Jornada de Seguimiento de Proyectos, 2004
Programa Nacional de Tecnologías InformÃ<ticas

MELODIAS: Logical Methods and Tools for the Design and Verification of Multiparadigm Software TIC2002-01167

Francisco Javier López Fraguas *
Departamento de Sistemas Informáticos y Programación, UCM Madrid

Abstract

The aim of the project MELODIAS is the development of logical methods and tools to give conceptual and instrumental support to the design, development and verification of multiparadigm declarative software systems.
The project started from the wide previous experience of the team in the area of formal methods and declarative programming languages, to which the team has been making for a long time significant theoretical and practical contributions, related in particular to the development of the systems *Toy* and *Maude*.
With this project we plan to continue with producing such contributions, in the following main aspects: Formal meta-tools for rewriting logic and the *Maude* language; Declarative debugging of constraint functional logic programs; Constructive finite failure in functional logic programming; Programming with hereditary Harrop formulas and constraints.
From the scientific and methodological point of views, the project covers in a well balanced way both theoretical aspects about the foundations of the new proposals, and practical aspects leading to their effective implementations.

1 Objectives and organization of the project

1.1 Introduction

Logic is at the core of many aspects of computer science. In particular, most of formal methods related to software production can be expressed in some kind of logical formalism. Sometimes logic give *conceptual support* to some aspects of software production. This happens, for instance, with declarative programming languages, whose essential notions and constructs are borrowed from logic. Some other uses give *instrumental support*. Examples of this could be the various logical techniques, like *model checking*, used for software verification.

Spanish research groups are well positioned in the field of logic in computer science. In particular, in our group we start from previous well known contributions to the field of declarative programming. We could mention:
• Functional logic programming: the CRWL logical framework and its implementation through the

*Email: `fraguas@sip.ucm.es`

Toy system.
• Rewriting logic and its implementation through the *Maude* system.
In this project we plan to continue with this contributions as explained in the rest of this document.

1.2 Objectives

The **main objective** of the project can be described as follows:

> *Development, from the point of views of both theoretical foundations and practical implementation, of logical methods and tools to give conceptual and instrumental support to the design, development and verification of multiparadigm declarative software systems.*

We now concrete this general aim for each of the four main scientific lines around which we have centered our work.

Line 1: Constructive failure in functional logic programs
Main goal: Theoretical foundations and implementation of a notion of constructive failure for functional logic programs, using as starting point the logical framework *CRWL* and the system *Toy*.
Subobjectives: Propose constructive (i.e., able to operate with logical variables) operational procedures for functional-logic programs using failure constructs; Relate the operational procedures with the declarative semantics of the programs; Implement the operational procedures, integrating it into the *Toy* system; Develop applications; Propose a model semantics.
Line 2: Declarative debugging of (constraint) functional logic programs
Main goal: Theoretical foundations and implementation of a scheme for declarative debugging of constraint functional logic programs, using as starting point the logical framework *CRWL* and the system *Toy*.
Subobjectives: Provide the theoretical foundations of a scheme for constraint functional logic programming; Define a scheme of declarative debugging of wrong answers and missing answers in constraint functional logic programming; Develop a prototype of declarative debugger for the *Toy* system; Evaluate the debugger and improve its efficiency.
Line 3: Formal methods and meta-tools for rewriting logic
Main goal: Development of formal methods based on meta-tools, using as starting point the theory of rewriting logic and the system *Maude*.
Subobjectives: Develop an environment for proving properties of membership equational logic theories; Develop an environment for proving properties of rewriting logic theories, in particular about modal and temporal properties; Propose and implement a framework for proving properties of functional logic programs in the *CRWL* framework.
Line 4: Development of the scheme *HH(C)*
Main goal: Theoretical foundations and implementation of the scheme *HH(C)*, designed for realized logic programming with hereditary Harrop formulae with constraints.
Subobjectives: Provide a declarative semantics to languages based on hereditary Harrop formulae with constraints; Prove correctness and completeness results for such semantics; Develop decision procedures for the satisfiability of constraints in domains useful in practice; Implement a prototype for *HH(C)*, and concrete constraint solvers acting as parameter C.

1.3 Task organization and scheduling

We have planned our work around the four main lines mentioned above, according to the scheduling in Fig. 1. Tasks are numbered as $\mathbf{T}n.i$, indicating the i-th task of Line n.

