INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team MIRÓ

Objects, Types, and Prototypes: Semantics and Validation

Sophia Antipolis - Lorraine

THEME 2A

Activity Report

2003

Table of contents

1. Team

Miró is a common project between l'INRIA Lorraine and l'INRIA Sophia Antipolis.

Head of project-team
Luigi Liquori [CR INRIA Lorraine]

Vice-head of project team
Joëlle Despeyroux [CR INRIA Sophia-Antipolis]

Ph.D. Students
Alberto Ciaffaglione [Co-tutelle INPL-Univ. Udine, -04/2003]
Benjamin Wack [UHP,-2004]

Administrative assistant
Nathalie Bellesso [INRIA Sophia Antipolis, à temps partiel]
Laurence Benini [INRIA Lorraine, à temps partiel]

2. Overall Objectives

2.1. Why

We consider Johachim Miró [24] as a great object-oriented painter. In fact, his paintings from the 1940-1960 period use numerous geometrical objects : points, colored points, squares, circles, windows, lines, curves, etc. All these elements are dear to object-oriented programming aficionados. We could even go as far as imagining the painter, who lived in France at Montmartre in the beginning of the 50s, probably inspired the conceptors of Simula, Smalltalk, and the community of computer scientists who studied the theoretical aspects of the object-oriented paradigm.

2.2. Context and Objectives

One of the goals of the Team consists in investigating the possibility to "reconcile" object-oriented programming and functional programming while keeping the spirit of the former and the elegance of the latter.

Object-oriented languages have become a major trend in large scale computer applications. This has made it necessary to study these languages both from a theoretical and a practical point of view, in order to better exhibit their fundamental characteristics and at the same time define new object oriented and concurrent languages able to combine expressiveness to security and efficiency. Our research belongs to this context.

We study type theory, new systems improving formal proofs, efficient compilation and execution of object-oriented languages, and object-oriented operating systems. We are interested in certifying the tools developed for these languages (interpreters, compilers, ...), using the Coq system as a favorite proof assistant.

Hence, our research program focuses mainly on the three following points :

- the study, definition and certified implementation of a class-based language called SmallTalk2K, and of a prototype-based language called FunTalk, used as intermediate language for the SmallTalk2K compiler, defined later in this document ;
- type theory for object-oriented languages and for proof assistants, certified software, rewriting and formal calculi which are the foundation of object-oriented, functional and concurrent languages.

2.3. The Team is going to close

The Team is going to close on 12/31/2003. This is our last research activity report. We would like to thanks all the people that supported us during those exciting (at least for me) three years. The external/internal referees reports of the Team are available on demand (for the non-INRIA) or by looking to the INRIA-Sophia internal server of the *"Comité des Projets"*

http://www-sop.inria.fr/interne/vie/comites/cp/AvantProjets/index.html

For further reading, the interested readers could refer to the following documents (in French):

- The official document of the Team
  http://www-sop.inria.fr/interne/vie/comites/cp/AvantProjets/Miro/MiroV1.3.ps.gz
- The 2002 and 2001 Research Activity Reports
  http://www.inria.fr/rapportsactivite/RA2002/miro/miro_tf.html
  and
  http://www.inria.fr/rapportsactivite/RA2001/miro/miro_tf.html

3. Scientific Foundations

3.1. Introduction

The scientific foundations of the Team can be found in

http://www.inria.fr/rapportsactivite/RA2002/miro/module6.html

4. Application Domains

4.1. Introduction

Key words: *PDA*, *PDA-OS*, *Mobile Phones*, *Mobile-OS*, *Safe programming*, *Certified Software*, *Mobile Code*, *Internet*, *Telecommunications*.

The application domains of the Team can be found in

http://www.inria.fr/rapportsactivite/RA2002/miro/module9.html

5. Software

5.1. Introduction

The description of the software developed inside the Team can be found in

http://www.inria.fr/rapportsactivite/RA2002/miro/module12.html
http://www.inria.fr/rapportsactivite/RA2002/miro/module13.html
http://www.inria.fr/rapportsactivite/RA2002/miro/module14.html

6. New Results

6.1. Introduction

N.B. Before enumerating the new results, the interested reader could have a look on our main research objectives in

http://www.inria.fr/rapportsactivite/RA2002/miro/module15.html

6.2. On Rewriting Calculus

Key words: *Type Theory*, *Calculus of Constructions*, *Rewriting-Calculus*, *Pattern-Matching*.

