A. Pettorossi (Ed.)

Meta-Programming in Logic

Third International Workshop, META-92
Uppsala, Sweden, June 10-12, 1992
Proceedings

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Series Editors

Gerhard Goos
Universität Karlsruhe
Postfach 69 80
Vincenz-Priessnitz-Straße 1
W-7500 Karlsruhe, FRG

Juris Hartmanis
Cornell University
Department of Computer Science
4130 Upson Hall
Ithaca, NY 14853, USA

Volume Editor

Alberto Pettorossi
University of Rome Tor Vergata, Via della Ricerca Scientifica
I-00133 Roma, Italy

CR Subject Classification (1991): I.2.2-4, F.4.1

ISBN 3-540-56282-6 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-56282-6 Springer-Verlag New York Berlin Heidelberg

Typesetting: Camera ready by author/editor
Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
45/3140-543210 - Printed on acid-free paper

Preface

Meta-programming is a technique which is widely used in logic programming. It is also an important technique for other programming paradigms and it has been present in various areas of computer science throughout the history of its development.

Meta-programming can be understood as the treatment of programs as objects. Thus, the various methodologies which refer to program transformation, program analysis, and program manipulation for imperative, functional, and logic languages, are all included in the area of meta-programming.

It is assumed that the programs under consideration have an associated semantics. This hypothesis is essential, because program manipulations and program transformations are meaningful only if they are performed while preserving the semantic values.

In the case of logic languages where computations and programs can naturally be viewed as proofs (recall the 'programs as proofs' paradigm), one can consider the techniques for representing and manipulating proofs to be an essential part of the meta-programming idea.

In particular, one may refer to proofs by using the 'demo' predicate, which can be decorated with suitable arguments for explicitly describing the relevant properties of the proofs in hand.

Proofs may belong to different logical theories. One may stratify those theories in a hierarchy or one may amalgamate them into a unique theory. There are advantages and disadvantages for either choice: the best solution very much depends on the application domain.

In both cases one may refer to a proof in a given logical theory by a name (or a term). The reader will find in this book some papers which address this 'naming problem' and propose some solutions. The origin of the naming problem is linked to Gödel's incompleteness theorem, which uses a naming technique for encoding logical formulas.

A particularly interesting field where logical theories are used is knowledge representation, where it is often required to represent what agents know about their own knowledge and the knowledge of other agents. This 'reflexive' power can easily be encoded into suitable logic programs, provided that the language includes some meta-programming features.

It is not required to use modal theories to represent these situations in a natural way. Some of these issues have been considered in the invited lectures and in other papers of these proceedings.

Foreword

This volume contains the papers presented at the Third International Workshop on "Meta-Programming in Logic" held at the Department of Computer Science of the University of Uppsala (Sweden), 10-12 June, 1992. This workshop is the successor of the ones organized by John Lloyd in Bristol, U.K. (June 1988) and by Maurice Bruynooghe in Leuven, Belgium (April 1990). This volume also includes the invited lectures and the advanced tutorials from the workshop.

The Programme Committee received 35 papers, from which 18 were chosen for a long presentation and 6 for a short one. The papers give an interesting and stimulating view of the major topics under investigation on: i) the *foundations and applications of meta-programming and transformational programming*, ii) the *design and implementation of language facilities for meta-programming*, and iii) *knowledge representation and meta-programming.*

Unfortunately, we were not able to include in this volume the following 6 papers relative to the short presentations:
i) F. Giunchiglia and L. Serafini (IRST, Povo, Italy): "Hierarchical Meta-Logics (or: How We Can Do Without Modal Logics)", ii) K. Hinkelmann (DFKI, Kaiserslautern, Germany): "Forward Logic Evaluation: Developing a Compiler from a Partially Evaluated Meta-Interpreter", iii) R. M. Jones (Optismsoft, Wegberg, Germany): "Why Must the King Speak Out?", iv) H. Ohnishi and S. Akama (Asao-ku, Kawasaki-shi, Japan): "Indexed Knowledge in Epistemic Logic Programming", v) J. S. Santibáñez (A.I. Dept., Polytechic University Madrid, Spain): "A Formal Model for Temporal Knowledge Based Systems Verification", and vi) D. G. Schwartz (Dept. Computer Engineering, CWRU, Cleveland, Ohio): "Metaprograms: the Glue to Integrate and Control Blackboard Knowledge Sources".