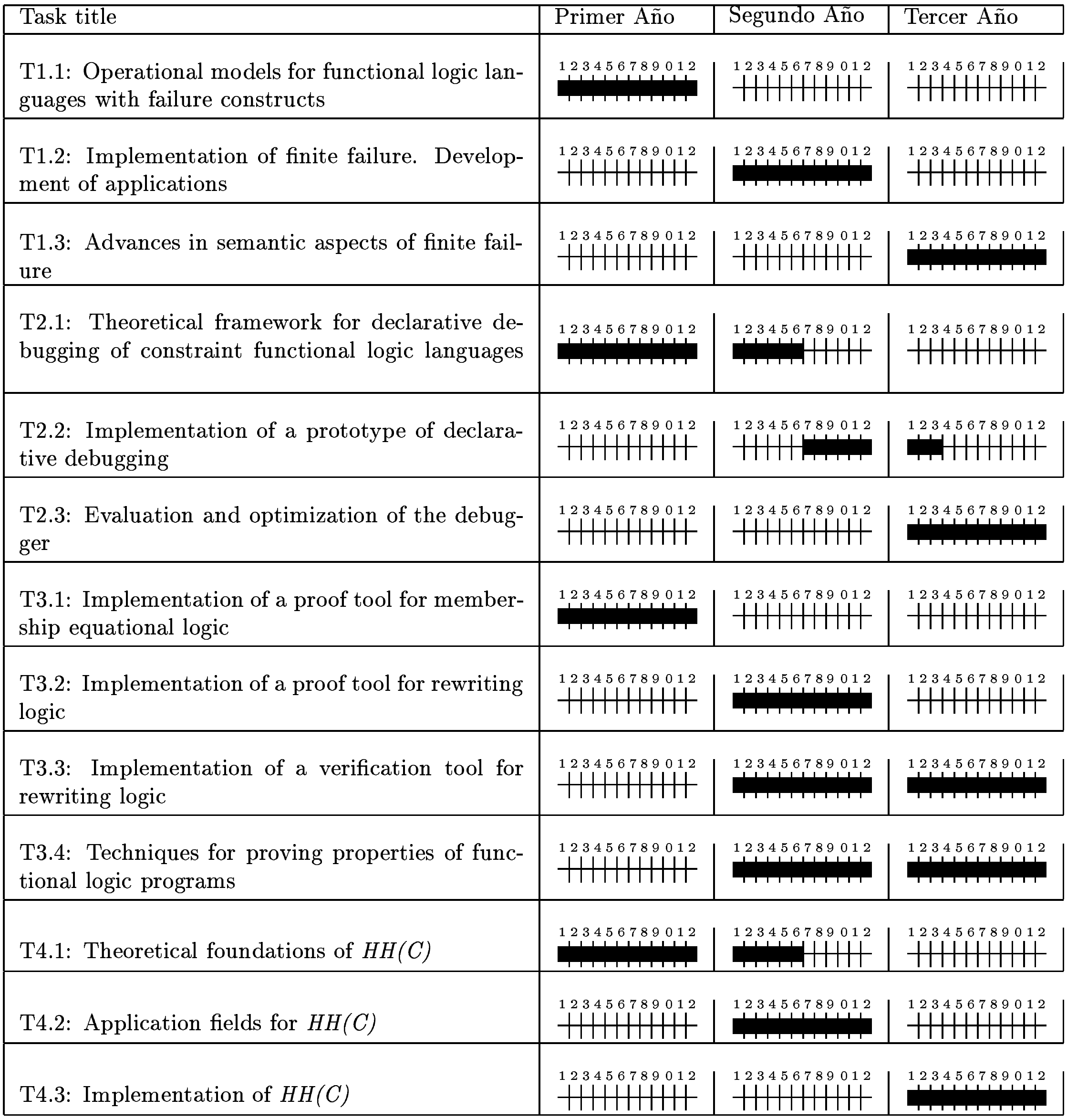

Figure 1: Task scheduling

2 Progress and achievements of the project

During the first two years of the project life, we have made significant progresses towards achieving our objectives, according quite closely to the planification of tasks. We describe them for each scientific line of the project.

Line 1: Constructive failure in functional logic programs

Operational models for constructive failure in functional logic programming We have proposed in [21] the novel notion of *set narrowing* as a computational mechanism to deal with a failure construct, even in presence of logical variables (i.e., constructive), in an ambient of functional logic programs with possibly non-strict non-deterministic functions with call-time choice semantics. We have proved correctness and completeness results with respect a proof theoretic semantics [24] of such programs. We have extended [22, 23] the operational model and the results by including a built-in equality function.