Participants: Luigi Liquori, Joëlle Despeyroux, Benjamin Wack, Claude Kirchner [Projet-Team Prothéo], Horatiu Cirstea [Projet-Team Prothéo], Bernard Serpette [Project-Team Oasis], Gilles Barthe [Team Everest].

6.2.1. Pure Pattern Type Systems

In [11], we introduce a new framework of algebraic pure type systems in which we consider rewrite rules as lambda terms with patterns and rewrite rule application as abstraction application with built-in matching facilities. This framework, that we call *"Pure Pattern Type Systems"*, is particularly well-suited for the foundations of programming (meta)languages and proof assistants since it provides in a fully unified setting higher-order capabilities and pattern matching ability together with powerful type systems. We prove some standard properties like confluence and subject reduction for the case of a syntactic theory and under a syntactical restriction over the shape of patterns. We also conjecture the strong normalization of typable terms. This work should be seen as a contribution to a formal connection between logics and rewriting, and a step towards new proof engines based on the Curry-Howard isomorphism.

6.2.2. Rewrite Strategies in the Rewriting Calculus

In [14], we present an overview on the use of the rewriting calculus to express rewrite strategies. We motivate first the use of rewrite strategies by examples in the **Elan** language [21]. We then show how this has been modeled in the initial version of the rewriting calculus and how the matching power of this framework facilitates the representation of powerful strategies.

6.2.3. An Imperative Rewriting Calculus

In [19], we present an imperative version of the Rewriting-calculus, a calculus based on pattern matching, pattern-abstraction and side-effects, which we call **imp**ρ.

We formulate a static and a *lazy call-by-value* dynamic semantics of **imp**ρ in the style of G. Kahn's *Natural Semantics*. The operational semantics is deterministic, and immediately suggests how to build an interpreter for the calculus. The static semantics is given via a first-order type-system based on a form of product-type, which can be assigned to **imp**ρ-terms, like structures (*i.e.* pairs).

The calculus is *à la* Church, in the sense that pattern-abstractions are decorated with the types of the free-variables of the pattern. The capability of safely access and modify a (monomorphic) typed store, and of defining fixed-points, makes **imp**ρ a good candidate for a core, or an intermediate language.

Properties like subject reduction and decidability of type checking are checked completely by a machine assisted approach, using the **Coq** proof assistant [23].

6.2.4. Rewriting Calculus with Fixpoints: Untyped and First-order Systems

In [17], we present a typed version of the rewriting calculus (ρ-calculus), a minimal framework embedding λ-calculus and Term Rewriting Systems, by allowing abstraction on variables and patterns. The higher-order mechanisms of the λ-calculus and the pattern matching facilities of the rewriting are then both available at the same level. Many type systems for the λ-calculus can be generalized to the ρ-calculus: in this paper, we study extensively a first-order ρ-calculus *à la* Church, called $\rho\rightarrow$. The calculus enjoys subject reduction, type uniqueness and decidability of typing.

The type system of $\rho\rightarrow$ allows one to type (object oriented flavored) fix-points, leading to an expressive and safe calculus. In particular, using pattern matching, one can encode and type-check term rewriting systems in a natural and automatic way. Therefore, we can see our framework as a starting point for the theoretical basis of a powerful typed rewriting-based language.

6.2.5. The Polymorphic Rewriting Calculus [Type Checking vs. Type Inference]

Many type systems for the λ-calculus can be generalized to the ρ-calculus: in [20], we study extensively a second-order ρ-calculus *à la* Church ($\rho 2^{\infty}$) that enjoys subject reduction, type uniqueness, and decidability of typing.

Then, we apply a classical erasing function to $\rho 2^{\infty}$ obtaining a corresponding type inference system *à la* Curry ($\rho\mathsf{F}^{\infty}$) that enjoys subject reduction. We discuss the two type systems *w.r.t.* a possible logic underneath the type systems via Curry-Howard Isomorphism.

Both systems can be considered as core calculi for polymorphic rewriting-based programming languages.

As "bonus-track", we present a variant of $\rho\mathsf{F}^{\infty}$ (called $\rho\mathsf{F}^{\infty}_{\mathsf{mlet}}$) featuring a principal type scheme, and we sketch an algorithm for type inference *à la* Damas-Milner-Tofte. We give answers to some conjectures presented last year by the first author.

