Déc. 1994.

[23] Michaudel É. *Un système de vues pour FLOO – Modélisation et application aux protections*. DEA d'informatique, UNSA – Univ. de Nice – Sophia Antipolis, I3S – URA CNRS 1376, Juin 1994, 50 p.

[24] Rousseau R., Lahire P. & Brissi P. “Reuse of Software Components requires quality and productivity : our experience with Eiffel.”. Tech. rep. 94-45, I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Mars 1994, 15 p.

[25] Jugant J.M. & Maetz M. “Specification of the SUA Module for the K2 project”. Tech. rep. FAO/I3S 94-1, Release 1.3, FAO - Food Agriculture Organisation of the United Nations, I3S – URA CNRS 1376 – Univ. de Nice-Sophia Antipolis, Oct. 1994.

[26] Jugant J.M. & Maetz M. “Specification of the Forest Module for the K2 project”. Tech. rep. FAO/I3S 94-2, Release 1.2, FAO - Food Agriculture Organisation of the United Nations, I3S – URA CNRS 1376 – Univ. de Nice-Sophia Antipolis, Nov. 1994.

[27] Lahire P. & Maetz M. “Specification of the Demand Module for the K2 project”. Tech. rep. FAO/I3S 94-3, Release 1.0, FAO - Food Agriculture Organisation of the United Nations, I3S – URA CNRS 1376 – Univ. de Nice-Sophia Antipolis, Dec. 1994.

[28] Rousseau R. “Rapport d'activité du thème OCL 1992-1994, Rapport d'activité du laboratoire I3S, Comité scientifique du 3 mai 1994”. Tech. rep., I3S – URA CNRS 1376 – Univ. de Nice – Sophia Antipolis, Fév. 1994, pp. 229-237. Démonstrations des systèmes IREC et FLOO.

Tutoriaux et écoles d'été

[29] Boussard J.-C. “Premières réflexions personnelles sur l'enseignement de la discipline informatiqué'. Tech. rep. 93-71, I3S – URA CNRS 1376 – Univ. de Nice - Sophia Antipolis, Mars 1993, 5 p.

[30] Boussard J.-C. “L’enseignement de la Programmation par Objets”. Journées nationales des MIAGE, 30-31 Mai 1994, IMAG, Grenoble.

[31] Boussard J.-C. “Et alors, l’Informatique en DEUG?”. Journées du CIES de l’Académie de Grenoble, 21-22 sept. 1994, IMAG, Grenoble.

[32] Rousseau R. “Enseignements & Recherches autour du langage Eiffel depuis six ans à Nice: Bilan & Perspectives”. *IFSIC – Univ. de Rennes 1*, IUT - Vannes, 8 Juil. 1993.

[33] Rousseau R. “Un panorama des langages à objets”. *Autour de l’objet – Méthodes de conception, langages, bases de données*, Nice. École d’été INA informatique / INRA, IUT de Nice, 4 Juil. 1994, 24 t.

[34] Rousseau R. “Eiffel 3 : un langage à objets pour la réutilisation”. *Autour de l’objet – Méthodes de conception, langages, bases de données*, Nice. École d’été INA informatique / INRA, IUT de Nice, 7–6 Juil. 1994, 122 t.

[35] Rousseau R., Boussard J.-C., Chignoli R., Collet P., Ducasse S., Gallésio É. & Lahire P. “Approches par objets pour le GL, les BD et la CAO : vers un modèle commun”. *Cours de DEA Informatique*, UNSA Univ. de Nice – Sophia Antipolis, I3S – URA CNRS 1376, Sophia Antipolis, Oct. – Déc. 1994.