Implementations We have developed a prototype implementation of the above approach, the system *OOPS* [33], which is the first implementation of constructive failure in the functional logic field. The system includes a series of facilities: *program transformations* to make transparent to the user some cumbersome syntactic restrictions assumed by the set narrowing procedure, like the use of set-oriented syntax for expressions or the need that the programs must be overlapping inductively sequential; programming constructs like *default rules*, very useful in practice; a *trace facility* which has turned to be very interesting for experimenting with the prototype. The trace is generated as a Latex document for which a postscript file is created.

We have also partially integrated the implementation of failure in the *Toy* system, but in this case failure is not constructive. A fully integration is planned for the future.

Applications We have developed a collection of non-trivial examples showing the expressive power of failure as programming construct: default rules turn out to be very useful in many practical situations, and failure is in general useful in problems involving search or the need of collecting answers.

All the obtained results in this line can be also found in Jaime Sánchez's PhD thesis.

Line 2: Declarative debugging of (constraint) functional logic programs

A general scheme for constraint functional logic programs We have designed [25] a new scheme *CFLP(D)* to provide a formal framework for the declarative and operational semantics for constraint functional logic languages over a concrete computation domain D given as parameter. Many interesting domains including real numbers, pure Herbrand terms or finite domains fit well into our framework. In [26] we propose a correct and complete lazy narrowing calculus as operational semantics for the scheme, parameterized by a constraint solver to which we impose a set of natural restrictions.

A scheme for declarative debugging of (constraint) functional logic programs The declarative semantics of the scheme *CFLP(D)* leads naturally to a notion of intended model (which would correspond to least models in error free programs) and its proof theoretic face allow a natural definition of *positive proof tree.* Both notions are the basis for performing declarative debugging of *wrong answers* in constraint functional logic programs.

In a paper yet no published, we have also proposed a negative proof calculus providing *negative proof trees*, which will be the key for the declarative debugging of missing answers in the CFLP framework.

Implementation of a declarative debugger for functional logic programs We have implemented a quite sophisticated prototype for declarative debugging of wrong answers in functional logic programs with strict equality constraints. The debugger is based on a source-to-source program transformation, where the transformed program builds during execution (positive) proof trees for every evaluated expression, thus allowing the debugging of wrong answers. In its present state the debugger, called *DDT* [3], is embedded into the *Toy* system, and is equipped with a complex graphical interface enabling friendly browsing of proof trees. Different navigation strategies have been explored and implemented in *DDT*, and a certain degree of intelligence is ensured by recording and extracting logical consequences of the user answers in a debugging session, and also by the notion of 'reliable partial specification'. All the results related to the prototype can be found in Rafa Caballero's PhD thesis.

Line 3: Formal methods and meta-tools for rewriting logic

Implementation of a proof assistant tool for membership equational logic We have made significant advances in the implementation of the *ITP* tool [5]: the interface now admits parameterized modules; we have integrated into the rewriting engine of *ITP* decision procedures [9] for concrete domains, in particular for linear aritmethic expressions extended with uninterpreted functions.

In all cases the reflective design [1] of ITP, based on the underlying reflective capabilities of Maude have made things easier [8].

Applications of the ITP tool We have used ITP for the verification of imperative programs, following the methodology proposed J. Goguen and G. Malcolm in *Algebraic Semantics of Imperative Programs* (The MIT Press, 1996). Also, the ITP tool has been used in several academic courses, including the course *Program verification* teached by J. Meseguer at Urbana.

A verification logic for rewriting logic We have studied methods for proving properties of rewriting logic programs, based on the logic VLRL, an action modal logic where rewriting rules are captured as actions. In [31] the action formulae are represented as linear temporal logic (LTL) formulae, using a theory trnasformation and the model-checker of Maude is used to proved them. In [32], we present a technique for mechanical proofs of VLRL properties of rewriting logic proprams. Again, the reflective ability of Maude and its implementation in the Maude metalevel have been essential. In the next future we plan to integrate all these techniques into the ITP tool. We are in particular interested in inductive properties of operational semantics [34].