6.3. On Object Calculus

Key words: *Interactive theorem proving*, *Logical foundations of programming*, *Program and system verification*, *Object-based calculi with side effects*, *Logical frameworks*.

Participants: Luigi Liquori, Joëlle Despeyroux, Alberto Ciaffaglione, Marino Miculan [Team SLP, Udine, Italy], Furio Honsell [Team SLP, Udine, Italy], Pietro Di Gianantonio [Team SLP, Udine, Italy].

6.3.1. Imperative Object-based Calculi in (Co)Inductive Type Theories

In [12], we discuss the formalization of Abadi and Cardelli's **imp**ς, a paradigmatic object-based calculus with types and side effects, in Co-Inductive Type Theories, such as the *Calculus of (Co)Inductive Constructions* ($\mathrm{CC}^{(\mathrm{Co})\mathrm{Ind}}$).

Instead of representing directly the original system "as it is", we reformulate its syntax and semantics bearing in mind the proof-theoretical features provided by the target metalanguage. On one hand, this methodology allows for a smoother implementation and treatment of the calculus in the metalanguage. On the other, it is possible to see the calculus from a new perspective, thus having the occasion to suggest original and cleaner presentations.

We give hence a new presentation of **imp**ς, exploiting *natural deduction semantics*, *(weak) higher-order abstract syntax*, and, for a significant fragment of the calculus, *coinductive* typing systems. This presentation is easier to use and implement than the original one, and the proofs of key metaproperties, *e.g.* subject reduction, are much simpler.

Although all proof developments have been carried out in the **Coq** system, the solutions we have devised in the encoding of and metareasoning on **imp**ς can be applied to other imperative calculi and proof environments with similar features.

6.3.2. Reasoning on an Imperative Object-based Calculus in Higher Order Abstract Syntax

In [13], we illustrate the benefits of using *Natural Deduction* in combination with *weak Higher-Order Abstract Syntax* for formalizing an object-based calculus with objects, cloning, method-update, types with subtyping, and side-effects, in inductive type theories such as the *Calculus of Inductive Constructions*. This setting suggests a clean and compact formalization of the syntax and semantics of the calculus, with an efficient management of method closures. Using our formalization and the *Theory of Contexts*, we can prove formally the Subject Reduction Theorem in the proof assistant **Coq**, with a relatively small overhead.

6.3.3. Foundations for Dynamic Object Re-classification

In [16], we investigate, in the context of *functional prototype-based languages*, objects which might extend themselves upon receiving a message. The possibility for an object of extending its own "self", referred to by Cardelli as a *self-inflicted* operation, is novel in the context of typed object-based languages. We present a sound type system for this calculus which guarantees that evaluating a well-typed expression will never yield a `message-not-found` run-time error. The resulting calculus appears to be also a good starting point for a rigorous mathematical analysis of class-based languages.

Our contribute can be viewed as a sound foundations for *dynamic object re-classification* underpinned on *functional programming paradigm*. Re-classification changes at run-time the class membership of an object while retaining its identity. An imperative approach to re-classification, suggesting language features, has been undertaken through the language `Fickle` [22].

The present work is based on the paper "A Lambda Calculus of Objects with Self-Inflicted Extension", by the last three authors, which appeared in Proceedings of ACM-SIGPLAN OOPSLA-98, International Symposium on Object Oriented, Programming, System, Languages and Applications, Vancouver, British Columbia, ACM Press, 1998. With respect to that contribution, in the present work the reduction semantics has been slightly changed, the type system refined and the proofs are fully documented.

7. Contracts and Grants with Industry

7.1. ACI Security

The Team participate to the ACI *Action Recherche Incitative "Modulogic: Atelier de construction modulaire de logiciels certifiés. Application aux politiques de sécurité"*. The official web site of the action is:

http://pauillac.inria.fr/modulogic/

7.2. ARC INRIA

The Team participate to the ARC *Action Recherche Coopérative "Concert: Compilateurs Certifiés"*. The official web site of the action is:

http://www-sop.inria.fr/lemme/concert/

The principal objective of the co-operative research project Concert is to determine, if it is feasible in the current state of knowledge, to produce a realistic compiler which is certified, *i.e.* accompanied by semantic a **Coq** proof of equivalence between the source code and the generated computer code.