Their abstracts have been published in a document of the Computing Science Department of the University of Uppsala (Sweden).

I would like to thank the members of the Programme Committee who carefully read the submitted papers with the help of external referees and reported on time their comments and opinions, and in particular those who attended the Programme Committee meeting held in Rome at the beginning of February 1992. Special thanks also go to Maurice Bruynooghe and Robert Kowalski who supported my work with useful advice and suggestions.

I acknowledge the dedicated, patient, and skilful cooperation of Maurizio Proietti.

I am grateful to Jonas Barklund and the members of the Organizing Committee for their efforts in making the workshop possible and preparing this volume. They were very good at overcoming the inevitable difficulties due to the distance between Rome and Uppsala, which sometimes made things not so easy.

On behalf of the participants and the Programme Committee I would like also to thank the invited speakers, who delivered very interesting and stimulating lectures.

Finally, I want to thank the Department of Electronic Engineering of the University of Rome 'Tor Vergata' (Italy), the IASI Institute of the Italian National Research Council in Rome (Italy), and the Department of Computer Science of Uppsala University (Sweden) for their financial support and for providing the necessary facilities.

Alberto Pettorossi

Acknowledgements

The Meta 92 Workshop has been made possible through generous financial support from several sponsors:
- the Swedish Board for Technical Development (NUTEK),
- Uppsala University,
- the Association for Logic Programming,
- Swedish Institute of Computer Science (SICS), and
- SUN Microsystems AB.

The help from Civildatalogförbundet is also appreciated.

Being the chairman for the workshop, I find this an appropriate opportunity also to thank the other members of the Organizing Committee for the considerable time they have spent for preparing the workshop. I hope our combined efforts have produced a stimulating setting for the workshop.

The success of this workshop is of course mostly dependent on the quality of the program. The members of the Programme Committee have made a good effort to judge each submission carefully and to include as many significant contributions as time could allow.

In particular, we owe much to the programme chairman for the workshop, Alberto Pettorossi, and to his colleague Maurizio Proietti for hosting the Programme Committee meeting and for careful editing of the contributions.

Finally, I would like to thank Danny De Schreye for sharing valuable experiences from the preceding workshop in this series.

Jonas Barklund

Program Committee

Organizing Committee

List of Referees

This book also includes papers on: i) logical foundations of meta-programming, ii) model-theoretic and proof-theoretic problems, iii) analysis and transformation of logic programs, iv) use of meta-programming for deductive databases, and v) implementation aspects related to meta-programming, like modularization, compiler optimization, process communication, and object-orientation.

Meta-programming is not a special feature of logic programming. As we already said, it has been part of computer science since its very beginning. Indeed, if meta-programming is considered to be manipulation of programs as objects, one can say that it was already required when writing programs for the von Neumann computer. For instance, if we want to compute the sum of k integers which are assumed to be stored from memory location i to memory location i+k–1, and we assume that the values of k and i are known at run time only, it is necessary to write a program whose instructions manipulate the program itself (if indirect addressing and index registers are not available). Some of the instructions which will be executed at run time, are themselves obtained as the result of a computation.

Also interpretation or compilation can be viewed as instances of meta-programming. In both cases the executable program in machine language is the result of a computation over the source program given as input.

Abstraction mechanisms present in some programming languages, like i) making a procedure, say 'λx.x+1', out of an expression, say 'x+1', or ii) making a block, say 'begin S1; S2 end', out of a sequence of statements, say 'S1; S2', can be viewed as meta-programming techniques.

We do not have here the space to thoroughly explore many other programming techniques (such as partial evaluation) and examine their relationship to meta-programming. However, we want to remark that meta-programming is also present in functional languages, in particular if they allow a higher-order type discipline. In that case, in fact, one may obtain a function to be used as the result of the application of a higher-order function to an argument which itself is a function. Obviously, higher-order functions are extremely useful for conciseness and clarity. A standard example is the function composition operator, which is very often used in functional programming.

Rome, Italy
September 1992

Alberto Pettorossi

Table of Contents