A logical basis for verification of functional logic programs [10, 11, 12, 13, 14] We have started from CRWL as an adequate logical framework for functional logic programming. We have then proposed a mapping from CRWL-programs to Horn programs, where there is a close correspondence between the initial CRWL-model of a CRWL-program and to least Herbrand models of its associated logic program. The properties of interest, which are those valid in the initial model, can be then proved in a first order logic setting. Due to Gödel arguments, the set of properties valid in initial models is typically not recursively enumerable, and therefore we have proposed increasing approximations: the logic program itself, its completion, the inductive extension of the completion and, interesting by its novelty, the result of axiomatizing the CRWL-derivability relation. Our proposal can be seen as the first approach to the verification of functional logic programs covering non-deterministic functions.

Line 4: Development of the scheme *HH(C)*

A declarative semantics for HH(C) We have proposed [17, 18, 19] a model theoretic semantics and a fixpoint semantics for any instance of the scheme *HH(C)* for constraint logic programming with hereditary Harrop formulae. To this purpose we have generalized the notion of model. that Miller introduced for the case of unconstrained Harrop formulae. We have proved correctness and completeness results for that semantics with respect to the uniform sequent calculus which is the operational basis for *HH(C)*.

Development of constraint solvers The scheme *HH(C)* could only be practical if interesting concrete instances of the parameter C are realized. In this sense, we have focused our work [15, 16] on a mixed domain, called RH, which combines real numbers with arithmetic constraints and free Herbrand terms. with unification and disunification constraints. The situation is more complex than in classical CLP due to the greater generality of Harrop with respect to Horn formulae. The sover combines two different techniques for quantifier elimination.

3 A summary of concrete results

3.1 Web page of the project

A Web page has been created, describing the project and containing links to all the publications and systems mentioned below. The URL address is `http://geminis.sip.ucm.es/ clavel/melodia`.

3.2 PhD Thesis

Three PhD Thesis have been defended during the project:
• Isabel Pita Andreu, *Técnicas de especificación formal de sistemas orientados a objetos basadas en lógica de reescritura*, Marzo 2003. Supervised by Narciso Martí Oliet.
• Rafael Caballero Roldán, *Técnicas de diagnóstico y depuración declarativa para lenguajes lógico-funcionales*, Junio 2004. Supervised by Mario Rodríguez Artalejo.
• Jaime Sánchez Hernández, *Una aproximación al fallo constructivo en programación declarativa multiparadigma*, Junio 2004. Supervised by Francisco Javier López Fraguas.

Two more PhD Thesis are currently under development:
• José Miguel Cleva Millor, *Un marco lógico para la prueba de propiedades de programas lógico-funcionales.* Supervised by Francisco Javier López Fraguas.
• Miguel García Díaz, *Fundamentos teóricos, implementaci'on y Un esquema para la programación lógica con fórmulas de Harrop hereditarias y restricciones*, Supervised by Susana Nieva Soto.

3.3 System implementations

- The ITP tool, an interactive proof assistant for various kinds of rewriting theories in the Maude system.
- The OOPS system, a prototype for constructive failure in functional logic programming.
- The *Toy* system, a more mature system for functional logic programming, now enhanced with a declarative, intelligent debugger with a graphical interface.

3.4 Collaboration with other groups

To develop its activity, the project team has continued or in some cases started its collaboration with other national and foreign groups.

• Narciso Martí and Manuel Clavel have continued their intense collaboration with the Maude group, in particular with **Francisco Durán (Univ. de Málaga)**, **Steven Eker, Patrick Lincoln** and **Carolyn Talcott (SRI International, California)**, and **José Meseguer (Univ. of Illinois at Urbana-Champaign)**. They have participated in design, implementation and user manual writing of the new versions of the Maude system.

• The implementation of the ITP tool has motivated the collaboration of Manuel Clavel with Prof. **Cesare Tinelli (Univ. of Iowa)** in the area of combination of decision procedures for proof assistants, with Prof. **Deepar Kapur (Univ. of New Mexico)**, in relation with inductive principles applicable to membership equational logic, and with the Computer Science Department of **University of Stanford**, to compare the ITP tool and other current proof tools using decision procedures. Manuel Clavel has continued also his collaboration with **David Basin (ETH Zurich)** who leades a team at the Information Security Group —one of the groups integrating the Zurich Information Security

Center (ZISC)—. They have worked on the mechanization of metalogical reasoning and its application to the verification of metaprograms. Since a visit of M. Clavel to Zurich they also collaborate now in the use of the ITP tool to the specification and analysis of RBAC systems (Role Based Access Control).