8. Other Grants and Activities

8.1. European FP6 Projects

- The Team participates to the call for proposal IST Coordination Action (FET Open Scheme) Call FP6-2002-IST-C *Types for Proof and Programs : TYPES* in the 6th Framework Program of the European Union. J. Despeyroux is the local coordinator of the INRIA Sophia site. The full proposition can be found at:
  http://www.dcs.ed.ac.uk/~rap/TYPES03_DRAFT.ps
- The Team participates to the *"Appsem II: Applied Semantics"* working group in the 5th Framework Program of the European Union which officially started on 1 January 2003. The web site of the working group is:
  http://www.tcs.informatik.uni-muenchen.de/~mhofmann/appsem2/

8.2. Invitations

The Team invited the following researchers in 2003:

- May 2003 (14 DD) : Prof. Simona Ronchi della Rocca, University of Torino. (Talk: *A Typed Intersection Calculus*).
- December 2003 (1DD) : Prof. Stefane Ducassé, University of Berne, (Talk: *Reengineering and Traits*).

The complete list of visiting researcher of the Team can be found in

http://www.inria.fr/rapportsactivite/RA2002/miro/module29.html

9. Dissemination

9.1. Dissemination

L. Liquori was also the *"Publicity chair"* of the Team; the full list of presentations can be found in http://www.inria.fr/rapportsactivite/RA2002/miro/module30.html

A presentation of the Team can be found in http://www-sop.inria.fr/miro/Luigi.Liquori/Miro.pdf

9.2. Conferences, Talks, Invitations

During 2003:

- J. Despeyroux has participated to *ACM, PLI, Principles, Logics, and Implementations of High-Level Programming Languages*;
- L. Liquori has participated to *ACM POPL, Principle of Programming Languages* where he gave the talk *Pure Pattern Type Systems*;
- L. Liquori has visited for 5 days the *"Semantics and Logics of Computation"* group in Turin (Mariangiola Dezani, Stefano Berardi, Mario Coppo, Simona Ronchi della Rocca);
- L. Liquori has given talks at the University of Udine, at the University of Turin, at the École Normale Supérieure of Lyon, at LIX, and at the INRIA-Sophia: *"Pure Pattern Type Systems"*, and *"An Imperative Calculus"*;
- L. Liquori has given the talk *"Rho as (Meta) Logical Calculus and as (Multi) Paradigm Calculus"* at CNAM Paris, and at LIP6 Paris VI.

9.3. Researches Duties

The complete list of the researches duties of the Team can be found at: http://www.inria.fr/rapportsactivite/RA2002/miro/module32.html

9.4. Jury Member

L. Liquori participated as a jury member to the Ph.D. defense of A. Ciaffaglione, in Juin 2003, Udine, Italy.

9.5. Ph.D. and Stages

A. Ciaffaglione, defended his Ph.D. thesis *"Towards Certified Software for (manipulating) Reals and Objects"* in June 2003, Udine, Italy.

The complete list of Ph.D thesis and Stages of the Team can be found in http://www.inria.fr/rapportsactivite/RA2002/miro/module34.html

9.6. Teaching

- L. Liquori will be teaching from 5/01/04 to 12/03/04 the courses *"Functional Programming"* and *"Computability and Complexity"* at the Department of Informatics, University of Sussex, Brighton, UK (2nd and 3th year university courses, approx 60h).
- L. Liquori will teach from 9/8/04 to 20/8/04 in the 16th European Summer School in Logic, Language and Information Nancy, France, the Advanced Course *"The Rewriting Calculus"*.

10. Bibliography

Major publications by the team in recent years

[1] D. COLNET, L. LIQUORI. *Match-O, a Statically Safe (?) Dialect of Eiffel.* in « Proc. of TOOLS », IEEE Computer Society, 2000.

[2] J. DESPEYROUX. *Proof of translation in Natural Semantics.* in « Proc. of LICS », IEEE Computer Society, 1986.

[3] J. DESPEYROUX, A. FELTY, A. HIRSCHOWITZ. *Higher-order Abstract Syntax in Coq.* in « Proc. of TLCA », volume 902, Springer Verlag, pages 124–138, 1995.

[4] J. DESPEYROUX, P. LELEU. *Recursion over Objects of Functional Type.* in « Mathematical Structures in Computer Sciences », number 4, volume 11, 2001.

[5] P. D. GIANANTONIO, F. HONSELL, L. LIQUORI. *A Lambda Calculus of Objects with Self-inflicted Extension.* in « Proc. of OOPSLA », The ACM Press, pages 166-178, 1998.