• Rafael Caballero has started a collaboration with the group of Prof. **Herbert Kuchen (Univ. Münster, Germany)** by means of two visits of several months. They have worked on a prototype for test case generation for Java programs, using constraint solving technology embedded in a Java symbolic virtual machine. They are presently trying to use that test case generation as a basis for the declarative debugging of Java programs. This constitute an interesting extension of the original aims of the project with respect to the issue of declarative debugging.

• Susana Nieva and Miguel García have been collaborating with Prof. **James Lipton**, formerly at the **Univ. of Wesleyan** but currently linked to the Univ. Politécnica of Madrid. Thay have been investigating higher order extensions of $HH(\mathcal{C})$ and suitable model semantics for it. To this purpose Miguel García visited Prof. Lipton at Wesleyan for some months.

3.5 Publications

In addition to the finished PhD thesis, the project has produced up to now about 30 scientific publications. Just for the pleasure of escaping from Zermelo-Fraenkel limitations, we have also included this report in it.

1. D. Basin, M. Clavel y J. Meseguer. *Reflective metalogical frameworks*, ACM Transactions on Computational Logic, vol. 5, núm. 3, pp. 528–576, ACM Press, Julio 2004.
2. R. Caballero and F.J. López-Fraguas. *Improving Deterministic Computations in Lazy Functional Logic Languages*. Journal of Functional and Logic Programming, Volume 2003, Special Issue 1, number 1, EAPLS, 23 pages, 2003.
3. R. Caballero and M. Rodríguez Artalejo. *DDT: a Declarative Debugging Tool for Functional-Logic Languages*. In Proc. 7th International Symposium on Functional and Logic Programming (FLOPS 2004). Springer LNCS 2998, pp 70–84, 2004.
4. C., R. Caballero-Roldán, R. A. Müller, and H. Kuchen. *Constraint Solving for Generating Glass-Box Test Cases*. Proc. 13th International Workshop on Functional and (Constraint) Logic Programming. Technical Report AIB-2004-05, University of Aachen, 2004, 19-32.
5. M. Clavel, *Strategies and User Interfaces in Maude (Extended Abstract)*, Proc. 3rd Int. Workshop on Reduction Strategies in Rewriting and Programming (WRS'03), Technical Report DSIC-II/14/03, DSIC, UPV, 2003, 89–92.
6. M. Clavel, F. Durán, S. Eker, et al., *The Maude 2.0 System*, Proc. 14th Int. Conference on Rewriting Techniques and Applications (RTA'03), Springer LNCS 2706, 2003, pp. 76–87.
7. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer y C. Talcott. *Maude Manual (Version 2.1)*, Marzo 2004. `http://maude.cs.uiuc.edu/manual`.
8. M. Clavel, N. Martí-Oliet y M. Palomino. *Formalizing and Proving Semantic Relations between Specifications by Reflection*. Proc. 10th International Conference on Algebraic Methodlogy and Software Technology (AMAST'04), Springer LNCS 3116, pp. 72–86, 2004.
9. M. Clavel, M. Palomino y J. Santa-Cruz. *Integrating Decision Procedures in Reflective Rewriting-Based Theorem Provers*, Proc. 4th International Workshop on Reduction Strategies in Rewriting and Programming (WRS'04), pp. 15–24, Technical Report, RWTH Aachen, Junio 2004. `http://aib.informatik.rwth-aachen.de/`.
10. J.M. Cleva, J. Leach, F.J. López-Fraguas.*Verificación de propiedades de programas lógico-funcionales*. III Jornadas de Programación y Lenguajes (PROLE'03). Comunicación.
11. J.M. Cleva, J. Leach, F.J. López-Fraguas.*Una aproximación lógica a la verificación de propiedades de programas lógico-funcionales*. To appear in PROLE'04.
12. J.M. Cleva, J. Leach, F.J. López-Fraguas. *A logical approach to the verification of functional-logic programs*. Proc. of the 13th International Workshop of Fucntional and (Constraint) Logic Programming, RWTH Aachen Technical Report, pp. 33–48, 2004

13. J.M. Cleva, J. Leach, F.J. López-Fraguas. *A logic programming approach to the verification of functional-logic programs*. Proc. ACM SIGPLAN Conf. on Principles and Practice of Declarative Programming (PPDP'04), ACM Press, pp. 9–19, 2004.
14. J.M. Cleva, J. Leach, F.J. López-Fraguas.*Bases lógicas para la verificación de programas lógico-funcionales*. Invited to be presented at MAT'2005, Valencia, January 2005.
15. M. García-Díaz and S. Nieva Soto, *Solving Mixed Quantifiers Constraints over a Domain Based on Real Numbers and Herbrand Terms*, Proc. 6th International Symposium on Functional and Logic Programming (FLOPS'02), Springer LNCS 2441, 2002, 103–118.
16. M. García-Dïiaz and S. Nieva Soto, *Solving Constraints for an Instance of an extended CLP language over a domain based on Real numbers and Herbrand terms*. Journal of Functional and Logic Programming, Volume 2003, Special Issue 1, number 2, EAPLS, 32 pages, 2003.
17. M. García-Díaz and S. Nieva. *A Fixed Point Semantics for an Extended CLP Language*. Proc. 13th International Workshop on Functional and (Constraint) Logic Programming (WFLP'04). H. Kuchen (ed). Technical report AIB-2004-05, 118–136. RWTH Aachen. 2004.
18. M. García-Díaz and S. Nieva. *Providing Declarative Semantics for HH Extended Constraint Logic Programs*. Proc. Sixth ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (PPDP'04), 55–66. ACM Press, 2004.
19. M. García-Díaz and S. Nieva. *Formalizing Two Fixed Point Semantics for HH(C)* . To appear in Journal of Functional and Logic Programming, 2004.
20. F.J. López Fraguas . *MELODIAS: Logical Methods and Tools for the Design and Verification of Multiparadigm Software*, Jornadas de seguimiento de proyectos, Plan Nacional de Tecnolog'ias Informáticas, Málaga, Noviembre 2004.
21. F.J. López Fraguas and J. Sánchez Hernández. *Narrowing failure in functional logic programming*, Proc. 6th International Symposium on Functional and Logic Programming (FLOPS'02), Springer LNCS 2441, 2002, 212–227.
22. F.J. López Fraguas and J. Sánchez Hernández. *Functional Logic Programming with Failure and Built-in Equality*, Proc. 12th Int'l Workshop on Functional and (Constraint) Logic Programming (WFLP'03), Technical Report DSIC-II/13/03, DSIC, UPV, 2003, 61–74.
23. F.J. López Fraguas and J. Sánchez Hernández. *Failure and equality in functional logic programming*, Elec. Notes on Theor. Comp. Science, Vol 86, Issue 3, 21 pages, November 2003.
24. F.J. López Fraguas and J. Sánchez Hernández. *A proof theoretic approach to failure in functional logic programming. Theory and Practice of Logic Programming* 4 (1& 2):41-74, 2004.
25. F.J. López-Fraguas, M. Rodríguez-Artalejo and R. del Vado-Virseda *Constraint Functional Logic Programming Revisited*. In Proc. of the 5th International Workshop on Rewriting Logic and its Applications (WRLA'2004). To appear in Elec. Notes in Theor. Comp. Sci., 46 pp., 2004
26. F.J. López-Fraguas, M. Rodríguez-Artalejo and R. del Vado-Virseda *A Lazy Narrowing Calculus for Declarative Constraint Programming*. In Proc. ACM SIGPLAN Conf. on Principles and Practice of Declarative Programming (PPDP'04), pp. 43–54, ACM Press, 2004.
27. N. Martí-Oliet, M. Palomino y A. Verdejo. *A tutorial on specifying data structures in Maude*, To appear in PROLE'04.
28. J. Meseguer, N. Martí-Oliet y M. Palomino. *Theoroidal Maps as Algebraic Simulations*. Selected papers of WADT'04, Springer LNCS, 2004, Volume number to be assigned.
29. J. Meseguer, N. Martí-Oliet y M. Palomino. "Algebraic Stuttering Simulations", submitted.
30. J. Meseguer, N. Martí-Oliet y M. Palomino. *Algebraic Simulations*, draft in preparation.
31. M. Palomino y I. Pita. *Proving VLRL Action Properties with the Maude Model Checker*. En Fifth International Workshop on Rewriting Logic and its Applications (WRLA 2004), Barcelona, 27–28 Marzo 2004, to appear in *Electronic Notes in Theoretical Computer Science*, Elsevier.
32. M. Palomino y I. Pita. *Proving modal properties of rewrite theories using Maude's metalevel*, to appear in PROLE'04.
33. J. Sánchez-Hernández. *OOPS: An Implementation of Constructive Failure in functional Logic Programming*, submitted.
34. A. Verdejo y N. Martí-Oliet. *Executable Structural Operational Semantics in Maude*, to appear in *Journal of Logic and Algebraic Programming*, Elsevier, 2004.