[6] L. LIQUORI. *An Extended Theory of Primitive Objects: First Order System.* in « Proc. of ECOOP », series LNCS, volume 1241, Springer Verlag, pages 146-169, 1997.

[7] L. LIQUORI. *On Object Extension.* in « Proc. of ECOOP », series LNCS, volume 1445, Springer Verlag, pages 498-552, 1998.

[8] C. SCHÜRMANN, J. DESPEYROUX, F. PFENNING. *Primitive Recursion for Higher-Order Abstract Syntax.* in « Theoretical Computer Science », number 1-2, volume 266, 2001, pages 1-57.

[9] O. ZENDRA, D. COLNET, S. COLLIN. *Efficient Dynamic Dispatch without Virtual Function Tables. The SmallEiffel Compiler.*. in « Proc. of OOPSLA », volume 32(10), The ACM Press, pages 125-141, 1997.

Doctoral dissertations and “Habilitation” theses

[10] A. CIAFFAGLIONE. *Certified reasoning on Real Numbers and Objects in Co-inductive Type Theory.* Ph. D. Thesis, Dipartimento di Matematica e Informatica, Università di Udine, Italy and LORIA-INPL, Nancy, France, 2003.

Publications in Conferences and Workshops

[11] G. BARTHE, H. CIRSTEA, C. KIRCHNER, L. LIQUORI. *Pure Patterns Type Systems.* in « Proc. of POPL », The ACM Press, pages 250-261, 2003.

[12] A. CIAFFAGLIONE, L. LIQUORI, M. MICULAN. *Imperative Object-based Calculi in (Co)Inductive Type Theories.* in « Proc. of LPAR », series LNCS, volume 2850, Springer Verlag, pages 59-77, 2003.

[13] A. CIAFFAGLIONE, L. LIQUORI, M. MICULAN. *Reasoning on an Imperative Object-based Calculus in Higher Order Abstract Syntax.* in « Proc. of MERLIN », The ACM Digital Library, 2003, To appear.

[14] H. CIRSTEA, C. KIRCHNER, L. LIQUORI, B. WACK. *Rewrite Strategies in the Rewriting Calculus.* in « Proc. of WRS », series Electronic Notes in Theoretical Computer Science, 2003, To appear.

Internal Reports

[15] A. CIAFFAGLIONE, L. LIQUORI, M. MICULAN. *On the Formalization of Imperative Object-based Calculi in (Co)Inductive Type Theories.* Technical report, number RR-4812, INRIA, 2003, http://www.inria.fr/rrrt/rr-4812.html, Journal version of LPAR+MERLIN.

Miscellaneous

[16] A. CIAFFAGLIONE, P. D. GIANANTONIO, F. HONSELL, L. LIQUORI. *Foundations for Dynamic Object Reclassification.* 2003.

[17] H. CIRSTEA, L. LIQUORI, B. WACK. *Rewriting Calculus with Fixpoints: Untyped and First-order Systems.* 2003, Submitted.

[18] L. LIQUORI. *Book Review: Formal Methods for Open Object-Based Distributed Systems.* The Computer Journal, 46(6). Oxford University Press - British Computer Society., 2003.

[19] L. LIQUORI, B. P. SERPETTE. *An Imperative Rewriting Calculus.* 2003, Submitted.

[20] L. LIQUORI, B. WACK. *The Polymorphic Rewriting Calculus [Type Checking vs. Type Inference].* 2003, Submitted.

Bibliography in notes

[21] P. BOROVANSKÝ, C. KIRCHNER, H. KIRCHNER, P.-E. MOREAU, C. RINGEISSEN. *An Overview of ELAN.* in « Proc. of WRLA », volume 15, Electronic Notes in Theoretical Computer Science, 1998.

[22] S. DROSSOPOULOU, F. DAMIANI, M. DEZANI-CIANCAGLINI, P. GIANNINI. *More Dynamic Object Reclassification: FickleII.* in « ACM Transactions On Programming Languages and Systems », number 2, volume 24, 2002, pages 153-191.

[23] G. HUET, G. KAHN, C. PAULIN-MOHRING. *The Coq Proof Assistant - A tutorial, Version 6.1.* rapport technique, number 204, INRIA, 1997, http://www.inria.fr/rrrt/rt-0204.html, Version révisée distribuée avec Coq. http://coq.inria.fr/.

[24] WEB SITE FUNDACIÓ JOAN MIRÓ. 2002, http://www.info-france-usa.org/fr/.